Nr. 2 DM 14,80 • ÖS 124 • SFR 14,80

C16 / P4-SPECIAL

C16 - P4 SPECIAL

Programmieren wie ein Profi:

Eingabefelder Eingabemasken

Super-Listings

Basicerweiterung:

23 neue Befehle

Neu! Jetzt mit farbigem Spiele-Magazin

EDITORIAL

C16-Plus 4 Intern

Wer der Meinung ist, mit einem C16 oder Plus4 lasse sich nicht professionell arbeiten, der hat C16-P4-SPEZIAL noch nicht gelesen. Sie finden hier nicht nur Programme, sondern auch das Know-how, um selbst solche zu erstellen. Die Beiträge PROGRAMMIEREN WIE EIN PROFI, EINGABE FELD MIT PFIFF, DIE MASKE MACHT MUSIK und TEXTEN WIE DIE PROFIS in dieser Ausgabe von C16-P4-SPEZIAL zeigen Ihnen, wie Sie durch Ausschöpfen der Möglichkeiten Ihres Rechners Anwendungen selbst erstellen können, die den Vergleich mit gekauften Programmen keineswegs zu scheuen brauchen. Daß der C16 keineswegs nur ein Spielecomputer ist, sehen Sie bereits im Inhaltsverzeichnis: Die Anzahl der Anwenderprogramme hat beträchtlich zugenommen.
Im Ergebnis unserer Fragebogenaktion zeigte sich, daß Sie gerne mehr Anwendungen in unseren Heften sehen würden. Mit unseren Programmen in diesem Heft können Sie Ihren Rechner zur Buchführung, zur Kontrolle der Ausgaben für Ihr Fahrzeug, zur Übersicht über alle möglichen Daten benützen. Ein spezielles Druckprogramm, das ein Disketten-Inhaltsverzeichnis auf das richtige Format bringt, damit es leicht auf einer Diskettenhülle unterzubringen ist, hilft Ihnen, Ihre Ordnung besser zu verwalten. Wenn Sie sich vom Arbeiten entspannen möchten, so finden Sie dafür auch eine Anzahl von Spielen.
Sollten Sie einen besonderen Wunsch haben, von dessen Erfüllung auch andere profitieren können, so haben wir immer ein Ohr für Sie offen.
Viele Soft- und Hardware-Entwicklungen erfolgten bereits in der Vergangenheit nach Leserwünschen. Der Checksummer erleichtert die Listing-Eingabe, der Checkmon das Eingeben von Maschinenprogrammen. Unser Zeichensatzprogramm verhilft Ihrem Rechner und Drucker zu deutschen Umlauten. Damit Sie Grafiken nicht nur auf dem Bildschirm betrachten können, hilft Ihnen unser Hardcopy-Programm, diese dauerhaft auf Papier zu bringen. Ein Centronics-Kabel für den Plus4 und ein Centronics-Interface für den C16, C116 und den Plus4 ermöglicht Ihnen den Anschluß hochwertiger Drucker.
Auch künftig sind wir für Ihre Wünsche und Anregungen offen. Bitte verstehen Sie jedoch, daß die Entwicklung eines Programms oder einer Hardware manchmal etwas dauert.

COMMODORE WELT

TEST & TECHNIK

REPORT

GRUNDLAGEN

LISTINGS

UTILITIES, ANWENDUNGEN

INHALT

SPIELE

SERIE & SERVICE

TIPS & TRICKS

STÄNDIGE RUBRIKEN

LOAD & RUN

PC Pilotprojekt im Bayerischen Wald

Haben Sie ein Zimmer frei?

In Zwiesel vermittelt der Computer

Meldezettel bearbeiten, Zimmer vermitteln — der Kurverwaltung des bayerischen Städtchens Zwiesel wuchs der Verwaltungsaufwand über den Kopf. Ein EDV-Pilotprojekt, unterstützt von einem namhaften Computer-Hersteller, schafft Abhilfe.

Natur erleben und genießen" – die Werbeprospekte des Luftkurorts Zwiesel präsentieren die Vorzüge der Umgebung, auf Hochglanz, in den schönsten Farben. Hier, im Hinteren Bayerischen Wald, sollte man glauben, ist die Welt noch in Ordnung.

Doch die Idylle trügt. Das leicht verschlafen wirkende Kleinstädtchen ist putzmunter. Seit einigen Monaten rumort es im Rathaus: Die Zwieseler sind auf den PC gekommen. Die Kurverwaltung war es leid, mit umständlichen Arbeitsvorgängen wie der Zimmer-Vermittlung kostbare Zeit zu verplempern. Mit einem passenden EDV-System, so die Idee, müßte sich doch vieles schneller erledigen lassen.

Unterstützung hat man bei Commodore gefunden. Der Frankfurter Computer-Hersteller war interessiert und lieferte die Hardware, – leihweise, mit der Möglichkeit zum späteren Kauf. Das

Startschuß mit Musikeinlage

„Pilotprojekt Zimmervermittlungs-System" war geboren.

„Angefangen hat es auf der Münchner Systems im Oktober letzten Jahres", erzählt Gerald Hahn, Commodores Pressechef. „Die Zwieseler kamen mit Bier und Blaskapelle an unseren Stand". Das war der offizielle Start für das Projekt, besiegelt wurde die Zusammenarbei mit der feierlichen Übergabe einer Tastatur.

„Die Konkurrenz nebenan hat sich mächtig über den Rummel geärgert, den wir auf der Messe veranstaltet haben". Rainer Bleneke, Leiter der Kurverwaltung Zwiesel, gibt amüsiert die Geschichte zum besten: Schaulustige hätten sich in Massen um den Commodorestand gedrängt. Auch die Messeleitung sei nicht erbaut gewesen, denn die volkstümliche Musikeinlage wurde vorher mit niemandem abgesprochen.

Die Kontakte zwischen Commodore und Zwiesel bestehen bereits seit 1976. Alle drei Jahre veranstaltet der Kurort am Hausberg Arber ein Ski-Weltcup-Rennen. Dieser Arber-Kristall-Pokal wird von den Frankfurtern gesponsert. Was lag näher, als diese Beziehungen zu nützen?

Blencke ist bereits seit 1986 mit Commodore im Gespräch, das Projekt ist im Grunde sein „Kind": „Das wäre doch ein Bombenmarkt". – Vor allem für die Computer-Industrie. In der Tat gibt es bislang noch keine Hard- und Software-Pakete, die speziell auf die Bedürfnisse von kleineren Kurverwaltungen zugeschnitten wären. So wird denn, laut Hahn, ein Projekt dieser Art „erstmals in der BRD durchgeführt".

Ein Problem war allerdings die Software. An komplizierten Geschichten war man nicht interessiert. „Das Programm sollte simpel sein", sagt Blencke, der „von EDV so gut wie keine Ahnung" hat.

Schließlich hat man mit der Münchner Software-Firma Nexus einen

Zimmer-vermittlung maßgeschneidert

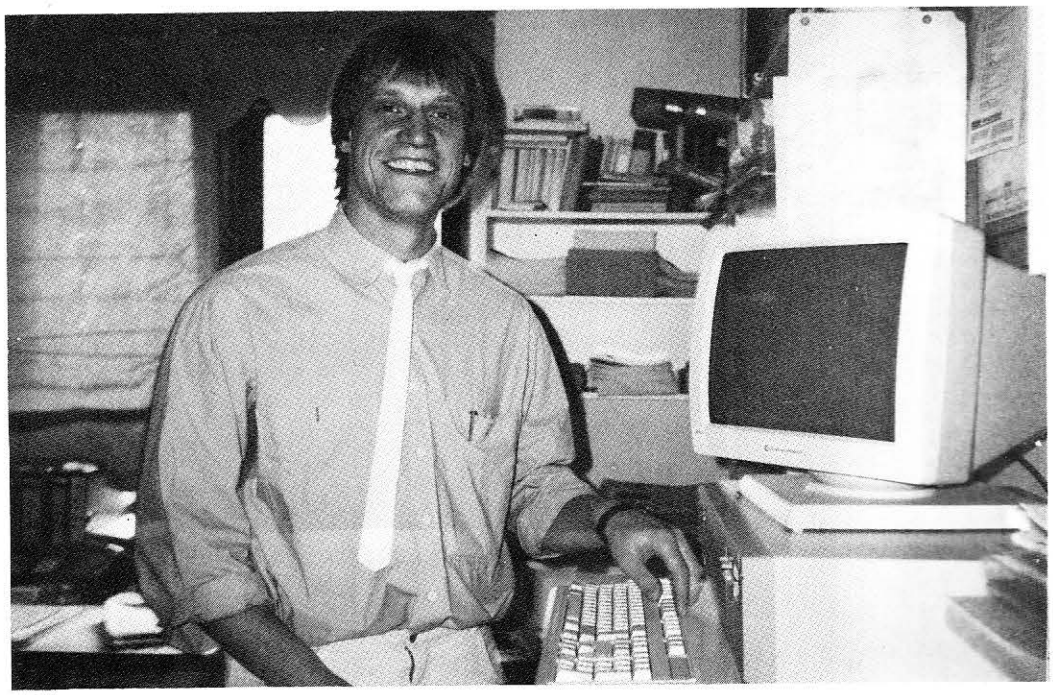
Hersteller gefunden, der das Programm innerhalb von fünf Monaten entwickelte. Mit ICBS (In-

W. Hoffmann von Commodore überreicht symbolisch eine Tastatur an den Leiter der Zwieseler Kurverwaltung

Coming Bureau System) liegt jetzt eine Software vor, die außer dem Kernstück der elektronischen Zimmervermittlung auch andere Aufgaben übernimmt: Verwaltung von Meldescheinen, Erstellung von Statistiken, Adressenverwaltung, Textverarbeitung und diverse verwaltungstechnische Berechnungen.

Blencke lobt die Zusammenarbeit mit den Münchnern: „Wir haben viel diskutiert, damit die Arbeit mit dem Programm einfach sein wird. Das Problem bei vielen Programmierern ist: Sie heben ab und machen eine Sache komplizierter, als sie sein müßte."

Das ist bei ICBS mit Sicherheit nicht der Fall. Auch Computer-Neulinge, und genau das sind die Angestellten der Kurverwaltung, dürften keine Schwierigkeiten damit haben.



Ein strahlendes Gesicht beweist, daß es funktioniert

»Die Programmierer heben ab«

Das Programm ist über die Tastatur oder mit der Maus zu bedienen. Mehrere Arbeitsschritte in verschiedenen Masken lassen sich gleichzeitig vornehmen. Auch der Datenschutz ist gewährleistet. Im besonders empfindlichen Statistik-Bereich erlaubt ein Mastercode nur dem Kurverwaltungsleiter Zugang.

ICBS ist zugeschnitten auf die Bedürfnisse von Fremdenverkehrsorten. Jedes Hotel oder Gasthaus, jede Pension oder Ferienwohnung wird mit den wichtigsten Informationen gespeichert.

Zur Vereinfachung werden die Betten zunächst als belegt geführt. Erst nach der Frei-Meldung des Wirts werden sie interessierten Gästen angeboten. Die einzelnen Betriebe sind in einer Warteschlange aufgelistet und werden der Reihe nach ausgegeben. Damit will man die Bevorzugung oder Benachteiligung einzelner Wirte verhindern. Außerdem, so Blencke: „Wenn sich jetzt einer beklagt, wir würden ihm keine Gäste vermitteln, können wir das jederzeit nachprüfen und das Gegenteil beweisen."

Ein Gast muß sich innerhalb von zwei Stunden bei seiner gewählten Unterkunft melden. Sie gilt während dieser Zeit als reserviert. Erst wenn die Kurverwaltung die Bestätigung des Wirts erhält, wird der Belegungsplan geändert.

Auch die schwarzen Schafe unter den „Herbergsvätern" lassen sich besser dingfest machen: Da der Meldeschein des Gastes schneller bearbeitet wird, ist so manches Schnäppchen nebenher, durch elegantes Umschiffen der Meldung, ausgeschlossen. Ungeschoren kommen diese vergeßlichen Zeitgenossen nicht davon: Das Programm verteilt bei derartigen Fehltritten schwarze Punkte. Wer drei davon beisammen hat, wird für einige Zeit bei der Vermittlung übergangen.

Gerade in diesem Bereich erwartet Blencke einige Schwierigkeiten: „Die Betriebe müssen unbedingt mitspielen". Es ist deshalb geplant, die Vermieter einzuladen und ihnen die Neuerungen vorzustellen. Zumal sie auch noch in anderer Form direkt davon betroffen sind: Durch einen eigenen Btx-Anschluß sollen sie in der Lage sein, den Zimmerbelegungsplan direkt zu ändern und eigenständig zu führen.

Keine Chance für schwarze Schafe

Große Chancen für die Post sieht in dieser Vernetzung Gerd Bruckner, zuständig für das Nexus-Marketing: „Privatpersonen so etwas anzubieten, lohnt sich wohl kaum". Ganz im Gegensatz zu Hotels und Gasthäusern, die damit auch ihre Bestellungen aufgeben könnten.

Bislang, obwohl nicht mehr allzu lange, ist das aber noch Zukunftsmusik. Im Zwieseler Verkehrsverein arbeitet man noch immer im „Handbetrieb" – bei etwa 370 000 Übernachtungen in Zwiesel pro Jahr ein immenser Zeitaufwand für die Angestellten.

Ähnlich sieht es auch im Marketing-Bereich aus. Zu erfahren, wodurch die Besucher auf Zwiesel aufmerksam gemacht wurden und woher sie gekommen sind, war bislang nur in Grenzen möglich: „Es ist kaum zu glauben, aber die Leute

sind mitunter rumgelaufen und haben sich Autokennzeichen notiert, um herauszufinden, von woher die Gäste angereist sind", sagt Bruckner.

Der Marketing-Nutzen ist auch ein Hintergedanke von Blencke: „Wir bekommen mit diesem System ein Kontrollinstrument in die Hand, mit dem wir feststellen können, wie viele Anfragen letztlich zu einer festen Buchung führen". Das heißt auch, falls Aufwendungen für Werbematerial und Anzahl der Besucher in keinem angemessenen Verhältnis stehen: „Was ist mit unseren Prospekten los?" Wenn eine Anzeigenkampagne, twa in überregionalen Medien, keinen sichtbaren Erfolg bringt, wird sie gestoppt.

Stiefkind Deutschland-Urlaub

Durch die EDV wird auch die engere Zusammenarbeit mit Reisebüros möglich. In dieser Hinsicht liegt einiges im argen. Laut Blencke, der aus der Touristik-Branche kommt, sind Deutschland-Urlaube noch immer Stiefkinder der Reise-Vermittler: „Es erfordert viel zu viel Aufwand und Kosten, bis eine Buchung zustande kommt".

In dieser Lücke liegt jedoch eine Chance für kleinere Fremdenverkehrsorte, sofern sie sich nicht gegen die Technik sträuben. Gerade mit den reisefreudigen „Neuen Alten" entstehe ein vielversprechender Markt: „Sehen Sie sich die Alterspyramide an: Weniger Junge, dafür umso mehr Alte. Das ist der Personenkreis, der Reisebüros in Anspruch nimmt", meint Gerd Bruckner. Ältere Menschen, so seine Ansicht, gingen auf Nummer sicher. „Die fahren nicht einfach ins Blaue, die wollen vorher wissen, wo sie übernachten werden. Denen ist auch wichtig, daß deutsch gesprochen wird".

Offensichtlich hat man in Zwiesel den richtigen Riecher gehabt. Das scheint auch nötig, da eine Zeitlang der Besucherstrom nachgelassen hatte. Jetzt verzeichnet man zwar wieder einen Aufwärtstrend, doch für größere Hotels ist die Stadt nicht attraktiv. Blencke: „Die Preise sind einfach zu schlecht".

So sind denn auch die etwa 350 Hotels, Gaststätten und Pensionen zum großen Teil reine Familienbetriebe, die rund ein- bis anderthalbtausend Personen (von rund 10.000 Einwohnern) beschäftigen.

Die Möglichkeit, den Verwaltungsaufwand zu reduzieren, ist nicht nur für die Zwieseler interessant. Auch andere Kurorte sind bereits hellhörig geworden. Als das Projekt Anfang März auf der ITB, der Internationalen Tourismus-Börse, in Berlin vorgestellt wurde, war, so Blencke, „der Stand permanent belegt."

Was leistet und was kostet ICBS?

Zimmervermittlung: In einer Datenbank werden die wichtigsten Daten der Beherbergungsbetriebe zusammengefaßt, vor allem Informationen über die momentane Belegung.

Meldescheine: Eingabe und Verwaltung von Meldescheinen.

Telekommunikation: Verschiedene Schnittstellen ermöglichen eine Anbindung an Btx und andere Übertragungsarten zum Anschluß an Reiseveranstalter, Busunternehmen und die Beherbergungsbetriebe.

Textverarbeitung: Ein vorhandenes Textverarbeitungs-Programm (etwa Word oder Wordstar) kann sofort integriert werden. Die spezielle ICBS-Textverarbeitung ist in Kürze verfügbar.

Statistik: Durch kontinuierliche Eingabe der Meldescheine sind Vergleiche mit den Daten der Zimmervermittlung möglich. Angaben der Beherbergungsbetriebe können kontrolliert werden. Die äußere Werbung (Inserate) läßt sich flexibel gestalten.

Berechnung: Verwaltungstechnische Berechnungen wie Fremdenverkehrs- oder Kurabgaben werden erfaßt und abgewickelt. Die Einbindung in das Statistik- und Textverarbeitungsprogramm erbringt Vergleichszahlen für Vergangenheit oder Zukunft.

Netzwerk: Durch Installation eines Netzwerks kann eine den jeweiligen Bedürfnissen angepaßte Anzahl von Rechnern angeschlossen werden.

Kosten:
Ein Arbeitsplatz:
Commodore PC-AT 40/40
Drucker NEC P7 (DIN A3)
Aufsatztraktor bidirektional
Software
Leasing: 458.04 Mark/Monat
Kauf: 20.000 Mark

Bis zu fünf Arbeitsplätze:
Fünf PC
Zwei Drucker
Kauf: 40.000 Mark

Netzanschluß:
(für bis zu acht Terminals)
10.000 Mark

Textverarbeitung:
etwa 300 bis 400 Mark

Btx-Anschluß:
65 Mark/Monat

Info-Boy meldet freie Zimmer

Die Vorbereitungen für die endgültige Inbetriebnahme sind voll im Gange. Bisher stehen der Kurverwaltung zwei AT 40/40 zur Verfügung. Laut Gerald Hahn sind etwa zehn Rechner vorgesehen. Bis Weihnachten, hofft Blencke, soll die Anlage einsatzbereit sein.

Ein sogenannter Info-Boy ist bereits installiert: An der Außenwand des Rathauses können Gäste per Knopfdruck freie Zimmer auf dem Bildschirm abrufen. Für den Leiter der Kurverwaltung nur ein zusätzlicher Service, denn wichtig sei die persönliche Betreuung der Gäste. Und die ist bisher zu kurz gekommen. Der Verwaltungsaufwand dürfte zwar nach Einführung der EDV um die Hälfte verringert werden. Weniger Arbeit bedeutet dies allerdings nicht: „Sie wird endlich dahin verlagert, wo sie auch hingehört: In die effektivere Betreuung und den Service für die Erholungssuchenden. *was*■

Neues bei der Post

Im vergangenen Jahr fegte ein Sturm durch den deutschen Blätterwald, als das Sachverständigen-Gutachten zur weiteren Entwicklung der Deutschen Bundespost vorgelegt wurde. Die Politiker stritten, die Postgewerkschaft protestierte und die Bevölkerung schwankte zwischen Zustimmung und leisen Zweifeln, ob eine Neustrukturierung für sie ein Vorteil sein würde. Was hat es wirklich auf sich mit der Umorganisation des bisher monopolistischen Postriesen?

In diesen Wochen soll ein Gesetzentwurf des Postministers zur Neugestaltung der Deutschen Bundespost im Kabinett und anschließend im Deutschen Bundestag beraten und verabschiedet werden. Die führenden Beamten im Postministerium geben nur zögernd nähere Details bekannt. Trotzdem haben wir Interessantes erfahren. Diese Informationen wollen wir Ihnen nicht vorenthalten.

Ausverkauf der Post: Wer will was?

Die Telefongebühren in Deutschland sind wesentlich überhöht. Aus den Überschüssen, die in diesem Bereich erwirtschaftet werden, wird die sogenannte Gelbe Post, also der Bereich des Brief-, Paket- und Päckchendienstes, subventioniert. Dieser innerbetriebliche Finanzausgleich soll innerhalb von einigen Jahren, man spricht von drei, abgebaut werden. So sehr sich jeder, der oft zum Telefonhörer greift, freuen wird, daß das Telefonieren billiger werden müßte – einen Wermutstropfen hat das Ganze schon.
Denn mit Sicherheit müssen dann das Briefporto und die Paket- und Päckchengebühren angehoben werden. Und zwar teilweise nicht unerheblich. Doch wen trifft das? Den Geschäftsmann sicher nicht so stark wie die Rentnerin. Denn der Unternehmer hat ja viele seiner Kontakte bisher schon über das Telefon abgewickelt und wird dies in Zukunft dann verstärkt tun. Und wie bisher wird er auch weiterhin die Gebühren als Geschäftskosten absetzen können.

Opfer Nummer 1: Die Rentnerin

Aber die alleinstehende Rentnerin, die sich auf Grund ihrer schmalen Rente kein Telefon leisten kann, wird wieder einmal zur Kasse gebeten, wenn sie den Kontakt mit ihren Angehörigen aufrecht erhalten und einen Brief frankieren will. Ob dies politisch wirklich so gewollt ist? Sicher, denn sonst würde es nicht so gemacht werden.
Oder wird von der Regierung in diesem unserem Land beabsichtigt, den Briefdienst noch mehr in die roten Zahlen rutschen zu lassen und auf Kosten der Steuerzahler, also auf Kosten aller Geldbeutel, über Wasser zu halten?

Dann ist abzusehen, daß dieser Dienstzweig bald so hoch verschuldet sein wird wie jetzt die Bahn. Oder die Leistungen werden noch mehr eingeschränkt, um Kosten zu sparen. Denn mit Rationalisierung ist beim Briefdienst kaum mehr etwas möglich, da die höchsten Kosten in diesem Bereich Personalkosten sind.

Opfer Nummer 2: Der Briefkunde

Man könnte natürlich die Briefkästen nur noch einmal täglich leeren, die Samstagszustellung aufgeben, die Paketzustellung in die Wohnung einstellen, den Zeitungsdienst sterben lassen und die Zahl der Briefkästen drastisch reduzieren. Doch mal ganz ehrlich. Will das jemand? Oder würde nicht dann die Schimpferei über die Post noch mehr zunehmen? Und mit der Tatsache, daß etwa 95 Prozent der bis 17 Uhr eingeworfenen Briefe am nächsten Tag beim Empfänger im Briefkasten liegen, wäre es wohl auch vorbei.

Mehr Auswahl bei den Telefonapparaten

Doch was kommt auf der anderen Seite auf den Telefonkunden zu? Auch dort ist noch so manches unausgegoren. Es soll zwar der sogenannte Endgerätemarkt freigegeben werden. In der Praxis läuft das darauf hinaus, daß jeder, der einen normalen Hauptanschluß besitzt (Postjargon: Tante-Emma-Anschluß), nicht mehr auf die Standardapparate der Post angewiesen ist, sondern einen Apparat seiner Wahl beim Händler kaufen kann, wenn dieser den technischen Normen der Post an der Schnittstelle entspricht.
Damit wäre es ohne weiteres möglich, sich ein schnurloses Telefon mit besonderem Komfort zuzulegen oder sich für ein besonderes Design zu entscheiden. Zweifellos wartet die einschlägige Fernmelde-Industrie darauf schon lange. Der Kunde sicher auch.
Doch wer sagt dann dem Kunden, was er wirklich anschließen darf und was nur für eine Haustelefonanlage geeignet ist? Denn über eine Tatsache müssen sich alle Beteiligten im klaren sein: Weltweit gibt es in diesem Bereich soviel No-Name-Ramsch, daß der Käufer sicher schnell die Übersicht verliert. Und wenn er dann schlecht beraten wird, ist es schon passiert.

Er hat zwar für sein gutes Geld einen teuren Apparat erstanden, darf ihn aber nicht anschließen, weil die Bedingungen zum Anschluß an das öffentliche Netz nach wie vor von der Post festgelegt werden. Und dies muß auch so sein, denn sonst könnte der Fall eintreten, daß ein Apparat, der technisch nicht dem hohen Standard der Post entspricht, nicht nur seinen eigenen Anschluß lahmlegt.

Trotzdem bedeutet die Vielfalt von zukünftigen Apparaten ohne Zweifel mehr Auswahl, da hat die Post trotz einiger guter Ansätze in der Vergangenheit wohl selig geschlafen. Doch was ist, wenn dieser Apparat eines Tages nicht mehr geht? Woher soll besagte Tante Emma wissen, wo der Fehler liegt? Ist ihr Apparat kaputt, liegt es an der Leitung auf der Straße, hat diese Leitung vielleicht ein Bagger angerissen oder liegt ein Fehler in der Vermittlungsstelle der Post vor?

Opfer Nummer 3: Der Telefonkunde

Bisher war dies kein Problem. Ging das Telefon nicht mehr, genügte ein Anruf bei der Entstörungsstelle, um, mehr oder weniger schnell, wieder telefonieren zu können. War der Apparat defekt, kam der Entstörer vorbei und wechselte ihn aus. War etwas anderes gestört, kümmerte sich die Post ebenfalls darum. Und das alles kostenlos!
Dies wird in Zukunft anders sein. Wer nicht den Normapparat der Post haben will, sondern sich einen im privaten Handel gekauft oder gemietet hat, der erhält in seiner Wohnung eine neue Anschlußdose als Schnittstelle installiert. Wenn eine Störung gemeldet wird, kann von der Entstörungsstelle der Post festgestellt werden, ob die Leitung bis zu diesem Übergabepunkt in Ordnung ist oder nicht.
Ist das nicht der Fall und liegt auch sonst kein Fehler in den weiterführenden technischen Einrichtungen der Post vor, so ist die Behebung des Fehlers allein Sache des Telefonbesitzers. Er muß sich darum kümmern, daß sein Apparat wieder funktioniert.

Zahlen müssen Sie immer!

Es bleibt nur die Wahl, entweder den Kundendienst der Firma anzurufen, bei der der Apparat gekauft wurde oder sich gleich einen neuen zuzulegen. Beides kostet Geld. Denn wer einen Defekt an der Waschmaschine hat, muß ja auch für die Reparatur bezahlen. So der Gedankengang bei den Verantwortlichen im Postministerium.
Doch was ist, wenn jemand einen Telefonapparat mit FTZ-Zulassung hat und dieser dann trotzdem stört: Kann die Post dafür auch Geld verlangen?

Opfer Nummer 4: Der Computer-Freak

Besonders kompliziert wird es, wenn jemand keinen normalen Apparat hat, sondern eine Mailbox betreibt oder am Btx-Verkehr teilnimmt, – oder auch nur einen Akustikkoppler angeschlossen hat. Welche Geräte sind nun von der Post, welche sind selbstgekauft, wo ist die Schnittstelle, wer ist wofür zuständig? Sicher werden auch besonders für die Computer-Fans einige neue Apparate auftauchen, so zum Beispiel bessere Akustik-Koppler oder Btx-Schnittstellen. Aber was ist, wenn diese einmal nicht in Ordnung sind? Halten die Richtlinien der Post mit dem technischen Fortschritt mit oder ist der User der Dumme?
Fragen über Fragen, die bisher ungeklärt sind und die auch in Zukunft noch Stoff für Streitereien bieten und wohl auch zu einigen Musterprozessen vor den zuständigen Gerichten führen werden. Wer will der erste sein?

Wer macht was, wo und wann?

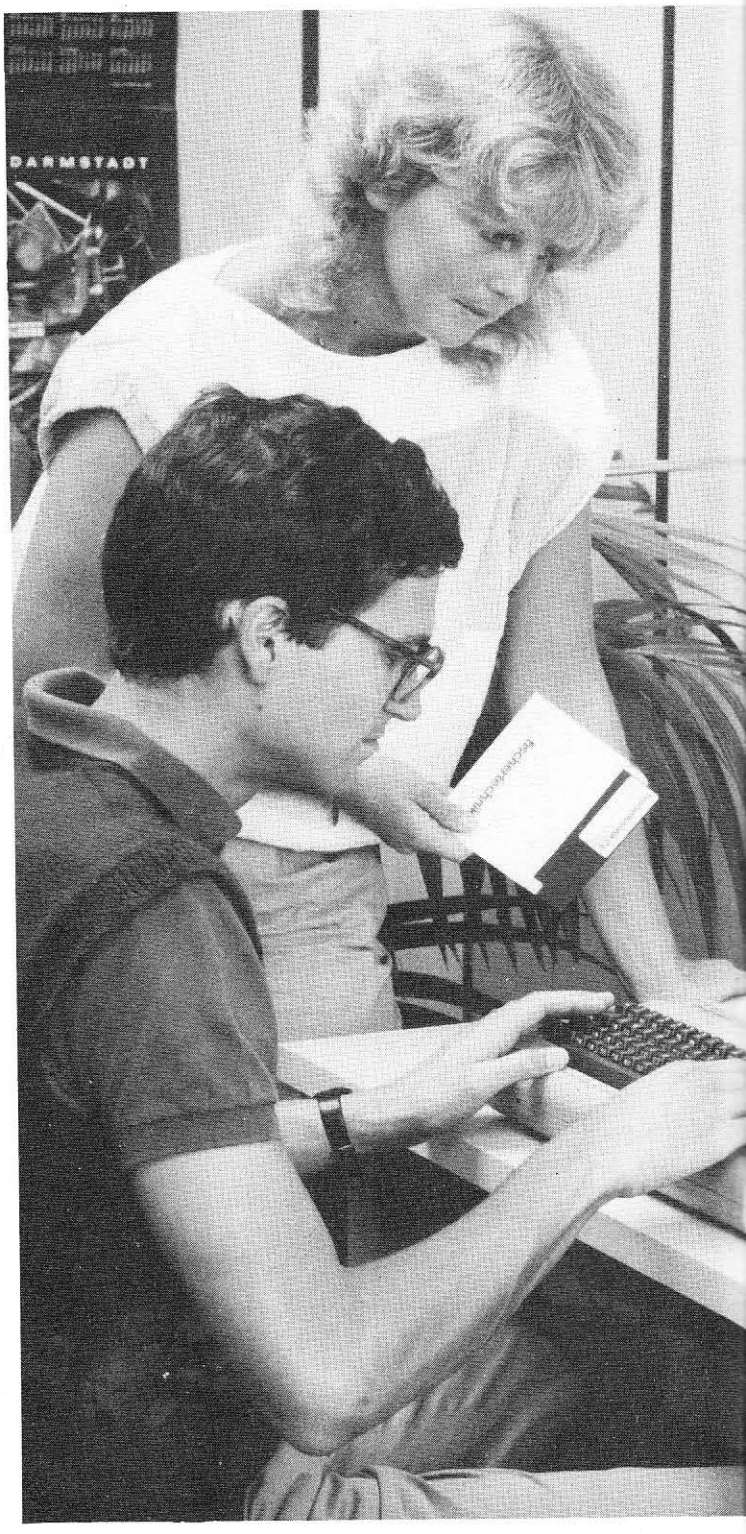
Denn was ist zum Beispiel, wenn bei einer Störung zwar die Leitung in Ordnung ist, der Telefonbesitzer aber trotzdem einen Entstörer verlangt? Muß der bezahlt werden, wenn er tatsächlich feststellt, daß der private Telefonapparat kaputt ist und stört, obwohl er eine Zulassungsnummer der Post hat? Darf er ihn trotzdem entstören oder

den Worten: „Da müssen Sie sich schon selbst drum kümmern."
Spätestens dann werden einige von denen, die jetzt für die Freigabe des Endgerätemarktes sind, aufwachen und sich sagen, daß sie das nicht gewußt haben und auch nicht informiert wurden, weder von der Post noch vom privaten Händler.
Darum merke: Es ist nicht alles Gold, was glänzt. Auch nicht, wenn es privates ist!

(AN)

Progran wie ei

Es geht neuen Zeiten entgegen; Zeiten, in denen es kein Herumirren mit dem Cursur und kein versehentliches Bildschirmlöschen mehr geben wird. Schluß mit Bedienfehlern beim INPUT. Windowtechnik, zwei SYS-Aufrufe und ein POKE-Befehl ermöglichen professionelle Eingabefelder mit komfortablen Editiermöglichkeiten. Folgen Sie aufmerksam dem Dialog zwischen Holmes und Watson, und auch Sie werden schon bald wie ein Profi programmieren.



Wer kennt sie nicht, Dr. Watson und Sherlock Holmes, das Duo, das jeden Kriminalfall löst. Der Eifer und die Dienstbeflissenheit eines Watson brauchen das Genie und den Scharfsinn eines Holmes. Nur so entstehen aufsehenerregende Ergebnisse. Lauschen wir doch einmal einem Gespräch dieser zwei Helden, das sich in unserer heutigen Zeit um folgendes hätte drehen können.
Nach einem ausgiebigen englischen Frühstück zündet Holmes sich seine Pfeife an. Sinnend bläst er blaue Rauchwölkchen in die Luft, während sich Watson mit der Tageszeitung beschäftigt. Die Besprechung bei Scotland Yard ist erst am späten Nachmittag, und so bleibt noch etwas Zeit zur Muße. Auf einmal fragt Holmes: „Was gibt es Neues, Watson?“
Watson blickt von seinem Artikel über Computer und gesellschaftliche Tendenzen auf, und es entwickelt sich ein Gespräch, in dessen Verlauf Watson allerlei Aktivitäten entwickelt, Holmes aber lediglich weiter genüßlich an seiner Pfeife zieht und ganz nebenbei, als wäre dies gar nichts, unbezahlbare Entdeckungen macht.

Watson: Ich lese hier gerade, bei vielen Computerneulingen bestehe die Tendenz, das Programmieren nach ein paar anfänglichen Versuchen bald wieder aufzugeben, da mit ein paar BASIC-Kenntnissen keine Professionalität zu erzielen sei und die Ergebnisse daher unbefriedigend blieben.
Holmes: Glauben Sie, daß das den Tatsachen entspricht?
Watson: Sie meinen, die Tendenz sei eine andere, und wir sollten eigene Untersuchungen anstellen?

INPUT MIT MANKOS

Holmes: Nein, lieber Watson, was wäre dadurch schon gewonnen. In der Tat lassen viele Programme, die sich in Computerzeitschriften finden, einiges zu wünschen übrig. Wir brauchen uns nur einmal den INPUT-Befehl anzusehen, der meist zum Einlesen von Eingaben verwendet wird. Der Benutzer kann kreuz und quer über den Bildschirm fahren, die Eingabemaske überschreiben, sie nach oben wegscrollen oder den ganzen Bildschirm löschen. Komma oder Doppelpunkt fungieren gar als Trennzeichen für mehrere Eingabefelder und lassen sich daher nicht erfassen. Was ist eine Textverarbeitung wert, die nicht einmal einen Doppelpunkt oder ein Komma annimmt? Ich will daher nicht einmal den ersten Teil der Behauptung in Frage stellen.
Watson: Wieso ersten Teil, gibt es denn einen zweiten? Ah, ich sehe schon, Sie spielen darauf an, ob nicht auch mit ein paar BASIC-Kenntnissen professionell programmiert werden könne. Aber, Sir, ein komfortables Eingabefeld mit dem GET-Befehl zu schaffen, erfordert eine

nmieren
n Profi

Menge an Programmieraufwand. Außerdem dürfte die Sache dann wohl ziemlich langsam werden, so daß die Bildschirmausgabe der Eingabe hinterherhinkt. Was bleibt also übrig? Doch nur die Maschinensprache, die aber nur etwas für Profis ist. Für unseren Neuling oder Durchschnittsprogrammierer ist also keine Lösung in Sicht.
Holmes: Wer vorschnell urteilt, beraubt sich selbst der Möglichkeit, die Problemlösung zu finden.
Watson: Sie glauben also, diese Möglichkeit würde tatsächlich bestehen?
Holmes: Lieber Watson, ich glaube gar nichts. Aber ich denke, die Sache wäre der Untersuchung wert. Außerdem bieten C16, C116 sowie der Plus/4 mit ihrem komfortablen BASIC und den zusätzlichen Editor-Funktionen sehr wohl die Möglichkeit, ein örtlich begrenztes Eingabefeld zu entwickeln, das ein Herauswandern des Cursors unterbindet.

WINDOW BEGRENZT EINGABEFELD

Watson: Ah, an die Windowtechnik hätte ich jetzt nicht im Traume gedacht. Das wollen wir gleich ausprobieren. Der Computer ist eingeschaltet; zwei Doppelpunkte, die Anfang und Ende des Eingabefeldes markieren sollen: ESC T nach dem ersten Doppelpunkt, um den Beginn des Fensters zu setzen; ESC B vor dem zweiten Doppelpunkt, um das Ende zu setzen. Ich teste jetzt alle Möglichkeiten durch. Cursor nach unten bringt ein unerwünschtes Resultat: der eingegebene Text verschwindet. Das gleiche geschieht auch am Feldende bei Cursor nach rechts oder bei der Eingabe eines Buchstabens. Prima funktioniert der Einfügemodus mit ESC A. Schon hat ein Buchstabe den rechten Feldrand erreicht.
Was wohl beim nächsten Verschieben geschieht? Der Cursor ist weg, nichts rührt sich mehr. Oh je, ein regelrechter Systemabsturz. Die Idee war gut, die Resultate jedoch weniger.
Holmes: Warum wohl sind die Resultate unbefriedigend, mein lieber Watson? Ganz einfach, weil nach Erreichen des rechten Feldrandes oder bei der Verschiebung des Cursors der Bildschirm, oder was von ihm noch im Fenster erscheint, scrollt. Sie brauchen daher nur den Scroll-Modus auszuschalten.
Watson: Also alles noch einmal von vorne: Reset ist durchgeführt, Fenster wurde neu definiert, mit ESC M das Scrolling ausgeschaltet. Der Cursor läßt sich nicht mehr nach unten bewegen. Will ich mit ihm nach rechts das Eingabefeld verlassen, erscheint er am Feldanfang. Der eingegebene Text

SCROLLING UNTERDRÜCKEN

bleibt jedoch erhalten. Im Einfüge-Modus bleiben die Buchstaben bei Erreichen des rechten Randes stehen und können, da es nichts mehr zu verschieben gibt, durch entsprechenden Tastendruck überschrieben werden. Am rechten Feldrand angelangt, bewegt sich der Cursor nach Eingabe eines beliebigen Buchstabens nicht mehr weiter, sondern bleibt auf diesem stehen. Sehr gut, einfach wunderbar. Wir können die Sache wohl so lassen.
Holmes: Etwas haben wir allerdings noch nicht berücksichtigt, lieber Watson.
Watson: Richtig, Sir, es gibt noch unerwünschte Tastendrucke, die unser Eingabefeld wieder zunichte machen können: etwa zweimaliges Drücken der HOME-Taste oder

das Wiedereinschalten des Scroll-Modus mit ESC L.
Holmes: Nicht zu vergessen die ESC-Sequenzen ESC R, ESC T, ESC B oder ESC N.
Watson: Das bedeutet aber, daß nicht mit INPUT eingelesen werden darf, sondern daß wir eine aufwendige GET-Routine schreiben müssen.
Holmes: Das ist nicht gesagt, Watson. Zwar ist der GET-Befehl zu benützen, jedoch sollte sich eine aufwendige Programmkonstruktion vermeiden lassen. Wir verwenden den GET-Befehl nicht zur vollständigen Erfassung der gesamten Eingaben. Wir brauchen ihn nur zur Erfassung der jeweiligen Taste, die wir durch einen PRINT-Befehl auf den Bildschirm zu bringen gedenken.
Watson: Wie aber sollen wir dann den auf dem Bildschirm sichtbaren Text in eine Variable bringen?

GET PRINT UND INPUT

Holmes: Natürlich mit einem INPUT-Befehl.
Watson: Ich verstehe nicht: Wir sollen also den INPUT-Befehl vermeiden und statt dessen GET benützen. Anschließend aber doch wieder INPUT?
Holmes: Ganz richtig, Watson, genau in dieser Reihenfolge. Was wissen Sie über den INPUT-Befehl?
Watson: Der INPUT-Befehl ermöglicht die Eingabe von Zahlen oder Strings. Folgt dem INPUT eine ganze Eingabeliste, so können mit nur einem Befehl gleich mehrere Eingaben erfaßt werden.
Holmes: Ist das schon alles? Nehmen wir an, durch einen INPUT-Befehl werde die Eingabe eines einzigen Textstrings erwartet. Sie geben diesen ein. Die Stringvariable sei A$. Was ist mit dem Inhalt der Variablen nach Ihrer Eingabe?
Watson: Die Variable enthält meine Eingabe.
Holmes: Und wenn in der Eingabezeile bereits etwas anderes gestanden hat?
Watson: Aha, daher weht der Wind. Bisher war ich der Auffassung, nur die Eingabe würde erfaßt werden. Da ich aber Ihren Scharfsinn kenne, bin ich nicht mehr sicher. Sie glauben, der gesamte Inhalt der Zeile würde erfaßt?
Holmes: Zumindest das, was sich rechts des INPUT-Fragezeichens befindet. Machen wir einen kleinen Versuch.
Geben Sie ein:

```
1 PRINT CHR$(147)
  "                REST"
2 PRINT CHR$(19);
  :INPUT A$
3 PRINT A$
```

Watson: Sie haben recht, Sir. Das heißt also, wir können mit dem INPUT-Befehl bereits auf dem Bildschirm Vorhandenes einlesen, folglich auch unsere durch eine GET-Schleife gemachte Eingabe. Ärgerlich wären aber zwei einzugebende RETURNs: das erste in der GET-Schleife, um das Ende der Eingabe zu signalisieren, das zweite, um den INPUT abzuschließen. Doch halt, warum das zweite RETURN nicht einfach vorgeben durch POKEn in den Tastaturpuffer? Wirklich, Sir, einfach genial. Wenn nur das störende Fragezeichen und die Sache mit dem Komma und dem Doppelpunkt nicht wären.

MODIFZIERTER INPUT

Holmes: Watson, Sie machen mich auf ein Versäumnis meinerseits aufmerksam. Ich hätte Ihnen ans Herz legen sollen, COMMODORE WELT zu lesen, deren Specials neuerdings in zweimonatigen Abständen erscheinen. Schlagen Sie doch in CW 12/87 auf Seite 74 den Artikel „Computer-Fieber" auf und führen Sie sich diesen zu Gemüte.
Watson: Wirklich interessant, die Geschichte. Mit SYS 34906 bekommen wir unsere Eingabe in den Eingabepuffer und von dort durch eine PEEK-Routine in unsere Variable.
Holmes: Nur nicht zu eilig, lieber Watson. Vielleicht lassen sich die PEEKs vermeiden. Ich denke, wir sollten uns die INPUT-Routine, und zwar den Teil, der die Variablenzuweisung vornimmt, im ROM-Listing einmal näher ansehen.
Watson: Hier ab Adresse $9156 ist das, was wir suchen. Ein wenig befremdlich erscheinen mir allerdings die Abfragen, ob es sich um INPUT, GET oder READ handelt. Sollte das etwa heißen, daß alle drei Befehle ein und dieselbe Zuweisungsroutine benützen?
Holmes: Dies ist sehr gut möglich. Warum sich eine Arbeit dreimal machen, haben sich die Systemprogrammierer bestimmt gefragt. Anstatt drei Routinen, die sich nur geringfügig voneinander unterscheiden, gibt es eine einzige, die die Zuweisung für READ, GET und INPUT vornimmt.
Watson: Es bestehen aber doch gravierende Unterschiede: Der INPUT-Befehl liest den Bildschirminhalt in den Eingabepuffer und kann bis zu 88 Zeichen erfassen. Der GET-Befehl liest aber nur ein einziges Zeichen aus dem Tastaturpuffer.
Holmes: Letzteres ist nur ein Teil der ganzen Wahrheit. Watson, sehen Sie sich doch bitte auch den ersten Teil der GET-Routine an.
Watson: Stimmt, Sir, das Zeichen aus dem Tastaturpuffer wird in den Eingabepuffer geschrieben. Durch eine daran anknüpfende Null wird der nachfolgenden Zuweisungsroutine signalisiert, daß nur ein einziges Zeichen im Eingabepuffer zu finden ist.
Homes: So haben wir genau das, was wir brauchen. Der Zuweisungsteil der GET-Routine könnte also ebenso wie der der INPUT-Routine bis zu 88 Zeichen erfassen; mit dem Unterschied, daß der GET-Befehl sich durch Trennzeichen wie Kommas oder Doppelpunkte nicht irritieren läßt.
Durch Verwendung des Eingabeteils der INPUT-Routine mit SYS 34906 und des Zuweisungsteiles der GET-Routine mit POKE 17,64, gefolgt von SYS 37210, haben wir uns eine optimale Eingabe geschaffen. Der POKE-Befehl weist dem Eingabeflag in Adresse 17 die Zahl 64 zu, die der Zuweisungsroutine SYS 37210 ein GET signalisiert. Das INPUT-Flag dagegen hätte den Wert null und das READ-Flag den Wert 152. Dies soll uns aber zum jetzigen Zeitpunkt nicht weiter interessieren, obwohl sicher interessante Anwendungen hieraus erfolgen könnten.

EINGABEFELDER, EINGABEMASKEN, TEXTVERARBEITUNGEN

Watson: Die erste Grundlage eines guten Programmes ist ein gutes Eingabefeld. Erst dadurch werden Dinge möglich, von denen ein C16/116/Plus/4-User bislang oft nur träumen konnte.
Holmes: Ja, die logische Folge eines Eingabefeldes sind mehrere Eingabefelder, die nicht vereinzelt hintereinander abgearbeitet werden, sondern zu einer Eingabemaske verbunden sind. Falsche oder unvollständige Eingaben in einem vorangegangenen Feld brauchen nicht gesondert nach Abschluß der Eingaben korrigiert werden.
In einer Eingabemaske kann mit dem Cursor nach Belieben von Feld zu Feld gesprungen werden. Es gibt kein Vorher und Nachher mehr. Alles ist eines.

Einfach mit dem Cursor hinfahren und ändern. Die Datenbanken, Adreßdateien, Textverarbeitungen oder sonstigen Anwendungen dieser Technik können sich sehen lassen. Doch die weitergehende Arbeit wollen wir anderen Händen überlassen. Wir sollten statt dessen zu Scotland Yard aufbrechen.
So etwa hätte ein Gespräch zwischen Watson und Holmes verlaufen können, hätten diese heute gelebt und sich für unsere Computer interessiert. Die Ergebnisse dieser Unterhaltung wollen wir in mehreren Programmen festhalten. Eines davon heißt Eingabefeld, ein anderes Eingabemaske. Als vorläufige Krönung zeigen wir Ihnen, wie eine bildschirmorientierte Textverarbeitung realisiert wird. Mehr darüber erfahren Sie in den nachfolgenden Artikeln.

a.m.□

Eingabefeld mit Pfiff

Eindeutig lokalisierte Eingabefelder mit komfortablen Editiermöglichkeiten und Akzeptierung von Kommas und Doppelpunkten können auch Sie leicht in Ihre eigenen Programme einbauen.

Mit unserem Programm „Eingabefeld" steht Ihnen eine Routine zur Verfügung, die auch Ihnen erlaubt, Anwendungen zu erstellen, die in punkto Bedienerfreundlichkeit den Vergleich mit professionellen Programmen nicht zu scheuen brauchen.

EIGENSCHAFTEN

Diese Routine ist gedacht zum Einbau in eigene Programme. Besondere Vorteile gegenüber dem INPUT-Befehl sind:

- Die Eingabe ist auf ein eindeutig lokalisiertes Eingabefeld beschränkt. Effekte auf den Rest-Bildschirm sind ausgeschlossen. Es gibt also kein versehentliches Überschreiben oder Löschen des Bildschirminhaltes mehr. Auch ein Herumirren mit dem Cursor ist nicht mehr möglich.
- Die komfortablen Escape-Sequenzen des Editors können benutzt werden. Unerwünschte Sequenzen, die das Eingabefeld zerstören würden, wurden eliminiert. Bei einigen Editor-Routinen konnte gar eine Verbesserung erzielt werden.
- Anders als beim normalen INPUT-Befehl gibt es mit Kommas und Doppelpunkten keine Schwierigkeiten. Diese Satzzeichen können ohne weiteres verwendet werden.

HANDHABUNG

Wollen Sie unsere Eingaberoutine in Ihrem eigenen Programm nutzen, so brauchen Sie diesem nur die Zeilen 100 bis 480 voranzustellen. Die REM-Zeilen können Sie weglassen. Ihr eigenes Programm beginnt dann ab Zeile 520. Zur Lokalisierung des Eingabefeldes sind die Koordinaten und die Feldlänge an die Variablen X, Y und Z zu übergeben.

Parameterübergabe:

X = Spalte des Feldanfangs
Y = Zeile des Feldanfangs
Z = Feldlänge
F$ = Stringvariable
FA$ = Markierung des Feldanfangs
FE$ = Markierung des Feldendes

Mit GOSUB 170 rufen Sie die Eingaberoutine auf. Wünschen Sie ein leeres Eingabefeld, so weisen Sie vorher der Stringvariablen F$ den Leerstring zu mit F$=" ". Ansonsten erscheint der Inhalt von F$ als Vorgabe im Eingabefeld. Wer sich nicht mit einem Doppelpunkt als Markierungen von Feldanfang und Feldende zufriedengeben will, kann den Variablen FA$ und FE$ nach Belieben andere Zeichen zuweisen. Nach erfolgter Eingabe stehen die Eingabedaten in Form der Variablen F$ zur weiteren Verwendung bereit.

EDITIEREN

Neben den CURSOR-Tasten, der INST-, DEL-, HOME- und CLEAR-Taste, stehen Ihnen folgende ESCAPE-Sequenzen zur Verfügung:

ESC-A: Einschalten des Einfügemodus.
Dieser Modus ist normalerweise aktiv, sofern er nicht mit ESC-T ausgeschaltet wurde.

ESC-C: Einschalten des Überschreibemodus.

ESC-O: Flags ausschalten.
Das Quote-Flag braucht nicht ausgeschaltet zu werden, da es automatisch desaktiviert wird.

ESC-J: Cursor an den Feldanfang.
Entspricht dem Druck der HOME-Taste.

ESC-K: Cursor an das Ende der Eingabe.
Im Unterschied zur normalen Editierfunktion sitzt der Cursor nicht auf dem letzten Buchstaben, sondern springt um eine Position weiter nach rechts, sofern der letzte Buchstabe nicht am Feldende liegt.

ESC-P: Löschen bis zur Cursorposition.
Im Unterschied zur normalen Editierfunktion wird der Buchstabe unter dem Cursor nicht mehr gelöscht. Außerdem wandert der nicht gelöschte Rest an den Feldanfang.

ESC-Q: Löschen ab Cursorfunktion.

Da die GET-Routine keinen sichtbaren Cursor hervorbringt, wird dieser mit PEEK und POKE erzeugt und blinkt im Unterschied zum gewohnten Cursor nicht. Zwei Gründe sprachen für einen nichtblinkenden Cursor: Zum einen ist er mit geringerem Aufwand zu realisieren, zum anderen empfinden einige unserer Leser das Blinken als störend.

PROGRAMM-BESCHREIBUNG

Zeilen 100 bis 130
Hier findet eine Initialisierung folgender Variablen statt:

EC$ = Escape-Code chr$(27)
V$ = Cursor an Feldanfang
MO$= Editiermodus. Es stehen zur Wahl: der Einfüge- und der Überschreibmodus. MO$ wird durch Einschalten des betreffenden Modus bei der Eingabe ständig aktualisiert.
IN$ = Tasten mit Sonderbehandlung. RETURN, HOME und ESCAPE werden nicht wie die anderen Tasten einfach auf den Bildschirm ausgegeben, sondern gesondert abgehandelt.
ES$ = Zugelassene Escape-Sequenzen.
L$ = Stringvariable, die bei der Fensterdefinition Verwendung findet. Sie enthält zweimal Cursor nach links, um auf die letzte Feldposition zu gelangen, Setzen des Feld-

endes und Feld löschen.

Zeile 510
Sprung ins Hauptprogramm.

Zeilen 170 und 180
Fensterdefinition. Der CHAR-Befehl positioniert den Cursor und setzt die Feldanfangs-Markierung. Der PRINT-Befehl setzt die linke Grenze des Eingabefeldes, positioniert mittels SPC-Befehl den Cursor auf den Platz der Markierung für den rechten Rand, schaltet das Scrolling aus und setzt die Feldende-Markierung. Nach der Variablen L$, die die Fensterdefinition beendet, wird der in F$ abgelegte Text ausgegeben, der Cursor auf den Feldanfang gesetzt und die Einschaltung des aktuellen Editiermodus vorgenommen.

Zeilen 220 und 230
Zeicheneingabe. Aus der aktuellen Cursorzeile, der Cursorspalte und dem Anfang des Textbildschirmes wird die Speicheradresse AX der augenblicklichen Cursorposition errechnet. Der erste POKE-Befehl setzt das REVERS-Bit und macht so den Cursor sichtbar. Nach dem Einlesen der gedrückten Taste mittels GETKEY wird der Cursor durch Zurücksetzen des REVERS-Bit wieder unsichtbar gemacht.

Zeilen 270 bis 300
Zeichenauswertung. Die INSTR-Anweisung stellt fest, ob das eingelesene Zeichen normal abgearbeitet werden soll, oder eine Sonderbehandlung verdient. In Zeile 280 findet gegebenenfalls die entsprechende Verzweigung statt. Im Regelfall wird die Zeile 290 abgearbeitet, die das Zeichen auf den Bildschirm ausgibt, den Anführungszeichenmodus aufhebt und den Rücksprung zur nächsten Zeichenausgabe veranlaßt.

Zeile 300 bearbeitet die HOME-Taste. Da zweimaliges Drücken der HOME-Taste die Fensterdefinition normalerweise aufheben würde, wird HOME durch die Escape-Sequenz ESC-J ersetzt. Dies geschieht hier durch die Zuweisung X$="J" und den Sprung in die ESC-SEQUENZ-Bearbeitung.

Zeilen 340 bis 370
Zeichenübernahme. Wird die RETURN-Taste betätigt, sollen die Eingabe abgeschlossen und die Daten an die Variable F$ übermittelt werden. Dieses geschieht durch das Setzen des Cursors an den Feldanfang in Zeile 340, die RETURN-Vorgabe durch POKEn in den Tastaturpuffer in Zeile 350, einen modifizierten INPUT in Zeile 360 und das Aufheben der Fensterdefinition in Zeile 370. Der erste SYS-Befehl in Zeile 360 bewirkt die Übernahme der eingegebenen Zeichen in den Eingabepuffer.
Der POKE-Befehl setzt das GET-Flag. Damit kann der zweite SYS-Befehl, der für die Übergabe der im Eingabepuffer abgelegten Zeichen an die Variable F$ verantwortlich ist, auch Kommas und Doppelpunkte wie normalen Text behandeln. Die CHAR-Anweisung in Zeile 370 hebt die Fensterdefinition auf, die PRINT-Anweisung schaltet das Scrolling und den Überschreib-Modus wieder ein. Dem Rücksprung in das aufrufende Programm steht somit nichts mehr im Wege.

Zeilen 410 bis 470
Bearbeitung von ESCAPE-Sequenzen. Mit GETKEY wird der dem ESCAPE-Code folgende Tastendruck eingelesen. Sollte dieser in der Variablen ES$ vorgesehen sein, so findet in Zeile 420 die entsprechende Verzweigung statt.
Anderenfalls erfolgt der Rücksprung zur Zeichen-

EINGABEFELD MIT PFIFF

```
10 rem eingabefeld============c16 <eg>
20 rem (p) commodore welt team   <ho>
30 rem ========================== <ng>
40 rem (c) by alfons mittelmeyer <cg>
50 rem                           <pd>
60 rem                           <ah>
70 rem basic v3.5                <nl>
80 rem c16/116/plus4             <ki>
90 rem ========================== <jg>
100 ec$=chr$(27):v$=ec$+"j":mo$=ec
$+"a"                            <ki>
110 in$=chr$(13)+ec$+chr$(19):es$=
"acjopqk"                        <ih>
120 l$=chr$(157)+chr$(157)+ec$+"b"
+chr$(147)                       <ch>
130 fa$=":":fe$=":":goto520      <ed>
140 rem ------------------------ <fi>
150 rem  fensterdefinition       <cc>
160 rem ------------------------ <mk>
170 char,x,y,fa$                 <jn>
180 printec$"t"spc(z)ec$"m"fe$l$f$
v$mo$;                           <jg>
190 rem ------------------------ <ca>
200 rem  zeicheneingabe          <al>
210 rem ------------------------ <da>
220 ax=3072+40*peek(205)+peek(202) <fk>
230 pokeax,peek(ax)or128:getkeyx$:
pokeax,peek(ax)and127            <id>
240 rem ------------------------ <ch>
250 rem  zeichenauswertung       <ga>
260 rem ------------------------ <ob>
270 in=instr(in$,x$)             <ap>
280 oningoto340,410,300          <fp>
290 printx$;:poke203,0:goto220   <ko>
300 x$="j":goto460               <pk>
310 rem ------------------------ <me>
320 rem  zeichenuebernahme       <la>
330 rem ------------------------ <dk>
340 printv$                      <nn>
350 poke239,1:poke1319,13        <pg>
360 sys34906:poke17,64:sys37210f$ <ke>
370 char,0,0,"":printec$"l"ec$"c";
:return                          <on>
380 rem ------------------------ <kn>
390 rem  esc-sequenz             <co>
400 rem ------------------------ <fc>
410 getkeyx$:in=instr(es$,x$)    <he>
420 oningoto450,450,460,460,440,46
0,470                            <id>
430 goto220                      <ka>
440 gosub350:goto170             <cp>
450 mo$=ec$+x$                   <kc>
460 printec$x$;:goto220          <lk>
470 printec$x$;:ifpeek(2024)<>peek
(202)thenprintchr$(29);          <ik>
480 goto220                      <lp>
```

```
490 rem ========================== <pm>
500 rem  hauptprogramm             <fp>
510 rem ========================== <af>
520 scnclr                         <pm>
530 x=10:y=5:z=20:gosub170         <al>
540 printf$:end                    <ba>
550 rem ========================== <pp>
560 rem  p r o g r a m m e n d e   <hp>
570 rem ========================== <mo>
```

eingabe. Zeile 440 bearbeitet das Löschen bis zur Cursorposition. Mit GOSUB 350 wird der Text ab Cursorposition an die Variable F$ übergeben.
Der GOTO-Befehl ruft die Eingaberoutine neu auf. Die durch F$ vorgegebenen Daten erscheinen nun am Fensteranfang. Zeile 450 wird bei der Modus-Umschaltung angesprungen und bewirkt die Aktualisierung des Editiermodus. Die normale Bearbeitung einer ESCAPE-Sequenz findet sich in Zeile 460. Der ESCAPE-Code und nachfolgender Code werden ausgegeben, worauf der Rücksprung zur Zeicheneingabe folgt.
Einen Sonderfall bildet ESC-K. Nach der Ausgabe der ESCAPE-Sequenz in Zeile 470 fragen wir ab, ob die aktuelle Cursorspalte mit dem rechten Fensterrand zusammenfällt. Bei Nichtzutreffen geben wir noch ein Cursor-nach-rechts aus, so daß unser Cursor hinter das letzte Zeichen zu liegen kommt und optimal für weiteres Eingeben bereit ist.

Zeilen 520 bis 540
Hauptprogramm. Ab Zeile 520 beginnt Ihr eigenes Programm.
Wir haben drei Programmzeilen als Beispiel für den Aufruf der Eingaberoutine angefügt. Hier wird ein Eingabefeld mit der Länge von 20 Buchstaben ab Spalte 10 und Zeile 5 definiert. Die eingegebenen Daten erscheinen anschließend links oben am Bildschirm, da das Schließen des Fensters den Cursor an die linke obere Bildschirmecke setzt.
In Ihren Programmen werden Sie diese Eingaberoutine bestimmt nicht mehr missen mögen.

a.m. □

Die Maske macht Musik

Von unserem Eingabefeld ist es zu einer Maskeneingabe, bei der nach Belieben von Feld zu Feld gesprungen werden kann, nur mehr ein kleiner Schritt. Mit drei GOSUBs und einigen DATA-Zeilen realisieren Sie bereits Ihre individuelle Eingabemaske.

Bei vielen Anwendungen, besonders bei Dateien, sind mehrere Eingaben erforderlich, wie Name, Vorname, Straße, Wohnort und weiteres. Wer ist dabei schon gegen Tippfehler gefeit. Besonders ärgerlich ist es, wenn der Fehler nicht mehr rechtzeitig entdeckt wird.
Zwar ist die falsche Eingabe noch auf dem Bildschirm zu sehen, doch der Cursor befindet sich bereits im nächsten Eingabefeld. Es gibt daher kein Zurück mehr. Entweder kann eine Korrektur nach erfolgter Erfassung aller Datenfelder vorgenommen werden, oder es ist gar ein Wechsel des Menüs erforderlich: Eingabemenü verlassen, Änderungsmenü anwählen, Datensatz suchen und ändern. Darauf wieder Änderungsmenü verlassen, Eingabemenü wählen und weitererfassen. Eine ziemlich umständliche Geschichte.

EINGABEMASKE

Von professionellen Anwendungen sind wir anderes gewohnt. Tritt hier der nämliche Fall auf, so brauchen wir nur eine Taste zu drücken, und schon befindet sich der Cursor wieder im vorigen Eingabefeld. Wir können von Feld zu Feld springen, wie es uns beliebt. Das ist sicher der Idealfall. Warum sollten wir also nicht auch professionelle Eingabemasken programmieren? Weil es vielleicht zu schwierig wäre? Mit unserem Eingabefeld als Grundlage gehört gar nicht mehr allzu viel dazu.

STEUERUNG DER MASKENEINGABE

Außer den bereits vorgestellten Steuertasten beim Eingabefeld benötigen wir noch zwei weitere: eine zum Springen in das vorherige Feld und eine zum Abschließen der Maskeneingabe. Es kann sein, daß Einträge in weitere Felder sich erübrigen, obwohl wir uns noch nicht im letzten Eingabefeld befinden. Aus Symmetriegründen haben wir für den Sprung in das nächste Eingabefeld, wofür bereits die Return-Taste bereitsteht, noch eine Cursor-Taste verwendet.

Cursor oben
= vorheriges Feld
Cursor unten
= nächstes Feld
Return
= nächstes Feld
Control C
= Abschluß

Control C bewirkt dasselbe wie Return oder Cursor nach unten im letzten Feld.
Im Programm haben wir in Zeile 520 die Variable IN$ um die genannten Tasten-Codes erweitert. Eine entsprechende Ergänzung mußte daher in der Zeile 280 stattfinden, wo nun, ebenso wie bei Return, eine Verzweigung auf den Feldabschluß auch bei den dazugekommenen Steuertasten stattfindet. Die Variable MA$ in Zeile 520 enthält die Codes für die neu hinzugekommenen Feldsteuertasten. Am Ende der Zeile erfolgt der Sprung auf das neue Hauptprogramm. Damit sind die Vorbereitungen abgeschlossen, die Maskenroutine kann folgen.

MASKENROUTINE

Die Maskeneingabe besteht aus zwei Teilen, der Ausgabe und der Eingabe. Bevor mit der Eingabe begonnen wird, sollen alle Felder auf dem Bildschirm sichtbar sein. Besonders im Änderungsmodus ist die Übersicht über den Inhalt aller Felder wünschenswert.
Auch bei noch leeren Feldern im Eingabemodus erhöht die Information über die existierenden Felder und deren Lage die Übersicht. In Zeile 530 werden deshalb durch eine Laufschleife von Null bis M und Sprung in die Ausgaberoutine des jeweiligen Feldes alle Felder ausgegeben. M ist die Feldanzahl minus eins.
Die Zeilen 620 und 630 sind für die Ausgabe des jeweiligen Feldes F$(I) mit den Koordinaten X(I) und F$(I) verantwortlich. Im Unterschied zur Zeile 180 kann hier das Setzen des Feldendes wegfallen, da eine Eingabe noch nicht stattfin-

den soll. Am Ende der Zeile 530 wird I mit Null initialisiert, als Zeichen dafür, daß wir uns jetzt im ersten Eingabefeld befinden.
In den Zeilen 540 und 550 findet eine Parameterübergabe für das uns bereits bekannte Eingabefeld statt. Mit GOSUB 170 erfolgt die Eingabe, mit F$(I)=F$ die Datenübergabe an das jeweilige Feld. Daraufhin überprüfen wir den zuletzt getätigten Tastendruck und nehmen in Zeile 570 die entsprechende Verzweigung vor. Bei Cursor nach unten geht es, wie auch bei einem Return, weiter im Programm. In Zeile 580 wird der Feldindex um eins erhöht, und sofern wir uns noch nicht im letzten Feld befanden, erfolgt nach Sprung auf Zeile 540 die Eingabe des nächsten Datenfeldes. Bei Cursor nach unten erniedrigen wir in Zeile 600 den Feldindex um eins, sofern wir nicht schon im ersten Datenfeld angelangt waren. Bei Control C erfolgt der Abschluß in Zeile 590.
Damit ist die Maskeneingaberoutine fertig. Um Ihnen das Programmieren zu erleichtern, haben wir für Sie in den Zeilen 680 bis 700 noch zwei weitere Routinen bereitgestellt, mit deren Hilfe Sie bequem die Felddaten aus Datazeilen übernehmen können.
Wir unterscheiden hierbei zweierlei Arten von Daten. Das eine sind die Feldparameter, wie sie in den Zeilen 800 bis 820 zu finden sind. Je drei Zahlen bestimmen Lage und Länge des jeweiligen Feldes. In den Zeilen 700 und 710 werden X(I), Y(I) und Z(I), entsprechend Bildschirmspalte, Bildschirmzeile und Feldlänge, eingelesen, bis die Zahl Minus Eins das Ende der Daten signalisiert.
Leere Felder sind nicht besonders informativ. Benötigt werden Erläuterungen, was in die Felder einzutragen ist. Diese Daten, Feldbeschriftungen und sonstiger Text finden sich in den Zeilen 750 bis 790. Die beiden Zahlen sind die Bildschirmkoordinaten, darauf folgt der Text. Die Zahl Minus Eins markiert wiederum das Ende.
Die Routine in den Zeilen 680 und 690 veranlaßt die Textausgabe.
Wie in den Zeilen 830 bis 860 zu sehen ist, gestaltet sich der Aufbau einer maskenorientierten Eingabe sehr einfach.

HAUPTPROGRAMM

In Zeile 830 löschen wir den Bildschirm und schalten den Kleinschreibmodus ein. In Zeile 840 definieren wir uns zur Abwechslung einmal andere Markierungen für die Kennzeichnung von Feldanfang und Feldende.
Mit einem RESTORE und einem GOSUB sind die Feldparameter schnell eingelesen. Genauso schnell bekommen wir in Zeile 850 die Feld-Erläuterungen mit einem RESTORE und einem GOSUB auf den Bildschirm. Für die Maskeneingabe benötigen wir jetzt nur noch einen GOSUB530. Zur Überprüfung geben wir die erfaßten Daten in Zeile 860 auf den Bildschirm aus.

EINBAU IN DAS EIGENE PROGRAMM

Ersetzen Sie unser Hauptprogramm durch Ihre eigene Anwendung, so stehen Ihnen mit den drei genannten GOSUB-Routinen die Mittel zur Verfügung, eine professionelle Maskeneingabe auf einfache Weise zu realisieren. Sie brauchen nur noch für die Übergabe der Daten an die Variablen F$(I) zu sorgen.
Wir sind bereits gespannt, wann wir Ihre Anwendung mit Eingabemaske zur Veröffentlichung zugesandt bekommen. *a.m.* □

EINGABEMASKE

```
10 rem datenmaske============c16 <mi>
20 rem (p) commodore welt team  <ho>
30 rem ======================== <ng>
40 rem (c) by alfons mittelmeyer <cg>
50 rem                          <pd>
60 rem                          <ah>
70 rem basic v3.5               <nl>
80 rem c16/116/plus4            <ki>
90 rem ======================== <jg>
100 ec$=chr$(27):v$=ec$+"j":mo$=ec
$+"a"                           <ki>
110 in$=chr$(13)+ec$+chr$(19):es$=
"ecjopqk"                       <ih>
120 l$=chr$(157)+chr$(157)+ec$+"b"
+chr$(147)                      <ch>
130 fa$=":":fe$=":":goto520     <ed>
140 rem ---------------------- <fi>
150 rem  fensterdefinition      <cc>
160 rem ---------------------- <mk>
170 char,x,y,fa$                <jn>
180 printec$"t"spc(z)ec$"m"fe$l$f$
v$mo$;                          <jg>
190 rem ---------------------- <ca>
200 rem  zeicheneingabe         <al>
210 rem ---------------------- <da>
220 ax=3072+40*peek(205)+peek(202) <fk>
230 pokeax,peek(ax)or128:getkeyx$:
pokeax,peek(ax)and127           <id>
240 rem ---------------------- <ch>
250 rem  zeichenauswertung      <ga>
260 rem ---------------------- <ob>
270 in=instr(in$,x$)            <ap>
280 oningoto340,410,300,340,340,34
0                               <ml>
290 printx$;:poke203,0:goto220  <ko>
300 x$="j":goto460              <pk>
310 rem ---------------------- <me>
320 rem  zeichenuebernahme      <la>
330 rem ---------------------- <dk>
340 printv$                     <nn>
350 poke239,1:poke1319,13       <pg>
360 sys34906:poke17,64:sys37210f$ <ke>
370 char,0,0,"":printec$"l"ec$"c";
:return                         <on>
380 rem ---------------------- <kn>
390 rem  esc-sequenz            <co>
400 rem ---------------------- <fc>
410 getkeyx$:in=instr(es$,x$)   <he>
420 oningoto450,450,460,460,440,46
0,470                           <id>
430 goto220                     <ka>
440 gosub350:goto170            <cp>
450 mo$=ec$+x$                  <kc>
460 printec$x$;:goto220         <lk>
```

```
470 printec$x$;:ifpeek(2024)<>peek
(202)thenprintchr$(29);                <ik>
480 goto220                            <lp>
490 rem ======================         <pm>
500 rem  maskeneingabe                 <le>
510 rem ======================         <af>
520 ma$=chr$(17)+chr$(145)+chr$(3)
:in$=in$+ma$:goto670                   <aj>
530 fori=0tom:gosub620:next:i=0        <no>
540 x=x(i):y=y(i):z=z(i)               <dj>
550 f$=f$(i):gosub170:f$(i)=f$         <fp>
560 f$(i)=f$:in=instr(ma$,x$)          <gk>
570 oningoto580,600,590                <ie>
580 i=i+1:ifi<=mthen540                <eg>
590 return                             <co>
600 i=i-1:ifi<0theni=0                 <cm>
610 goto540                            <ph>
620 char,x(i),y(i),fa$                 <kn>
630 printec$"t"spc(z(i))ec$"m"ec$"
p"fe$chr$(19)f$(i):return              <fj>
640 rem ----------------------         <cf>
650 rem  routinen fuer felddatas       <gk>
660 rem ----------------------         <dg>
670 goto750                            <mn>
680 reada:dountila=-1                  <ea>
690 readb:reada$:char,a,b,a$:reada
:loop:return                           <fc>
700 i=0:reada:dountila=-1              <fo>
710 x(i)=a:ready(i):readz(i):reada
:i=i+1:loop:m=i-1:return               <gp>
720 rem ======================         <ai>
730 rem  hauptprogramm                 <nf>
740 rem ======================         <kk>
750 data 0,3,"Name:",0,4,"Vorname:
"                                      <nn>
760 data 0,5,"Strasse:",0,6,"Haus-
Nr.:"                                  <lb>
770 data 0,7,"P L Z:",0,8,"Wohnort
:"                                     <lk>
780 data 0,9,"Vorwahl:",0,10,"Tel-
Nr.:"                                  <hf>
790 data 0,11,"Geb.Dat:",-1            <dj>
800 data 10,3,20,10,4,20,10,5,20       <le>
810 data 10,6,5,10,7,4,10,8,20         <cl>
820 data 10,9,5,10,10,12,10,11,10,
-1                                     <bd>
830 scnclr:printchr$(14)               <lk>
840 fa$=">":fe$="<":restore800:gos
ub700                                  <gk>
850 restore750:gosub680:gosub 530      <oe>
860 scnclr:fori=0tom:printf$(i):ne
xt                                     <oh>
870 rem ======================         <bn>
880 rem  programmende                  <aa>
890 rem ======================         <da>
```

Texten wie die Profis

Freie Beweglichkeit des Cursors auf dem Bildschirm, Scrollen bei Erreichen des Bildschirmrandes und diverse Befehle zum Blättern zeichnen eine gute Textverarbeitung aus. Mit einem modifizierten INPUT können auch von externen Geräten Daten eingelesen werden.

Eine Textverarbeitung zu programmieren, setzt ein gewisses Maß an programmiertechnischem Können voraus. Auch wenn solche Programme fehlerfrei arbeiten, richtig speichern und laden, mangelt es doch vielen an Bedienerkomfort. Häufig lassen sich die Zeilen nur hintereinander eingeben. Soll eine bereits erfaßte Zeile korrigiert, sollen Zeilen gelöscht oder eingefügt werden, ist nicht selten ein Wechseln in ein Änderungsmenü erforderlich. Anwenderfreundlich wären dagegen die freie Beweglichkeit des Cursors, das Auf- und Abwandern in den Zeilen, das Scrollen bei Erreichen der Bildschirmgrenzen. Daß dies auch in BASIC möglich ist, zeigt das Programm SCREEN-TEXT.

STEUER-FUNKTIONEN

Neben Cursor auf und ab gibt es zusätzliche Steuerfunktionen:

ESC I
= Zeile einfügen
ESC D
= Zeile löschen
Cbm N
= Zeile einfügen
Cbm Y
= Zeile löschen
Control E
= 15 Zeilen rückwärts
Control X
= 15 Zeilen vorwärts
ESC E
= Textanfang
ESC X
= Textende

ZEILENVERWALTUNG

Bemerkenswert ist die Zeilenverwaltung. Sie sorgt dafür, daß bei Einfügen oder Löschen von Zeilen nicht alle folgenden Zeilendaten F$(ZI) entsprechend verschoben zu werden brauchen. Dieses nähme zu viel Zeit in Anspruch und wäre deswegen ineffizient.
Die Zeilenverwaltung funktioniert mit zwei Stapeln, die in Form eines Strings realisiert sind. ST$ enthält in Form von Character-Codes die Nummern der noch zur Verfügung stehenden Datensätze, T$ die Nummern der bereits verwendeten Zeilenfelder.
Wird eine Zeile an- oder eingefügt, so wird für sie das Feld F$(ZI), dessen Nummer an der letzten Stelle von ST$ abgelegt ist, hergenommen. Dieser Code wird aus dem Stapel ST$ beseitigt und, gemäß der Position ZE der Zeile im Text, an die richtige Stelle im Stapel TS$ eingebaut. Die Anzahl der verbleibenden Felder LS wird um die Zahl Eins erniedrigt, die Anzahl der Textzeilen ZM um Eins erhöht. Vergleichen Sie hierzu am besten die Zeilen 540 bis 560 und 600 bis 640.

LESEN VON EXTERNEM GERÄT

Zum Einlesen der Daten von Datasette oder Diskette verwenden wir wieder die modifizierte INPUT-Anweisung, die wir be-

```
10 rem texteditor==========c16 <im>
20 rem (p) commodore welt team <ho>
30 rem ========================== <ng>
40 rem (c) by alfons mittelmeyer <cg>
50 rem <pd>
60 rem <ah>
70 rem basic v3.5 <nl>
80 rem c16/116/plus4 <ki>
90 rem ========================== <jg>
100 rem ------------------------- <op>
110 rem  zeicheneingabe <pe>
120 rem ------------------------- <po>
130 gosub1990:printchr$(14);:goto1
280 <go>
140 ax=3072+40*peek(205)+peek(202) <gk>
150 pokeax,peek(ax)or128:getkeyx$:
pokeax,peek(ax)and127:return <fn>
160 rem ------------------------- <mk>
170 rem  zeichenauswertung 2 <ha>
180 rem ------------------------- <mp>
190 gosub650:poke203,0:char,0,bz,"
" <mk>
200 printchr$(27)"t"spc(39)chr$(27
)"b"chr$(27)"m"chr$(27)"a"1$f$(zi) <hh>
210 in=instr(in$,x$) <eb>
220 oningoto250,270,240,250,250,25
0,250,250,250,250 <gn>
230 printx$;:poke203,0:gosub140:go
to210 <hd>
240 x$=chr$(27)+"j":goto230 <pd>
250 printchr$(27)"j":gosub420 <pm>
260 goto370 <ah>
270 getkeyx$:in=instr(es$,x$):ifx$
="p"thengosub420:x$="":goto190 <nb>
280 ifin=0thengosub140:goto210 <ff>
290 ifin<5thenprintchr$(27)"j":gos
ub420:goto490 <nf>
300 printchr$(27)x$; <eh>
310 ifx$="k"andpeek(2024)<>peek(20
2)thenprintchr$(29); <dl>
320 gosub140:goto210 <pn>
330 rem ------------------------- <dk>
340 rem  zeichenauswertung 1 <hk>
350 rem ------------------------- <ie>
360 gosub140:in=instr(in$,x$) <ih>
370 oningoto830,480,190,690,760,93
0,880,1180,1270,1520 <gb>
380 goto190 <id>
390 rem ------------------------- <fj>
400 rem  input <jn>
410 rem ------------------------- <oa>
420 poke239,1:poke1319,13 <jd>
430 sys34906:poke17,64:sys37210f$(
zi) <lj>
440 char,0,bz,"":printchr$(27)"l"c
hr$(27)"c";:return <ln>
450 rem ------------------------- <ni>
460 rem  esc-sequenzen <be>
470 rem ------------------------- <ia>
480 getkeyx$:in=instr(es$,x$) <pf>
490 oningoto880,930,1040,1110 <la>
500 ifin=0then360 <af>
510 poke239,2:poke1319,27:poke1320
,asc(x$):x$="":goto190 <jk>
520 rem ------------------------- <pg>
530 rem  zeile einfuegen <ak>
540 rem ------------------------- <eg>
550 ls=ls-1:zm=zm+1 <hb>
560 ze$=right$(st$,1):st$=left$(st
$,ls) <in>
570 ts$=left$(ts$,ze)+ze$+right$(t
s$,zm+1-ze):gosub650:f$(zi)="":ret
urn <hl>
580 rem ------------------------- <ha>
590 rem  zeile loeschen <jc>
600 rem ------------------------- <hm>
610 zi=asc(mid$(ts$,ze+1,1)):ls=ls
+1:zm=zm-1 <ni>
620 ze$=mid$(ts$,ze+1,1):st$=st$+z
e$ <bn>
630 ts$=left$(ts$,ze)+right$(ts$,z
m+2-ze) <ma>
640 f$(zi)="":return <oj>
650 zi=asc(mid$(ts$,ze+1,1)):retur
n <mi>
660 rem ------------------------- <jj>
670 rem  cursor down <gh>
680 rem ------------------------- <ef>
690 ifze=zmthen360 <ae>
700 ze=ze+1 <cf>
710 printx$;:ifbz<24thenbz=bz+1:go
to360 <fp>
720 x$="":goto190 <fk>
730 rem ------------------------- <de>
740 rem  cursor up <ea>
750 rem ------------------------- <ch>
760 ifze=1then360 <oo>
770 ze=ze-1:printx$; <dh>
780 ifbz>0thenbz=bz-1:goto360 <bp>
790 printchr$(27)"w";:goto720 <cb>
800 rem ------------------------- <bn>
810 rem  return-taste <bh>
820 rem ------------------------- <me>
830 ifze<>zmthen700:elseifls=1then
360 <ag>
840 ze=ze+1:gosub550:x$=c4$:goto71
0 <mh>
850 rem ------------------------- <ip>
860 rem  esc-i <ea>
870 rem ------------------------- <gg>
880 ifls=1then360:elsegosub550 <gf>
890 printchr$(27)"i";:goto360 <bm>
900 rem ------------------------- <fj>
910 rem  esc-d <pa>
920 rem ------------------------- <oa>
930 printchr$(27)"d"; <ei>
```

```
940 ifzm=1then360                          <nc>
950 gosub610                               <gl>
960 ifze>zmthenx$=chr$(145):goto77
0                                          <ho>
970 if(ze+24-bz)>zmthen360                 <bm>
980 zb=ze:ze=ze+24-bz:gosub650             <oc>
990 char,0,24,"":printchr$(27)"t"c
hr$(27)"m"spc(39)chr$(27)"b"chr$(2
7)"j"f$(zi);                               <gi>
1000 ze=zb:goto720                         <km>
1010 rem ------------------------          <ga>
1020 rem  esc-x                            <hg>
1030 rem ------------------------          <oe>
1040 bz=zm-1:ifbz>24thenbz=24              <bn>
1050 ze=zm-bz:printchr$(147)chr$(2
7)"m";                                     <ph>
1060 i=ze:forze=itozm:gosub650:pri
ntchr$(27)"q"f$(zi):next                   <nd>
1070 ze=zm:x$="":goto190                   <kh>
1080 rem ------------------------          <fh>
1090 rem  esc-e                            <pg>
1100 rem ------------------------          <om>
1110 i=zm:ifi>25theni=25                   <co>
1120 printchr$(147)chr$(27)"m";            <pb>
1130 forze=1toi:gosub650:printchr$
(27)"q"f$(zi):next:printchr$(27)"1
";                                         <pm>
1140 bz=0:ze=1:x$="":goto190               <hf>
1150 rem ------------------------          <cf>
1160 rem  ctrl-x                           <ef>
1170 rem ------------------------          <dg>
1180 ifze-bz+25>zmthen360                  <nf>
1190 ze=ze-bz+15:ee=ze+24:ifee>zmt
henee=zm                                   <mo>
1200 printchr$(147)chr$(27)"m";            <ec>
1210 i=ze:forze=itoee:gosub650:pri
ntchr$(27)"q"f$(zi):next:bz=0              <ce>
1220 printchr$(27)"l";                     <ng>
1230 ze=i:x$="":goto190                    <jo>
1240 rem ------------------------          <id>
1250 rem  ctrl-e                           <bd>
1260 rem ------------------------          <hn>
1270 ze=ze-bz-15:ifze<2then1110:el
se1200                                     <nm>
1280 rem ========================          <jh>
1290 rem  initialisierung                  <jf>
1300 rem ========================          <ma>
1310 rem maximale zeilenzahl               <ip>
1320 rem ist bei 16 kb speicher-           <bk>
1330 rem platz entsprechend                <cf>
1340 rem anzupassen                        <fb>
1350 mf=253:rem maximale zeilenzah
l                                          <ph>
1360 rem ------------------------          <db>
1370 dimf$(mf-1)                           <lg>
1380 in$=chr$(13)+chr$(27)+chr$(19
)+chr$(17)+chr$(145)+chr$(183)+chr
$(170)+chr$(24)                            <kd>
1390 in$=in$+chr$(5)+chr$(3)               <ng>
1400 es$="idxepacjoqk":l$=chr$(147
):goto1520                                 <gb>
1410 rem ------------------------          <bg>
1420 rem  laden & neuer text               <ba>
1430 rem ------------------------          <nj>
1440 st$="":fori=mfto1step-1               <ec>
1450 st$=st$+chr$(i):next                  <hn>
1460 ts$=chr$(0)+chr$(0)+chr$(0)           <el>
1470 bz=0:zm=1:ze=1:ls=mf                  <pc>
1480 zi=0:f$(zi)="":return                 <pf>
1490 rem ------------------------          <kj>
1500 rem  menue                            <gk>
1510 rem ------------------------          <dl>
1520 gosub1650                             <hk>
1530 print:printspc(5)"1 = Neuer T
ext"                                       <ac>
1540 printspc(5)"2 = Text Aendern          <oi>
1550 printspc(5)"3 = Drucken"              <gb>
1560 printspc(5)"4 = Laden von Kas
sette"                                     <pi>
1570 printspc(5)"5 = Speichern auf
 Kassette"                                 <ib>
1580 printspc(5)"6 = Laden von Dis
kette"                                     <pa>
1590 printspc(5)"7 = Speichern auf
 Diskette"                                 <da>
1600 print:printspc(5)"0 = Ende"           <ga>
1610 print:print" ----------------
--------------------"                      <bf>
1620 print:print" (Editieren beend
en = Control C)":print                     <dm>
1630 printchr$(27)"j";:gosub140:if
x$<"0"orx$>"7"then1630                     <ck>
1640 onval(x$)+1goto1830,1680,1810
,1870,1970,1980,1690,1940                  <em>
1650 scnclr:char,3,1,rn$+"Bildschi
rmorientierter Texteditor"+rf$             <an>
1660 print:print" ----------------
--------------------"                      <pk>
1670 return                                <kh>
1680 gosub1440:scnclr:x$="":goto19
0                                          <gh>
1690 gosub1960:open8,8,8,a$+",s,r"         <db>
1700 z=0:poke19,8                          <ec>
1710 gosub1440:ze=0:dountilst<>0           <og>
1720 poke2035,8:sys65478:sys34906:
poke17,64                                  <cj>
1730 sys37210f$(ze):ze=ze+1:loop           <kk>
1740 close8:poke19,0:zm=ze:ls=mf-z
m+1                                        <od>
1750 sys65484                              <gj>
1760 st$=left$(st$,ls)                     <de>
1770 ts$=chr$(0)                           <hn>
1780 fori=0toze-1                          <kn>
1790 ts$=ts$+chr$(i):next:ze=1             <ec>
1800 ts$=ts$+chr$(0)                       <nb>
1810 ifst$=""thengosub1440                 <oc>
```

TEXTVERARBEITUNG

```
1820 x$="":goto1110                                  <em>
1830 end                                             <nk>
1840 rem ------------------------                    <ml>
1850 rem  druckerausgabe                             <fd>
1860 rem ------------------------                    <ig>
1870 trap1890:open4,4,7                              <jb>
1880 forze=1tozm:gosub650:print#4,
f$(zi):next:close4:trap:goto1520                     <md>
1890 char,1,20,rn$+" drucker ansch
alten und taste druecken"+rf$                        <gi>
1900 getkeya$:resume1880                             <fn>
1910 rem ------------------------                    <ea>
1920 rem  speichern                                  <gj>
1930 rem ------------------------                    <jc>
1940 print"Textname: ? ";:sys34906
:poke17,64:sys37210a$::open8,8,8,a
$+",s,w"                                             <pd>
1950 forze=1tozm:gosub650:print#8,
f$(zi):next:close8:goto1520                          <do>
1960 print:input"Textname: ";a$:re
turn                                                 <fp>
1970 gosub1960:open8,1,0,a$:goto17
00                                                   <jp>
1980 gosub1960:open8,1,1,a$:goto19
50                                                   <dc>
1990 rem nachspann ===============                   <lk>
2000 rem * farbcodes/steuercodes *                   <bd>
2010 c4$=chr$(017):rn$=chr$(018)                     <oo>
2020 rf$=chr$(146)                                   <eo>
2030 return                                          <hj>
2040 rem ========================                    <ge>
2050 rem 60671 bytes memory                          <cc>
2060 rem 05699 bytes program                         <jd>
2070 rem 00154 bytes variables                       <ed>
2080 rem 00766 bytes arrays                          <nl>
2090 rem 11212 bytes strings                         <am>
2100 rem 42840 bytes free (0)                        <cb>
2110 rem (bei 253 vollen zeilen)                     <bg>
2120 rem ========================                    <pd>
```

reits vom Eingabefeld her kennen. Zum Lesen von einem externen Gerät sind noch ein paar zusätzliche Vorbereitungen nötig.
Nachdem eine Datei mit OPEN eröffnet wurde, kann noch nicht mit einem INPUT eingelesen werden. Wenn wir nicht INPUT# verwenden wollen, so ist noch der Eingabekanal zu öffnen. Das Öffnen eines Ausgabekanals mit CMD ist uns bekannt. Für das Öffnen eines Eingabekanals existiert kein BASIC-Befehl, jedoch eine Kernel-Routine.
Zunächst legen wir die Kanalnummer zum Beispiel mit POKE 2035,8 in die Speicherzelle, deren Inhalt bei einem SYS-Aufurf an das X-Register übergeben wird. Jetzt können wir mit SYS 65478 die Routine CHKIN aufrufen, die den Eingabekanal öffnet.
SYS 34906 liest die Daten vom externen Gerät in den Eingabepuffer.
Die Übergabe mit POKE 17,64 und SYS 37210F$ (ZE) bedarf noch einer Vorbereitung, damit kein unerwünschtes Bildschirmscrollen stattfindet. Hierzu ist lediglich in die Speicherzelle 19 ein Wert ungleich Null zu POKEn.
Dies braucht nicht unmittelbar vor dem Übernahme-SYS-Aufruf erfolgen, sondern kann bereits vor CHKIN stattfinden.
Wichtig ist, daß nach dem Abschluß des gesamten Einlesevorgangs die Kanäle mit SYS 65484 wieder geschlossen werden und auch der Eintrag in Speicherzelle 19 wieder mit POKE 19,0 die Tastatur signalisiert.

WEITERENTWICKLUNG AUF WUNSCH

Die Textverarbeitung kann als Konzept für eigene Weiterentwicklungen dienen. Ihr fehlen noch verschiedene Details, die von einem ausgereiften System erwartet werden. So ist sie lediglich, auf 40 Zeichen pro Zeile angelegt. ASCII-Texte mit mehr als 40 Zeichen pro Zeile werden nicht richtig wiedergegeben, ASCII-Texte mit mehr als 88 Zeichen pro Zeile führen gar zum Absturz. Es fehlen diverse Blockverschiebe-, Such- und Austauschbefehle. Spezielle Drukkersteuerzeichen können auch noch nicht definiert werden.
Falls Sie Interesse an einem weiterführenden Ausbau haben und uns Gestaltungsvorschläge unterbreiten, sind wir gerne bereit, die Textverarbeitung Ihrem Wunsch gemäß weiterzuentwickeln. *a.m.*□

Kontoführung

Beim Eintippen mit dem Taschen- oder Tischrechner ist schnell eine Fehlbuchung passiert. Der Computer erfaßt neben den Summen auch den jeweiligen Verwendungszweck und gestattet damit Überblick und Kontrolle, sowohl auf dem Bildschirm als auch auf dem Papier

Dieses Programm macht es möglich, beliebige Arten von Konten relativ übersichtlich zu führen. Die einzelnen Dateien werden sequentiell abgespeichert. Sie sind nach Monaten unterteilt. Der Umfang einer Datei ist durch die Vordimensionierung der Variablen auf 100 Buchungen beschränkt. Durch Abänderung der Zeile 90 kann bei Bedarf eine Erweiterung vorgenommen werden.
Nach dem Starten verlangt das Programm die Eingabe des Datums. Dies ist Grundlage für den Terminvermerk bei Buchungen. Danach gelangt man in das Hauptmenü:
1) DATEN EINGEBEN
2) DATEN AUSGEBEN
3) DATEN AENDERN
4) PROGRAMMENDE
Durch Drücken der entsprechenden Zahl gelangt man in die Unterprogramme. Die Trap-Anweisung in Zeile 170 bewirkt, daß alle anderen Eingaben ignoriert werden.

1) DATEN EINGEBEN

Zunächst erscheint die Directory der Diskette. Wählt man einen der bereits vorhandenen Monate an, so werden die entsprechenden Daten geladen. Anschließend gelangt man in die Eingabemaske. Hier können das Datum der Buchung, die Summe (Minus steht für Ausgaben, Plus muß nicht eingegeben werden) sowie der Verwendungszweck des Betrages eingetragen werden.
Will man weiter eingeben,

```
10 rem kontofuehrung ==========c16 <do>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by henning koglin       <ia>
50 rem                             <pd>
60 rem basic v3.5                  <od>
70 rem c16/116/p4                  <de>
80 rem floppy + cbm-drucker        <pl>
90 rem ========================== <jg>
100 gosub 1660                     <fc>
110 dimd$(100),s(100),v$(100),da$(
100)                               <nl>
120 color0,7,4:color4,7,4:color1,7
,2                                 <cm>
130 scnclr:char1,10,12,"datum (tt.
mm.jj)":inputdm$:ifdm$=""thendm$="
"                                  <ka>
140 rem ------------------------- <fi>
150 rem  hauptmenue                <od>
160 rem ------------------------- <mk>
170 trap170:x=0                    <bc>
180 scnclr:char1,10,08,"1) daten e
ingeben"                           <kb>
190 char1,10,10,"2) daten ausgeben
"                                  <ng>
200 char1,10,12,"3) daten aendern" <pg>
210 char1,10,14,"4) programmende"  <dh>
220 getz                           <ao>
230 onzgoto240,460,1140,1650:goto2
20                                 <mc>
240 rem ------------------------- <ch>
250 rem  daten eingeben            <bk>
260 rem ------------------------- <ob>
270 scnclr:directory:print:print:p
rint"fuer welchen monat";:inputdt$ <io>
280 gosub1550                      <pe>
290 scnclr:printrn$b$b3$b3$"eingab
e"b$b3$b4$rf$                      <eg>
300 char,0,4,"datum der buchung":c
har,0,6,"summe (+ oder -)"         <dl>
310 char,0,8,"verwendungszweck"    <le>
320 char,18,4,":":inputd$(x)       <cf>
330 char,18,6,":":inputs(x)        <im>
340 char,18,8,":":inputv$(x)       <ma>
350 da$(x)=dm$:x=x+1               <mi>
360 char,8,24,"weiter eingeben ? (
j/n)"                              <lf>
370 getz$                          <pg>
380 ifz$="j"then290                <fk>
390 ifz$="n"then410                <ca>
400 goto370                        <na>
410 gosub1450                      <eb>
420 goto170                        <aj>
430 rem ------------------------- <be>
440 rem  untermenue daten ausgeben <dn>
450 rem ------------------------- <ni>
460 ei=0                           <ln>
470 scnclr:char1,10,10,"1) ausgabe
n":char1,10,12,"2) einnahmen"      <fc>
480 char1,10,14,"3) bilanz":char1,
10,16,"4) zurueck"                 <ng>
490 geta:onagoto530,820,860,170:go
to490                              <mj>
500 rem ------------------------- <on>
510 rem  einnahmen/ausgaben        <jj>
520 rem ------------------------- <pg>
530 scnclr:directory:print:print:p
rint"fuer welchen monat";:inputdt$ <gj>
540 gosub1550                      <jf>
550 scnclr:char1,7,12,"ausgabe auf
 drucker (j/n)":inputjn$           <ln>
560 ifjn$="j"thendr=1:elsedr=0     <ca>
570 ifdr=1thenopen4,4              <kk>
580 scnclr                         <he>
590 ifdr=1thenprint#4,chr$(27)chr$
(108)chr$(10):print#4:print#4:prin
t#4                                <gl>
600 ifei=0thenprintrn$+"a u s g a
b e n"+rf$:elseprintrn$"e i n n a
h m e n"rf$                        <fh>
610 print:print:poke2022,3         <hm>
620 ifdr=1andei=0thenprint#4,chr$(
18)"a u s g a b e n"chr$(146)      <pb>
630 ifdr=1thenprint#4              <hn>
640 ifdr=1andei=1thenprint#4,chr$(
18)"e i n n a h m e n"chr$(146)    <bi>
650 ifdr=1thenprint#4              <fa>
660 fori=0tox-1                    <ph>
670 ifei<>1ands(i)<0thengosub730   <le>
680 ifei=1ands(i)>0thengosub730    <ld>
690 nexti                          <ca>
700 print"ende der liste !":getkey
a$:printhe$he$                     <oe>
710 ifdr=1thenclose4               <pn>
720 dr=0:goto460                   <ia>
730 print"datum der buchung: "d$(i
)                                  <mk>
740 print"verwndungszweck"b2$": "v
$(i)                               <hp>
750 print"summe"b$b2$": "s(i)" dm" <ai>
760 print"eintrag der buchung erfo
lgte am "rn$da$(i)rf$:print        <of>
770 ifdr=1thenprint#4,"datum der b
uchung: "d$(i):print#4,"verwendung
szweck : "v$(i)                    <na>
780 ifdr=1thenprint#4,"summe"b$b2$
": "s(i)" dm"                      <mi>
790 ifdr=1thenprint#4,"eintrag der
 buchung am "chr$(18)da$(i)chr$(14
6):print#4                         <lg>
800 getkeyr$                       <ol>
810 return                         <oi>
820 ei=1:goto530                   <dm>
830 rem ------------------------- <kp>
840 rem  bilanz                    <hk>
850 rem ------------------------- <ip>
```

```
860 trap460:scnclr:directory:print
:print:print"fuer welchen monat";:
inputdt$                          <mn>
870 gosub1550                     <ok>
880 scnclr:char1,7,12,"ausgabe auf
 drucker (j/n)":inputjn$          <fm>
890 ifjn$="j"thendr=1:elsedr=0    <ld>
900 ifdr=1thenopen4,4             <ib>
910 scnclr                        <al>
920 ifdr=1thenprint#4,chr$(27)chr$
(108)chr$(10):print#4:print#4:prin
t#4                               <jn>
930 printrn$"b i l a n z"rf$b5$"da
tum: "dm$:print:print:poke2022,3  <ka>
940 ifdr=1thenprint#4,chr$(18)"b i
 l a n z"chr$(146)b5$"datum:"dm$:p
rint#4:print#4                    <me>
950 fori=0tox-1                   <mi>
960 gosub1070                     <cj>
970 su=su+s(i)                    <ph>
980 nexti                         <kl>
990 sm=su*100:su=int(sm+.5):su=su/
100                               <kp>
1000 forp=0to39:print"-";:nextp   <kk>
1010 print:printspc(10)"summe :"ta
b(31-len(str$(su)))su" dm"        <bl>
1020 ifdr=1thenforp=0to60:print#4,
"-";:nextp                        <nk>
1030 ifdr<>1then1060:elsels=len(st
r$(su))                           <pm>
1040 print#4:print#4,b$b3$b3$"summ
e :"chr$(16)right$(str$(43-ls),2)s
u" dm"                            <nn>
1050 ifdr=1thenclose4             <pd>
1060 getkeya$:su=0:printhe$he$:got
o460                              <fb>
1070 printd$(i)tab(10)v$(i)tab(31-
len(str$(s(i))))s(i)" dm"         <ee>
1080 ifdr<>1thenreturn:elsels=len(
str$(s(i)))                       <bl>
1090 print#4,d$(i)b4$b4$v$(i)chr$(
16)right$(str$(43-ls),2)s(i)" dm" <ba>
1100 return                       <co>
1110 rem ------------------------ <kh>
1120 rem  daten aendern           <jf>
1130 rem ------------------------ <aa>
1140 trap1140                     <gd>
1150 scnclr:directory:print:print:
print"fuer welchen monat";:inputdt
$                                 <gg>
1160 open9,8,9,dt$+",s,r"         <ma>
1170 input#9,x                    <cc>
1180 fori=0tox-1                  <pb>
1190 input#9,d$(i)                <gc>
1200 input#9,s(i)                 <di>
1210 input#9,v$(i)                <df>
1220 input#9,da$(i)               <lg>
1230 nexti                        <jh>
1240 close9                       <ol>
1250 fori=0tox-1                  <kl>
1260 scnclr:printrn$"a e n d e r n
"rf$                              <dd>
1270 char,0,4,"datum der buchung"+
b5$:printd$(i)                    <ka>
1280 char,0,6,"summe":printspc(17)
s(i)                              <im>
1290 char,0,8,"verwendungszweck":p
rintspc(6)v$(i)                   <if>
1300 char,0,20,rn$+"aendern = a"+b
5$+"weiter = w"+b5$+"ende = e"+rf$ <fa>
1310 getq$                        <ge>
1320 ifq$="a"thengosub1400        <dg>
1330 ifq$="w"then1360             <jd>
1340 ifq$="e"then1380             <gp>
1350 goto1310                     <ng>
1360 nexti                        <kb>
1370 goto1380                     <gl>
1380 gosub1450                    <og>
1390 goto170                      <nk>
1400 char,20,4,"":inputd$(i):ifd$(
i)=""thend$(i)=" "                <dn>
1410 char,20,6,"":inputs(i)       <hn>
1420 char,20,8,"":inputv$(i):ifv$(
i)=""thenv$(i)=" "                <mp>
1430 da$(i)=dm$                   <gk>
1440 return                       <nj>
1450 rem ------------------------ <pd>
1460 rem daten speichern          <lj>
1470 rem ------------------------ <nf>
1480 open7,8,7,"@:"+dt$+",s,w"    <ac>
1490 print#7,x                    <mh>
1500 fori=0tox-1                  <la>
1510 print#7,d$(i):print#7,s(i)   <bd>
1520 print#7,v$(i):print#7,da$(i) <fj>
1530 nexti                        <em>
1540 close7:return                <ie>
1550 rem ------------------------ <gn>
1560 rem daten laden              <ge>
1570 rem ------------------------ <no>
1580 open8,8,8,dt$+",s,r"         <op>
1590 input#8,x                    <hf>
1600 fori=0tox-1                  <am>
1610 input#8,d$(i):input#8,s(i)   <np>
1620 input#8,v$(i):input#8,da$(i) <eb>
1630 nexti                        <nn>
1640 close8:return                <md>
1650 sys65529:rem programmende    <nm>
1660 rem nachspann ============== <ag>
1670 rem * farbcodes/steuercodes * <pm>
1680 rn$=chr$(018):he$=chr$(019)  <jc>
1690 rf$=chr$(146):b$=chr$(32)    <pm>
1700 b2$=b$+b$:b3$=b2$+b$         <kk>
1710 b4$=b3$+b$:b5$=b4$+b$        <mi>
1720 b$=b5$+b5$:return            <fk>
1730 rem ======================== <ml>
1740 rem 12277 bytes memory       <jl>
```

KONTOFUEHRUNG

```
1750 rem 04598 bytes program          <gf>
1760 rem 00140 bytes variables        <co>
1770 rem 01442 bytes arrays           <aj>
1780 rem 00475 bytes strings          <mk>
1790 rem 05622 bytes free (0)         <ao>
1800 rem (bei 3 datensaetzen)         <pj>
1810 rem ======================       <ma>
```

so erscheint wieder die Eingabemaske. Ist die Eingabe für den angewählten Monat beendet, so wird die jetzt umfangreiche Datei überschreibend abgespeichert.
Gibt man nach der Directory einen noch nicht vorhandenen Monat ein, so tritt zwar ein Fehler in der Floppy auf, der jedoch nicht weiter beachtet zu werden braucht.
Der anschließende Programmablauf entspricht dem oben beschriebenen.
Die neu eingerichtete Datei wird unter dem Namen des neuen Monats abgespeichert.
Nach dem Abspeichern gelangt man wieder ins Hauptmenü.

2) DATEN AUSGEBEN

Wählt man diesen Punkt an, so gelangt man in ein Untermenü:
1) AUSGABEN
2) EINNAHMEN
3) BILANZ
4) ZURUECK
Ausgaben listet nach Eingabe des Monats jeweils auf Tastendruck nacheinander die entsprechenden Ausgaben auf. Wählt man zuvor die Ausgabe auf den Drucker, so erscheint die Liste auf Bildschirm und Drucker. Nach Ende der Liste gelangt man wieder ins Untermenü.
Die Funktion *Einnahmen* funktioniert analog dazu; hier werden die Einnahmen des angewählten Monats ausgegeben.
Sowohl bei Ausgaben wie auch Einnahmen werden angegeben:

- Datum der Buchung
- Verwendungszweck
- Summe
- Datum des Eintrags der Buchung.

Mit der Funktion *Bilanz* erfolgt unter Angabe des Datums, des Verwendungszwecks und der Summe eine Aufrechnung aller Einnahmen und Ausgaben eines Monats gegeneinander. Am Ende wird die Gesamtsumme angegeben. Diese Liste kann auch über den Drucker erstellt werden. Durch Druck einer beliebigen Taste gelangt man nach Abschluß der Auflistung wieder ins Untermenü.
Mit der Funktion *Zurueck* wird wieder ins Hauptmenü gesprungen.

3) DATEN AENDERN

Bei diesem Unterprogramm erscheinen nacheinander alle Buchungen des angegebenen Monats. Man hat die Wahl, entweder zu ändern, weiterzublättern oder die Funktion zu beenden. Drückt man a für ändern, so springt der Cursor in die Ausgabemaske und man hat die Möglichkeit, die vorhandenen Daten zu ändern oder zu übernehmen. Mit w für weiter kann man die nächste Buchung anwählen. Das e für Ende speichert die komplette Datei mit den Änderungen ab.
Nach Abschluß einer Änderung verschwindet der Cursor und das Programm wartet auf die Eingabe einer der drei Tasten.

4) PROGRAMMENDE

Mit dieser Funktion wird über eine Systemroutine ein Software-Set ausgelöst. Damit ist das Programm gelöscht.

Henning Koglin □

Input, Stolperstein für Datenbanken

Der Programmautor merkt es nicht, der Programmtester auch nicht. Wird in jedes Datenfeld etwas eingetragen, so arbeitet das Programm fehlerfrei. Bleibt ein Datenfeld allerdings leer, so sorgt ein Betriebssystem-Fehler für Kummer.

Leser rufen an und melden, die Wolfsoft-Datei aus CW SPEZIAL 5/88 laufe nicht. Beim Programm-Test wurde eine Fehlfunktion allerdings nicht festgestellt. Nochmals werden Daten eingegeben, diesmal allerdings, damit es etwas schneller geht, einige Felder leer gelassen.
Nach dem Abspeichern wollen wir die Datei wieder laden. Doch die Floppy meldet rotes Licht, wird nicht fertig und im Programm geht es nicht mehr weiter.
Ob dies wohl mit einem Phänomen zusammenhängt, das kürzlich beim Schreiben eines Line-Editors aufgetreten ist?

Nach einigen mit Text gefüllten Zeilen wurden einige Zeilen leer gelassen, danach folgte weiterer Text. Nach dem Abspeichern und erneuten Laden waren die Leerzeilen verschwunden, der letzte Text schloß direkt an den vorangegangenen an.

Ein Blick in das Betriebssystem brachte zutage, daß beim Vorliegen eines leeren Datenfeldes der Rechner so lange weiterlist, bis er auf eines mit Daten trifft. Dieses wird dann anstatt der vorangegangenen Leerfelder gelesen. Ist die Datei allerdings zu Ende, so kann der Rechner lan-

INPUT-KORREKTUR

```
10 rem input-korrektur========c16 <el>
20 rem (p) commodore welt team    <ho>
30 rem ============================ <ng>
40 rem (c) by alfons mittelmeyer  <cg>
50 rem                            <pd>
60 rem                            <ah>
70 rem                            <bl>
80 rem c16/116/plus4              <ki>
90 rem ========================== <jg>
100 ax=320:rem mc-anfang          <ja>
110 fori=axtoax+24                <pb>
120 reada:pokei,a:next            <ag>
130 hb=int(ax/256):lb=ax-256*hb   <ja>
140 poke802,lb:poke803,hb         <di>
150 data 032,232,235,008,201,013  <co>
160 data 208,015,224,000,208,011  <pa>
170 data 166,019,240,007,162,032  <oi>
180 data 142,000,002,162,001,040  <jm>
190 data 096                      <gj>
200 rem ========================= <ne>
210 rem  p r o g r a m m e n d e  <jl>
220 rem ========================= <hm>
```

ge lesen, bis er auf Daten stößt.

FEHLERBEHEBUNG

Drei Möglichkeiten stehen zur Diskussion. Am besten ist es, wenn der Programmautor von vornherein dafür sorgt, daß keine leeren Datenfelder auftreten. Ein Leerzeichen würde schon genügen. Nach der Initialisierung der Eingabefelder durch ein Leerzeichen würde dieses auch nach einer leeren INPUT-Eingabe erhalten bleiben und auf die Diskette abgespeichert werden.
Bei einer Eingabe mit der GET-Routine braucht nur im Fall eines leeren Eingabefeldes ein Leerzeichen nachträglich untergebracht zu werden. Schon ist das Datenfeld nicht mehr leer, und das Einlesen mit INPUT# von einem externen Gerät bereitet keine Probleme.
Anstatt mit INPUT# kann auch in einer Laufschleife mit GET# eingelesen werden. Wir lesen Buchstabe für Buchstabe, bis wir auf ein Carriage-Return mit dem ASCII-Code 13 stoßen. Die gelesenen Buchstaben fassen wir zu einem String zusammen, und die bei INPUT# nicht mehr zu ladenden Daten sind wieder vorhanden.
Das Laden geht allerdings, da wir mit BASIC jeden Buchstaben einzeln lesen, etwas langsamer. Außerdem müssen wir zuvor das uns vielleicht nicht so vertraute Programm zerpflücken, um die entsprechenden Änderungen anzubringen.
Diese Arbeit können wir uns ersparen, wenn wir uns für die dritte Lösungsmöglichkeit entscheiden. Wir korrigieren den fehlerhaften INPUT#-Befehl. Findet die BASIC-Routine ein leeres Datenfeld, so mogeln wir für die weitere Verarbeitung durch den INPUT#-Befehl noch schnell ein Leerzeichen hinein: eine elegante Lösung.

Laden und starten Sie vor der Inbetriebnahme der Wolfsoft-Datei das Programm INPUT-KORREKTUR, und kein Ladefehler wird mehr auftreten. Falls Sie deutschen Zeichensatz benützen, legen Sie die kurze Maschinenroutine nicht in den Stack, sondern am besten in den Kassettenpuffer. Hierzu ist in Zeile 100 lediglich AX=320 in AX=818 abzuändern.

KORREKTUR DER WOLFSOFT-DATEI

Wenn Sie folgende Änderungen vornehmen, brauchen Sie für künftige Dateien keine Korrekturroutine mehr zu benützen:

```
1010 IF E$=" "
       THEN E$="  "
1011 PRINT:RETURN
```

Möchten Sie auch Ihre alten Dateien auf den neuesten Stand bringen, so sind in Zeile 1560 bis 1580 die Speicheranweisungen umzuarbeiten. Statt PRINT#7,NA$(I) können Sie schreiben E$=NA$(I):GOSUB 5000. Eine geeignete Routine sorgt für die Korrektur:

```
5000 IF E$=" "
       THEN E$="  "
5010 PRINT#7,E$:
       RETURN
```

Wenn Sie alle PRINT#7-Anweisungen in den obengenannten Zeilen auf die angegebene Weise umgearbeitet haben und die INPUT-KORREKTUR installiert ist, brauchen Sie Ihre alte

STETS DARAN DENKEN

Datei nur hereinzuladen und wieder wegzuspeichern. Es genügt fortan die Programmversion mit den Änderungen in den Zeilen 1010 bis 1011. Beim Programmieren sollten wir in Zukunft stets an die Eigenheiten des INPUT#-Befehls denken.

VERSEHEN DES AUTORS

Ein Versehen des Autors in Zeile 540 sorgt für ein Verschieben der Bildschirmmaske im Änderungsmodus. Statt "540 p4$=p4$+..." muß es richtig heißen "540 p5$=p5$+...". Damit dürften alle Fehler getilgt sein.
Interessant wäre eine richtige Eingabemaske – wie in diesem Heft besprochen – auch in der Wolfsoft-Datei und die Möglichkeit, mit Kassette zu arbeiten. Interessant mögen auch Suchroutinen in Verbindung mit der Postleitzahl sein. Sofern eine Überarbeitung und Erweiterung gewünscht wird, können Sie uns dies wissen lassen und weitere Vorschläge einbringen.

a.m. □

KORREKTUREN UND RICHTIGSTELLUNGEN

So wäre es besser gewesen

COMMODORE WELT SPEZIAL 5/88 Seite 44

WOLFSOFT-DATEI

```
540 p5$2p5$+...
anstatt
540 p4$=p4$+...
1010 ifes$2""then
       es$="  "
1011 print:return
3120 x$=z3$:m=35:gosub
       3880:printx$
```

In 540 wurde die falsche Variable angegeben. Dies führt zum Verschieben des Bildschirms im Eingabe- und Änderungsmodus. Leere Datenfelder verursachen nicht ladbare Dateien, deshalb ist die Änderung in den Zeilen 1010 und 1011 nötig. Vergleichen Sie hierzu den Artikel „INPUT, Stolperstein für Datenbanken". In Zeile 3120 wurde die falsche Subroutinen-Adresse benutzt, was zu einem Verschieben der Bildschirmausgabe führt.

COMMODORE WELT SPEZIAL 5/88 Seite 114

QUALLY GOES IN TOWN

Es kam bei einigen Lesern in gewissen Situationen zu einem OUT-OF-MEMORY-ERROR. Er tritt auch mit Speichererweiterung auf, da das Programm für den nicht erweiterten C16 geschrieben wurde. Der BASIC-Bereich wurde eingeschränkt, da ein veränderter Zeichensatz im Bereich von $3800 bis $3FFF abgelegt ist. Anscheinend ist der Speicherplatz so knapp bemessen, daß das Programm die Anfangszeile Nummer 10 und auch die Schlußzeile Nummer 2890 nicht mehr verträgt. Sollte der Speicherplatz dann immer noch nicht reichen, kann zusätzlich die Zeile 2590 herausgenommen werden.

COMMODORE WELT SPEZIAL 5/88 Seite 38

CÄSAR

```
5160 close1:c=10
```

Ein abgespeicherter Spielstand konnte erst nach der ersten Spielrunde geladen werden. In Zeile 4850 muß c den Wert zehn aufweisen, um in das Spiel zu kommen.

C16-P4-SPEZIAL
Nr. 1 Seite 92

KASINO

```
2030 char1,po,11,b3$
     :char1,po,13,b3$
     :gosub3190
2040 char1,po,10,b3$
     :char1,po,14,b3$
     :printc1$c1$c1$
     c2$c2$;k(kp):sound
     1,10,.2:return
```

Zwar liegt in diesem Listing kein Fehler vor, jedoch sind zwei Zeilen schlecht gedruckt und daher teilweise unleserlich.

C16-P4-SPEZIAL
NR. 1 Seite 115

MANAGER

Hier wären zwei zusätzliche Hinweise angebracht gewesen. Das BASIC-Programm von Seite 115 bis 133 ist unter dem Namen „manager.2" abzuspeichern, damit es vom Startprogramm nachgeladen werden kann. Kassettenbenutzer müssen zunächst das Startprogramm und erst anschließend „manager.2" abspeichern.

C16-P4-SPEZIAL
Nr. 1 Seite 47

MEGA-TOOL

Sicher haben Sie das Versehen in der Anweisung bemerkt. Natürlich ist MEGA-TOOL nicht unter dem Namen MANAGER abzuspeichern.

C16-P4-SPEZIAL
Nr. 1 Seite 9

DRUCKEN MIT JEDEM MODELL

Die Überschrift hätte wohl heißen müssen: Drucken mit Centronics-Druckern. Denn es gibt auch die Anschlüsse RS-232 und IEE-488, für die ein Centronics-Interface keinen Nutzen bringt. Leider ist uns ein Fehler in der Beschreibung, sowohl im Text als auch in der Pin-Belegungstabelle, unterlaufen. Die Busy-Leitung liegt nicht auf Pin eins des Centronics-Steckers, die Strobe-Leitung auch nicht auf Pin 11, sondern genau umgekehrt. Tauschen Sie also in der rechten unteren Ecke der Pinbelegungs-Tabelle die untereinander stehenden Zahlen eins und elf gegeneinander aus. Anlaß zur Kritik gab die Darstellung der Diode, die zur Annahme verleiten kann, es handle sich hier um zwei Dioden. Tatsächlich ist es nur eine einzige zwischen Pin vier des seriellen Ports und Pin zwei des Inverters, mit der Sperr-Richtung gegen den Inverter. Die Diode kann bei Verwendung eines Inverters oder vergleichbaren Bausteines mit offenem Kollektor auch weggelassen werden.

Prinzipschaltung des Centronics-Interface

Ergänzend ist zu vermerken, daß die Rechnermasse auf Pin zwei des seriellen Ports noch mit der Druckermasse auf Pin 16 des Centronics-Steckers zu verbinden ist. Die fünf Volt Spannung können, sofern der Drucker sie bereitstellt, von Pin 18 der Centronics-Schnittstelle abgegriffen werden, oder aber von Pin fünf eines Joystickports.

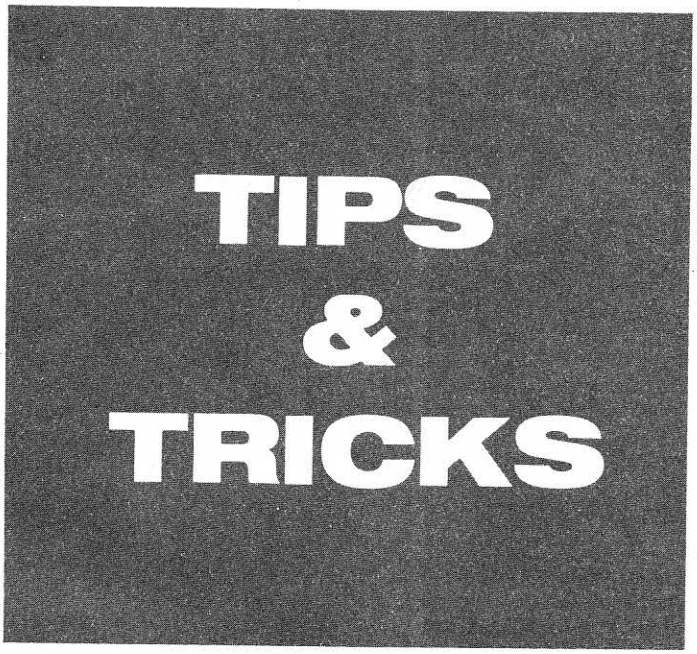
C16-P4-SPEZIAL
Nr. 1 Seite 29

ZEICHENSATZ-KOMPRIMIERUNG

Ein Fehler, der sich beim Arbeiten mit BASIC oder Script-Plus nicht bemerkbar macht, sorgte dafür, daß bei Verwendung des deutschen Zeichensatzes in der Tabellenkalkulation des Plus4 der Zeichensatz überschrieben wurde. Ursache dafür war, daß nach dem Heruntersetzen des BASIC-Endes der Zeiger auf den Stringspeicher nicht ebenfalls zurückgesetzt worden war. Ein CLR oder NEW würde bereits dafür sorgen. Wir ändern im Komprimierungsprogramm die Zeile 555:

```
555 data 49,58,162,
    0,0,0
```

Bereits komprimierte Programme können dadurch berichtigt werden, daß der END-Befehl im BASIC-Kopf durch ein NEW ersetzt wird. Es ist darauf zu achten, daß die Anzahl der Byte nicht verändert wird. Zusätzliche Eingaben sind daher zu unterlassen. Die DRUCKERANPASSUNG ist übrigens bei der Plus4-Software nur für die Textverarbeitung möglich.



COMMODORE WELT SPEZIAL 5/88 Seite 4

UMLAUTE MIT WIDERHAKEN

Das von uns getestete Zeichensatzmodul war auf Grund eines Produktionsfehlers defekt. Das Drucken von deutschen Umlauten ist also entgegen unserem Bericht auch mit Script-Plus möglich, sofern ein Commodore-Drucker mit deutschem Zeichensatz, wie der MPS 1000, der MPS 1200 oder ein vergleichbarer Kompatibler, zur Verfügung steht.

COMMODORE WELT 2/88 Seite 37

BASIC-TOOL

In den Tips zum Umgang mit BASIC-TOOL stand zu lesen, daß für den Diskettenbetrieb zwei Zeilen umzuändern seien. Dies ist richtig. Da das Programm umnumeriert wurde, stimmten aber die Zeilennummern nicht mehr.
Geändert werden müssen nicht die Zeilen 654 und 676, sondern die Zeilen 664 und 686.

C16-P4-SPEZIAL
Nr. 1 Seite 30

Ä Ö Ü – KEIN PROBLEM

Bei der Zeichensatzanpassung für die integrierte Software des Plus4 in SCREEN-KEYBOARD hatte sich ein Fehler eingeschlichen, der das Pi-Zeichen auf die für das Sternchen vorgesehene Taste verwies. In der Zeile 355 wurde der Code 42 vergessen. Richtig ist:

```
350 data 251,252,253,
    254,219,220
355 data 221,64,92,42,
    94,222
```

□

In Basic programmieren? Kein Problem!

Immer wieder erreichen uns Leser-Zuschriften, die ein großes Interesse an Anwendungsprogrammen bekunden. Viele dieser Leser geben auch ohne große Umschweife zu, daß ihre BASIC-Kenntnisse noch nicht sehr umfangreich sind, vor allem, wenn sie den Computer erst vor kurzem gekauft haben. Wir haben uns nun gedacht, künftig diesem Teil unserer Leser ein wenig den Einstieg in meistverbreiteste Programmiersprache der Welt zu erleichtern.

Anwendungsprogramme können sowohl professioneller als auch eigen-gestrickter Natur sein. Professionelle Programme können Sie kaufen und sofort einsetzen. Sie können aber auch versuchen, selbst welche zu schreiben. Da aber hakt es doch bei manchem, das Wissen reicht noch nicht ganz aus. Programmieren lernen Sie aber nicht nur durch Lesen von Fachbüchern und Zeitschriften, sondern vor allem durch Übung.

WAS SOLL DENN PROGRAMMIERT WERDEN?

Dabei tritt anscheinend bei dem einem oder anderen ein Problem auf, nämlich die Frage: Was soll ich denn programmieren? Die Antwort darauf ist eigentlich recht einfach: Ein Programm, das Ihr Anwendungsproblem löst. Um Ihnen aufzuzeigen, wie Sie ein Problem lösen können, werden wir eines darstellen und Ihnen einen Lösungsweg bieten und selbstverständlich ein Programm hierfür bieten und besprechen.
Sicher haben auch Sie schon einmal darüber nachgedacht, wie denn chiffrierte Nachrichten decodiert werden können. Im Zeitalter der Computer sind für solche Arbeiten zwar auch noch Spezialisten erforderlich, aber ein Großteil der – oft sehr mühseligen – Arbeit kann durch Computer in sehr kurzer Zeit erledigt werden.
Bestimmt haben Sie auch schon von verschiedenen Methoden der Verschlüsselung gehört. Möglichkeiten gibt es viele. Eine vielleicht bereits bekannte Art ist das Verschlüsselungsquadrat.
Das bedeutet, der zu codierende Text wird beispielsweise senkrecht in ein Rasterfeld mit gleicher Spalten- und Zeilenanzahl eingegeben und waagrecht ausgelesen.
Selbst wenn Sie aber bei einem Text wissen, daß er nach dieser Methode verschlüsselt wurde, ist es „zu Fuß" trotzdem noch schwer, die Decodierung durchzuführen. Da muß der Computer ran.

VERSUCHEN SIE ES ZUERST OHNE PROGRAMM!

In Zeile 130 des Listings sehen Sie eine „verschlüsselte" Mitteilung.
Versuchen Sie doch einmal, ohne Computerunterstützung herauszufinden, wie die Nachricht lautet. Wenn Sie es so schnell schaffen wie Ihr Commodore-Rechner mit Hilfe des abzutippenden Programmes, dann sind Sie absolut Spitze und könnten Ihren Computer eigentlich wieder verkaufen.

VOM PROBLEM ZUR LÖSUNG

Nun wissen Sie also, worum es geht und haben die Aufgabe, ein Programm zu schreiben, das diesen verschlüsselten Text als Klartext ausgibt. Bevor Sie nun Ihren Computer einschalten und versuchen, wie ein Weltmeister zu programmieren, sollten Sie vielleicht erst einmal überlegen, wie ein Lösungsweg aussehen könnte.
Bei einem Quadrat haben wir es mit einer Fläche zu tun, deren vier Seitenlängen gleich sind. Denken Sie zum Beispiel an ein Schachbrett.
Die Buchstaben des Klartextes wurden so auf die Felder geschrieben, daß der Text – links oben beginnend – nach rechts läuft, so wie eben normalerweise geschrieben und gelesen wird. Ausgegeben wird der Text aber derart, daß von oben nach unten gelesen werden muß, um ihn zu verstehen.
Obwohl Sie den codierten Text kennen, wissen Sie nicht, welche Seitenlänge dieses Verschlüsselungsquadrat hat – oder doch? Wenn, wie in unserem Beispiel, eine sehr einfache Codierung erfolgt, kann Ihnen die Länge des Textes diese Information liefern. Bei einer quadratischen Anordnung muß der Text im Quadrat komplett enthalten sein. In diesem simplen Beispiel muß die Quadratzahl der Seitenlänge also gleich oder größer der Zeichenketten(= String)-Länge sein. Das heißt, die nächstmögliche, ganzzahlige Wurzel aus der Länge der verschlüsselten Nachricht ergibt die Seitenlänge des Quadrates.

ES GEHT NICHT OHNE MATHEMATIK

Kurz zur Erinnerung und Auffrischung: Die Wurzel einer Zahl ist der Wert, der mit sich selbst multipliziert eine Quadratzahl ergibt, zum Beispiel Wurzel aus 49 = 7, weil 7 mal 7 eben wieder 49 ergibt. Wenn Sie nun beispielsweise einen Text mit dreißig Zeichen Länge haben, dann paßt dieser in ein minimales Feld mit einer Seitenlänge von 6, denn eine mit nur 5 ergäbe als maximale Textlänge nur fünfundzwanzig Zeichen (5 * 5 = 25).

Damit haben wir auch schon den Lösungsweg dieses Problems gefunden. Wir brauchen nun nur – beim ersten Buchstaben beginnend – den codierten Text vertikal (senkrecht) in ein gezeichnetes Quadrat einzutragen und horizontal (waagrecht) zu lesen. Aber wir wollen diese Aufgabe ja nicht selbst erledigen, sondern ein Programm entwerfen, das uns diese Arbeit abnimmt.

DIE PROGRAMM-STRUKTUR

Zunächst einmal wird bei Programmstart der Bildschirm gelöscht (PRINT

CHR$(147)). Damit wir aber auch sehen, welchen Zweck unser kleines Programm erfüllen soll, geben wir eine entsprechende Mitteilung durch den PRINT-Befehl aus.
PRINT ist die einfachste Anweisung an den Computer, etwas auszugeben, im Normalfall auf dem Bildschirm. Wenn wir Text ausgegeben haben wollen, dann kann dies so geschehen, daß nach dem PRINT-Befehl der gewünschte Text in Anführungszeichen eingeschlossen steht, dann handelt es sich um einen Textstring. Hierfür ist in unserem Programm Zeile 100 zuständig.
Zeile 110 bewirkt die Ausgabe des Textes: „Waagrecht schreiben" und in der folgenden Zeile „Senkrecht auslesen".
Als nächstes wollen wir dem Computer mitteilen, welcher Text decodiert werden soll. Dies kann beispielsweise über eine

PS
&
CKS

Zuweisung oder auch durch den INPUT-Befehl erfolgen (INPUT = Gib etwas ein!). Der INPUT-Befehl in seiner einfachsten Form benötigt aber einen Zusatz.
Der Computer muß sich ja unsere Eingabe vorübergehend merken können, deshalb gehört zum INPUT-Befehl noch die Zuweisung eines Variablen-Namens, unter dem Ihre Eingabe zwischengespeichert wird. Es können sowohl String- als auch reine Zahlen-Variablen sein.
Allerdings kennt Ihr Computer dabei nur ein Entweder–Oder: Haben Sie String-Variablen vorgesehen, so wird auch jede eingegebene Zahl als reine Zeichenkette (wie etwa Buchstaben) behandelt und interpretiert. Rechnen können Sie dann mit diesen Zahlen nicht, sondern sie nur wie einen Text auf dem Bildschirm ausgeben. So ein Variablenname wird immer mit einem davorgesetzten Semikolon (Strichpunkt) an die INPUT-Anweisung angehängt, in unserem Beispiel eben CO$.
Reine numerische Variablennamen werden ebenso angefügt, haben aber den Vorteil, daß die Eingabe nicht als Zeichenkette, sondern als im weiteren Programmverlauf als berechenbare Gleitkommazahl behandelt wird.
Allerdings dürfen Sie hier auch nur echte Zahlen eingeben, also keine anderen Zeichen wie Buchstaben, Interpunktions- oder Grafikzeichen, auch nicht mit einer Zahl gemischt (beispielsweise A1), das wird Ihr Computer immer grimmig mit einer Fehlermeldung quittieren.
Dann ist es für ihn nämlich eine Zeichenkette, ein String, und Sie haben ihn ja angewiesen, eine rein numerische Variable zu speichern, etwa INPUT"ZAHL:"; Z (also ohne $-Dollarzeichen!).
Verzeihen Sie uns die kleine Abschweifung zum Wesen des INPUT-Befehls, fahren wir in unserem Beispielprogramm fort.
Die Aufforderung zur Eingabe erfolgt – seitens des Computers – durch Ausgabe eines Fragezeichens (das mit ein paar Tricks auch unterdrückt werden kann) und des Cursorsymbols.
Bei Programmen, deren Erstellung schon längere Zeit zurückliegt, wissen wir nicht immer, was als Eingabe zu folgen hat. Eine kurze Mitteilung innerhalb des INPUT-Befehls ist daher unbedingt zu empfehlen.
Betrachten Sie dazu einmal Zeile 120. Allerdings haben wir zunächst diese Zeile durch das davorgesetzte REM außer Gefecht gesetzt. Alles, was nach so einer Anweisung in einer Programmzeile folgt, wird vom Computer ignoriert und nicht ausgeführt, sondern nur als Bemerkung betrachtet, auf Englisch REMARK, daher die Abkürzung REM.
Um Ihnen (und uns) die Arbeit zu erleichtern, haben wir in der folgenden Zeile 130 der Variablen CO$ die codierte Zeichenfolge direkt zugewiesen. Falls nämlich noch Programmierfehler enthalten sein sollten, dann braucht nach deren Korrektur diese sehr fremdländisch klingende Zeichenfolge nicht jedesmal mühsam von Hand eingegeben zu werden.
Für spätere Abläufe, also wenn das Programm fertig ist und fehlerfrei läuft, können Sie vor die Anweisung CO$ = ... ein REM setzen und dafür das in Zeile 120 (vor der INPUT-Anweisung) entfernen.

DURCH DIE ZEICHENKETTE WEISS DER COMPUTER ALLES NÖTIGE

Alles, was wir dem Computer an Informationen geben konnten, haben wir getan. Die Zeichenfolge ist ihm nun bekannt, die Länge dieses Strings kann er sich gefälligst selbst ausrechnen.
Die Anweisung dazu lautet LEN(String). LEN ist leicht zu merken (was übrigens für die meisten BASIC-Befehle gilt), wenn Sie ein bißchen Englisch können. Es wurde aus dem Wort length (=Länge) hergeleitet.
Die Befehlsfolge von LE = LEN(CO$) läßt sich am besten so übersetzen: Die Länge der Zeichenkette CO$ in der numerischen Variablen LE merken.
Nachdem der Computer die Länge berechnet hat, soll er die Wurzel ziehen, damit er die Seitenlänge des Quadrates weiß.
Die Wurzel (in diesem Fall die Quadratwurzel) wird durch den Befehl SQR(LE) gezogen. Da es aber keine halben Felder gibt, ist die Lösung nur dann richtig, wenn es sich bei der Wurzel um eine echte INTeger-Zahl handelt. Deshalb wird in Zeile 150 gleich festgelegt, daß für die Variable WU der Integer-Wert berechnet wird.
Wenn die Mitteilung aber nicht so lang ist, daß sie ein Quadrat ganz ausfüllt, dann sind noch die Spaces, die Leerstellen im Text. Dadurch würde sich über die Integerberechnung ein Fehler einschleichen.
Wir fangen dies in Zeile 160 ab, indem immer dann, wenn das Quadrat der Seitenlänge kleiner ist als die Anzahl der Zeichen, ein Leerzeichen angefügt und die Berechnung neu durchgeführt wird. Der Code fürs Leerzeichen ist 'CHR$(32)'.

DIE WURZEL DARF NUR GANZZAHLIG SEIN

Zur Verdeutlichung des eben Gesagten: Die Länge der Mitteilung beträgt 63 Zeichen. Berechnen wir daraus die Quadratwurzel, dann ergibt sich ein „krummer" Wert von 7,937.
Dieser Wert kann in unserem Falle nicht die Seitenlänge des Quadrates sein, denn wir arbeiten ja nicht mit Bruchteilen von Buchstaben oder Zeichen.
Deshalb die künstliche Verlängerung der Zeichenkette, eine neuerliche Berechnung der Wurzel und dann wieder ein Vergleich. Dies erfolgt so lange, bis der Wert für WU der Zahl 8 entspricht.
Nun ist die Seitenlänge bekannt. Was aber noch zu tun ist: Die einzelnen Buchstaben in korrekter Weise zusammenzufügen und auszugeben.
Zu diesem Zweck haben wir zwei Schleifen pro-

grammiert, die uns den jeweils aktuellen Buchstaben, im Programm EM$ genannt, herausfiltern. Dies passiert in den Zeilen 160 bis 210.
Dabei erledigen wir in diesen Zeilen gleich zweierlei: Einmal möchten wir den Klartext so ausgeben, wie er im ursprünglichen Quadrat stand, und zum anderen wollen wir ihn gleich in einen neuen String schreiben, der anschließend ausgegeben werden soll.
Wir wissen es und der Computer berechnet es aus der Länge; unser Beispieltext bestand aus einem 8 * 8-Quadrat. Deshalb brauchen wir nur, beim ersten Zeichen beginnend, jeweils immer das neunte (acht plus eins) Zeichen zu lesen, bis wir am Ende angelangt sind; dann wird der gleiche Vorgang wiederholt, nur diesmal beim zweiten Zeichen beginnend und so fort. Aber dafür haben wir ja das Programm entwickelt.

DAS CODIERUNGS-PROGRAMM WIRD ENTSCHLÜSSELT

Die Lösung ist recht einfach: Eine Schleife (Zeile 160) zählt immer von 1 bis WU und die andere von 0 bis LE, mit der Schrittweite von WU. Die Schrittweite wird durch den Befehl STEP und den anschließenden numerischen Ausdruck definiert (STEP WU).
Die Zeile 180 isoliert nun den jeweilig korrekten Buchstaben aus der Gesamtzeichenkette. Denken wir das einmal theoretisch durch.
Beim ersten Durchlauf der Zeilen 160 bis 210 ist der Wert in der Variablen J = 1, der Wert in K = 0. Das heißt, in der Zeile 180 wird der Stringvariablen EM$ der erste Buchstabe (= 'C') zugewiesen. Dies können wir leicht feststellen, wenn wir die Variablennamen in Zeile 180 durch ihre Werte ersetzen. Daraus ergibt sich folgender Befehl:

EM$ = MID$(CO$,
1 + 0, 1)

MID$, in dieser Form angewandt, ordnet EM$ die Zeichenkette aus CO$ zu, die an der Position J + K (=1) beginnt und ein Zeichen lang ist (,1).
Durch Zeile 190 wird die Zeichenkette für KL$ zusammengebaut. Beim ersten Durchlauf enthält KL$ deshalb das Zeichen C.

Codierter Text

```
10 rem codierter text                    <hp>
20 rem by lothar miedel                  <jl>
30 rem (c) commodore welt                <mi>
100 scnclr: print"loesung fuer ver
schluesselungsquadrat":print             <nl>
110 print"waagrecht schreiben,":pr
int"senkrecht auslesen!":print           <hd>
120 rem input"bitte text eingeben"
;co$                                     <em>
130 co$="cedsf n o-icucu mwehe1t m
e rr2z olzi 8e dtefd-r o-iteb  r t
 ne!"                                    <nf>
140 le=len(co$):wu=int(sqr(le))          <bf>
150 if(wu^2)<le thenco$=co$+chr$(3
2):goto140                               <ii>
160 forj=1towu                           <da>
170 fork=0tolestepwu                     <ob>
180 em$=mid$(co$,j+k,1)                  <kn>
190 kl$=kl$+em$                          <dg>
200 printem$"  ";                        <da>
210 nextk:print:nextj                    <mp>
220 print:print"klartext: ";kl$          <dk>
230 rem                                  <fm>
240 forj=1towu:fork=0tolestepwu          <fd>
250 em$=mid$(kl$,j+k,1):c1$=c1$+em
$:nextk,j                                <gj>
260 print:print"codierter text: "c
1$                                       <ih>
```

Durch Zeile 200 wird das jeweils aktuelle Zeichen und anschließend ein Leerzeichen ausgegeben. Das Semikolon am Ende dieser Zeile weist den Computer an, nun keinen Zeilenvorschub auszuführen, sondern in der gleichen Zeile zu bleiben.
Der Befehl NEXT K in Zeile 210 bewirkt, daß der Schleifenwert für K erhöht und dort weitergemacht wird, wo die eigentliche FOR-NEXT-Schleife begann.
Es wird deshalb in Zeile 170(!) fortgefahren. Diese bewirkt nun eine Erhöhung des Wertes für K um den Wert WU. Das heißt, K erhält eine Erhöhung um acht und hat nun den Wert 8. Wenn Sie die Zeile 180 nun wieder auflösen, dann steht dort:

EM $ = MID$ (CO$,
8 + 1, 1)

Das zweite Zeichen steht also nun an neunter Stelle und entspricht dem Buchstaben 0. Beim nächsten NEXT K wird wieder um acht erhöht und so weiter.
Wenn Sie sich die Mühe machen wollen, können Sie nun – Schritt für Schritt – den Programmablauf nachvollziehen.
Der Befehl PRINT in Zeile 210 bewirkt, daß bei der Ausgabe, nach jeweils acht Zeichen, ein Zeilenvorschub durchgeführt wird. Nach diesem wird dann der Wert für J erhöht. Da in der Schleife für J keine Schrittweite angegeben ist, wird jeweils um 1 erhöht.
Nach der ersten PRINT-Ausführung in Zeile 210 wird bei Zeile 160 weitergemacht. Der Wert für J ist dann 2. Gleichzeitig wird durch Zeile 170 der Wert für K wieder auf 0 gesetzt. Dadurch ergibt sich für Zeile 180 nun folgende Wertstellung:

EM $ = MID$ (Co$,
0 + 2, 1)

Das heißt, nun wird beim zweiten Buchstaben begonnen und die innere Schleife (wieder neu) abgearbeitet. Dadurch stehen die aktuellen Zeichen nun an den Positionen: 2, 10, 18, 26 usw.
Diese Vorgänge wiederholen sich so lange, bis der komplette Text entschlüsselt, also sowohl in der 8 * 8-Matrix ausgegeben als auch in KL$ (=Klartext) zusammengebaut ist. In Zeile 220 wird dann der decodierte String ausgegeben.

UND ALLES AUCH NOCH UMGEKEHRT

Die Zeile 240 bewirkt das Gegenteil, nämlich die Rück-Verschlüsselung des Klartextes (KL$) in die ursprüngliche Form. Deshalb kann dieses Programm auch zur Verschlüsselung eingesetzt werden. Es ändert sich dann lediglich immer der Sinn der Ausgaben. Aus Klartext wird codierter Text und umgekehrt.
Ein kurzer Hinweis bezüglich des Programmes sei uns noch gestattet: Bei der Ausgabe der Zeichenkette in der quadratischen Bildschirmdarstellung fehlt selbstverständlich eine Zeile (es sind nur sieben), denn der ursprüngliche Klartext war ja nur 50 Zeichen lang, der Rest waren Leerzeichen. *LM/hb*□

An uns über uns

EINIGE C16-PROGRAMME LAUFEN NICHT AUF DEM PLUS4

Ich bin Besitzer eines Commodore Plus4-Computers und habe mit einigen Software-Programmen, die für den Commodore C16 geschrieben wurden, Ärger.
Seltsamerweise funktionieren die Spiele (zum Beispiel Schatztaucher) auf dem Original-C16 wunderbar, auf dem auf 64 KByte erweiterten C16 oder Plus4 jedoch nicht. Bei den letztgenannten Computern bricht immer das Bild zusammen und man kann nichts mehr lesen beziehungsweise sehen.
Ich kann mir den Vorgang nicht erklären, da doch beide Computer, bis auf den Speicherplatz, gleich sein sollen. Selbst bei der softwaremäßigen Umstellung auf 64 KByte kann ich diese Spiele nicht zum Laufen bringen.
Andree Schumacher, Büdelsdorf

Woran es liegt, daß Programme, die auf dem C16 laufen, auf dem erweiterten C16 oder dem Plus4 jedoch nicht funktionieren, kann die verschiedensten Ursachen haben. Einmal genügte bereits die geänderte Funktionstasten-Belegung beim Plus4 oder eines C16 mit eingestecktem Software-Modul, daß das Programm nicht mehr recht wollte.
Einige Programme sind fehlerhaft programmiert. Wo es aber nun im Einzelfall hapert, kann nicht so ohne Weiteres gesagt werden. Wenn Sie solche Programme besitzen, die auf einem erweiterten Rechner nicht laufen, so informieren Sie uns am besten darüber, damit wir eine Liste darüber zur Veröffentlichung anfertigen können. Vielleicht weiß der eine oder andere unserer Leser bereits, woran es bei einigen Programmen krankt.

FEHLER IM CHECKSUMMER?

Nachdem ich das Programm CHECKSUMMER, wie ich meine, fehlerlos abgeschrieben und abgesaved habe, meldet er sich nach dem Laden und nach RUN auch mit READY.
Aber wenn ich nur die erste Zeile eines Programmes eingebe und mit RETURN abschließe, ertönt ein Klingelzeichen und auf dem Bildschirm erscheint folgendes:

```
   PC   SR AC XR YR SP
; 0152 30 64 22 00 F7
```

Vielleicht können Sie mir kurz und formlos sagen, was ich verkehrt mache.
Gerhard v. Wippern, Walldorf

Das, was auf dem Bildschirm zu sehen ist, kennzeichnet den typischen Absturz eines Maschinenprogramms. Steht im Hauptspeicher des Computers nicht das richtige Maschinenprogramm oder erfolgt der Sprung an eine falsche Stelle, so hängt sich entweder der Rechner in einer Laufschleife auf, oder trifft irgendeinmal auf den Operationscode 00, der einen Break-Interrupt auslöst. Das System geht dabei in den Maschinenmonitor, wo die Registerinhalte angezeigt werden.
Leider lassen sich hier kaum Rückschlüsse auf den Fehler im Programm ziehen. Es bleibt daher nichts anderes übrig, als das Programm nochmals auf das Genaueste zu überprüfen. Um die Fehlermöglichkeiten einzuschränken, haben wir darum den CHECKSUMMER entwickelt. Doch erst gilt es, diesen richtig abzutippen. Schade, daß dafür noch kein CHECKSUMMER zur Verfügung steht.

FEHLERHAFTER INPUT

Ich habe das Listing ab Seite 43 der CW-SPEZIAL Nr. 5/88 in meinen Commodore P4 eingegeben. Nach dem Wiederladen von der Floppy werden bis zu 20 Datensätze einwandfrei verarbeitet. Bei 21 Datensätzen gibt es schon ein Durcheinander. Hat man 25 Datensätze wieder eingeladen, erfolgt ein Programmabsturz. Woran kann das liegen?
Anmerkung: Die Checksumme der Zeile 1110 ist nicht oj sondern jj.
Karl-N. Wellbrock, Papenburg

FALSCHE CHECKSUMMEN?

Viele Leseranfragen zu diesem Programm haben uns schon erreicht. Ihren Niederschlag haben sie in den Korrekturhinweisen und einem eigenen Artikel zum INPUT-Befehl in diesem Heft gefunden.
Die andere Checksumme kommt durch ein geshiftetes Leerzeichen zustande. Ein normales Leerzeichen hat den ASCII-Wert 32, ein geshiftetes den Wert 160.
Unsere Checksumme wird stets für das normale Leerzeichen berechnet. In Programmzeilen mit Leerzeichen zwischen einzelnen Großbuchstaben werden erfahrungsgemäß gerne geshiftete Leerzeichen eingegeben. So ist es nicht verwunderlich, wenn viele glauben, die Checksumme stimme nicht. Das Ersetzen mit normalen Leerzeichen beweist es. Wenn die Checksumme dann immer noch nicht stimmen sollte, so liegt noch ein Tippfehler vor.

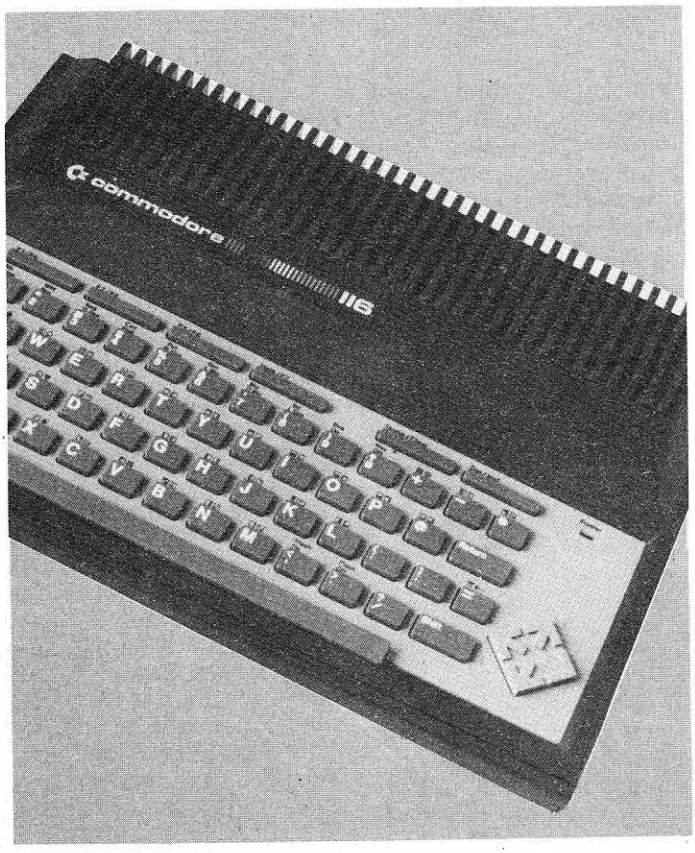
DEUTSCHER ZEICHENSATZ MIT ZUSÄTZLICHEM SYS-AUFRUF

Mit Interesse habe ich in Ihrem jüngsten Sonderheft C16-P4-SPEZIAL Nr. 1 den Artikel „Deutscher Zeichensatz" gelesen. Insbesondere die ZS-Komprimierung beschleunigt die Programme zur Zeichensatz-Anpassung ungemein. Meine Frage lautet nun, ob es durch Ergänzung dieses Programmes durch den entsprechenden SYS-Befehl möglich ist, in einem Arbeitsgang nach dem Laden und Starten des komprimierten Zeichensatzes die integrierte P4-Software beziehungsweise Script-Plus zu aktivieren?
Bei der nicht komprimierten Fassung des SCREEN-KEYBOARD ist dies ja ohne weiteres möglich (anstelle des END-Befehls SYS 1525 für die eingebaute P4-Software). Wichtiger aber wäre dies in Verbindung mit Script-Plus, das ja die Taste F2 mit dem SYS-Befehl belegt und somit den DLOAD-Befehl ersetzt. Dann könnte das Programm jeweils zu Beginn der Text-Diskette abgespeichert werden und mit der Commodore-Taste in Verbindung mit RUN/STOP in einem Vorgang geladen, gestartet und die entsprechende Textverarbeitung aktiviert werden. Ich würde mich freuen, wenn Sie mir mitteilen würden, ob der Einbau der SYS-Befehle in das Programm ZS-Komprimierung in diesem Sinne möglich ist.
Rüdiger Weber, Vilkerath

Möglich ist im Grunde genommen fast alles. Nur oft sind Dinge nicht so einfach zu realisieren.
Im BASIC-Kopf der ZS-Komprimierung dürfen keine zusätzlichen Eintragungen gemacht werden. Wir haben aber bereits eine neue Version des Komprimierungsprogrammes entwickelt, die einen zusätzlichen SYS-Aufruf erlaubt. Im nächsten C16-P4-SPEZIAL werden wir sie veröffentlichen.

Mikrocomputer-Grundlagen, Teil 6

Befehle und Adressierungsarten der CPU 6502



Neben bedingten und unbedingten Sprungbefehlen stellt uns die CPU 6502 verschiedene Befehle, Register und Adressierungsarten zur Verfügung, mit denen wir Zugriff auf unsere Daten in den diversen Speicherplätzen erhalten. Da Datenverarbeitung auch des Datenzugriffes bedarf, zeigen wir, wie dieser erfolgt.

Um Speicherinhalte zu laden, Registerinhalte zu speichern oder zu vergleichen, stellt uns die CPU 6502 drei Register und die entsprechenden Load-, Store- und Compare-Befehle zur Verfügung:

CPU-Register	Load	Store	Compare
A-Register	LDA	STA	CMP
X-Register	LDX	STX	CPX
Y-Register	LDY	STY	CPY

Das A-Register, der Akkumulator, ist ein echtes Universalregister. Wir können mit ihm, außer den bereits erwähnten Operationen, auch die verschiedensten logischen, arithmetischen und Schiebe-Operationen durchführen.

LOGISCHE OPERATIONEN

ORA: Bitweises Odern. Jedes Bit des Akkumulator-Inhaltes wird mit dem entsprechenden Datenbit auf folgende Weise verknüpft:

Akkubit x Datenbit→Akkubit

0	x	0	→	0
0	x	1	→	1
1	x	0	→	1
1	x	1	→	1

Beeinflußte Statusbits:

NZ – – – –

AND: Bitweises Undieren

Akkubit x Datenbit→Akkubit

0	x	0	→	0
0	x	1	→	0
1	x	0	→	0
1	x	1	→	0

NZ – – – –

EOR: Exklusives bitweises Odern:

Akkubit x Datenbit→Akkubit

0	x	0	→	0
0	x	1	→	1
1	x	0	→	1
1	x	1	→	0

NZC – – –

BIT: Bitweises Undieren ohne Veränderung des Akkuinhaltes, sondern nur des Statusregisters. Zusätzlich zur Vorzeichen- und Zeroflag wird hier auch das Überlaufflag beeinflußt. Der beim Undieren resultierende Wert von Bit 6 gelangt in das Überlaufflag.

NZ – – – V

ARITHMETISCHE OPERATIONEN

ADC: Addieren mit Übertrag (Carry). Akkuinhalt und Datenbyte werden addiert, das Ergebnis wird wiederum im Akku abgelegt. Bei gesetztem Carry wird der Akkuinhalt zusätzlich im eins erhöht.

NZC – – V

SBC: Subtrahieren mit Übertrag (Borrow). Das Datenbyte wird vom Akkuinhalt subtrahiert und das Ergebnis im Akku abgelegt. Bei nicht gesetztem Carry (= gesetztes Borrow) wird der Akkuinhalt zusätzlich um eins vermindert.

NZC – – V

SCHIEBEBEFEHL

Beeinflußte Statusbits: N,Z,C. Bei Akkuadressierung werden die Bits des Akkuinhaltes, sonst die der adressierten Speicherstelle verschoben. Die CPU 6502 kennt vier Schiebebefehle: ASL, LSR, ROL und ROR.

ASL
Arithmetic Shift Left. Bit 7 wandert in das Carry, Bit 0 bis 6 werden um eins nach links verschoben, so daß deren Inhalt sich darauf in Bit 1 bis 7 wiederfindet. Bit 0 wird auf Null gesetzt. Diese Operation entspricht einer Multiplikation mit dem Faktor 2.

LSR
Logical Shift Right. Analog zu ASL, nur in umgekehrter Richtung. Bit 0 gelangt ins Carry, Bit 1 bis 7 gelangen ins Bit 0 bis 6, Bit 7 wird auf Null gesetzt.

ROL
Rotate Left. Im Unterschied zu ASL wird Bit 0 nicht von Hause aus Null, sondern übernimmt seinen Wert vom Carryflag, welches im Anschluß den Wert von Bit 7 bekommt.

ROR
Rotate Right. Wie LSR, mit dem Unterschied, daß Bit 7 seinen Wert vom Carryflag übernimmt.

INDEX-REGISTER

Die Register X und Y sind, anders als der Akkumulator, nicht für Rechenoperationen vorgesehen, sondern stehen

als Index-Register hauptsächlich für die indizierte Adressierung zur Verfügung. Um ihrem Zweck gerecht werden zu können, lassen sie sich per Befehl um den Wert eines vermindern (decrement) oder erhöhen (increment): Increment, Decrement.

INX, DEX
INY, DEY

Nicht nur die Register der Index-Befehle können dekrementiert oder inkrementiert werden, sondern auch Speicherstellen. Beeinflußt werden hierbei die Flags:

NZ – – – –

Um Daten an den Akku oder das Stapelregister zu übergeben oder von ihnen zu übernehmen, existieren einige Transfer-Befehle:

TXA	X → A
TAX	A → X
TYA	Y → A
TAY	A → Y
TXS	X → S
TSX	S → X

NZ – – – –

Mit Ausnahme von TXS werden hier, wie auch bei LDA, LDX und LDY das Vorzeichenflag N und das Zeroflag Z beeinflußt. Die Speicherbefehle STA, STX und STY haben keinen Einfluß auf das Flagregister. Die Vergleichsbefehle CMP, CPX und CPY entsprechen einer Subtraktion, ohne Veränderung des entsprechenden Registerinhaltes und ohne Heranziehung des Carryflags.
Beeinflußt wird nur das Statusregister in der gleichen Weise wie bei der Subtraktion.

ADRESSIERUNGS-ARTEN

Wollen wir Daten mit einem Registerinhalt verknüpfen, so muß dem Prozessor gesagt werden, wo sich diese im Hauptspeicher unseres Rechners befinden. Wir unterscheiden folgende Adressierungsarten:

- Unmittelbar
- Direkt
- Indiziert
- Indirekt

Um Zeit und Speicherplatz zu sparen, kennt unsere CPU neben einer absoluten Adressierung auch die sogenannte Zeropage-Adressierung. In Kombination mit den vorhergehenden ergibt sich eine Vielfalt möglicher Adressierungen, wovon die CPU 6502 folgende besitzt:

- Unmittelbar
- Direkt, Zeropage
- Direkt, Absolut
- Indiziert X, Zeropage
- Indiziert Y, Zeropage
- Indiziert X, Absolut
- Indiziert Y, Absolut
- Indiziert-indirekt X, Zeropage
- Indirekt-indiziert Y, Zeropage
- Akkumulator

Diese Liste ist nicht ganz vollständig, umfaßt aber alle Arten, mit denen wir uns näher befassen werden. Die bedingten Sprünge mit der relativen Adressierung, der indirekte Sprung und die implizierte Adressierung wurden bereits in der vorhergehenden Folge erwähnt.

Unmittelbare Adressierung

Dem Operationscode folgt ein Datenbyte. Dieses wird je nach der auszuführenden Operation mit dem entsprechenden Register verknüpft.

Zeropage-Adressierung

Zeropageadressen sind Adressen von 0 bis 255 ($00-$FF). Dem Operationscode folgt nur ein einziges Adreßbyte, das die gewünschte Speicherzelle in der Zeropage angibt.

Absolute Adressierung

Absolute Adressen gehen von 0 bis 65535 ($0000-$FFFF). Dem Operationscode folgen zwei Byte, und zwar zuerst das LOW-, dann das HIGH-Byte. Die adressierte Speicherstelle AD errechnet sich durch AD=256*HB+LB.

Direkte Adressierung

Der Inhalt einer Speicherzelle wird entweder mit einem Register verknüpft oder zum Beispiel durch eine Lade-, Verschiebe-, Decrement- oder Increment-Operation verändert. Dem Operationscode folgt entweder ein Adreßbyte bei Zeropage-Adressierung oder zwei Byte bei absoluter Adressierung.

Indizierte Adressierung

Zur Adressierung einer Speicherzelle wird zusätzlich der Inhalt eines Index-Registers herbeigezogen. Dieser, zu der dem Operationscode folgenden Adresse addiert, ergibt die Adresse der Speicherzelle, auf welche sich die auszuführende Operation tatsächlich bezieht. Hierbei ist den Zeropage-Adressen besondere Beachtung zu schenken. Sei beispielsweise 240 der Wert des Adreßbyte und 100 der Inhalt des Indexregisters: Die adressierte Speicherstelle ist nicht, wie vielleicht zu erwarten, Speicherstelle Nummer 340, sondern Nummer 84. Es lassen sich bei indizierter Zeropage-Adressierung nur Speicherstellen in der Zeropage adressieren. Sollte der Wert, gebildet aus Adreßbyte und Indexregister, die Zahl 255 übersteigen, so ist AD= AD–256 die Nummer der adressierten Speicherstelle. Für das Übersteigen des Wertes 65535 bei absoluter Adressierung gilt analog AD=AD–65535.

Indirekte Adressierung

Indirekte Adressierung bei der CPU 6502 gibt es nur in der Zeropage. Die Operation bezieht sich hierbei nicht auf die durch das Adreßbyte wiedergegebene Speicherstelle. Vielmehr findet sich dort und in der nachfolgenden Speicherstelle, wieder in der Folge LOW-Byte und HIGH-Byte, nur die Adresse der tatsächlich angesprochenen Speicherstelle. Zur Adreßbildung wird außerdem noch der Inhalt eines Index-Registers herbeigezogen, und zwar auf zweierlei Art und Weise.

Indiziert-indirekte Adressierung

Verwendet wird hierbei der Inhalt des X-Registers. Dieser, addiert zum Wert des Adreßbyte, ergibt die Adresse der zwei Speicherstellen, in welcher die Adresse der anzusprechenden Speicherstelle vermerkt ist.

Indirekt-indizierte Adressierung

Verwendet wird hierbei der Inhalt des Y-Registers. Dieser, addiert zum Inhalt der durch das Adreßbyte bestimmten zwei Speicherstellen, ergibt die Adresse der anzusprechenden Speicherstelle.
Bei der indiziert-indirekten Adressierung wird also der Inhalt des X-Registers gleich zum Wert des Adreßbyte ad-

MONITOR	ASSEMBLER	ADRESSIERUNG
LDA #$NN	LDA,#,Z	unmittelbar
LDA $NN	LDA,%,Z	direkt Z-Page
LDA $NNNN	LDA,$,Z	direkt absolut
LDA $NN,X	LDA,%X,Z	indiziert Z-Page
LDA XNN,Y	LDA,%Y,Z	indiziert Z-Page
LDA $NNNN,X	LDA,$X,Z	indiziert absolut
LDA $NNNN,Y	LDA,$Y,Z	indiziert absolut
LDA ($NN,X)	LDA,@X,Z	indiziert-indirekt
LDA ($NN),Y	LDA,@Y,Z	indirekt-indiziert

diert und den so bestimmten zwei Speicherstellen die letztendliche Adresse entnommen. Bei der indirekt-indizierten wird der Inhalt des Y-Registers zuerst zum Inhalt der durch das Adreßbyte bestimmten Speicherstellen addiert und so die eigentliche Adresse gebildet.

Syntax der Adressierung

Wir ziehen zum Vergleich mit unserem eigenen Assembler den Maschinenmonitor des C16/116/Plus4 oder des C128 heran. Die Syntax unseres Assemblers ist ziemlich einfach gehalten, um dem BASIC nicht allzuviel Arbeit zuzumuten. Die Übersicht gibt uns die nebenstehende Tabelle. Bei unserem Assembler wird die Adressierung bereits durch die dem Operations-Mnemonic folgende Kennung bestimmt. Beim Maschinenmonitor muß eine weitergehende Syntaxprüfung erfolgen, was diesem allerdings, da in Maschinensprache programmiert, keine besonderen Probleme bereitet. Der große Nachteil des Monitors gegenüber dem Assembler besteht darin, daß keine Variablen verwendet werden können und die entsprechenden Adressen direkt als Hexzahlen einzugeben sind.

Wenn wir von der unmittelbaren Adressierung absehen, so unterscheidet der Monitor die absolute und die Zeropage-Adressierung durch die Anzahl der Hexziffern, die in unserer Tabelle mit dem Buchstaben N gekennzeichnet sind. Unser Assembler betrachtet die der Adreßkennung folgende Zahl Z als die Nummer einer von uns zu definierenden Variablen. Wir können sie, analog zu der Labeldefinition im ersten Teil unseres Assemblerkurses, mit ADR oder = ADR festlegen. Eine Ausnahme ist die unmittelbare Adressierung, bei der die Zahl Z unser Datenbyte darstellt.

Diese Adressen werden, um einen Unterschied zu machen, nicht in die Labelliste mit eingetragen, sondern in einer gesonderten Adreßliste erfaßt. Der Vorteil davon ist, daß nachträgliche Änderungen des zu assemblierenden Codes jederzeit vorgenommen werden können, ohne daß wir uns um eine Neuberechnung aller Adressen zu kümmern brauchen.

Adressierung mit Offset

Um auch die einer definierten Adresse nachfolgenden Speicherstellen anzusprechen, kann der Nummer unserer Adresse ein Offsetparameter voranstehen:

```
Offsetparameter &0
LDA,$,&1,Z
```

In diesem Fall wird nicht der Inhalt der durch Z gekennzeichneten Speicherstelle in den Akku geladen, sondern jener der folgenden. LDA,S,&2, Z würde den Inhalt der übernächsten Speicherstelle laden.

Verwendung von Konstanten

Damit für unterschiedliche Systeme unterschiedliche Ausgabecodes kein Problem darstellen, haben wir eine Konstantendefinition eingeführt. So könnte ein Commodoredrucker für den Buchstaben "ä" den Wert 187 benötigen, ein IBM-Drucker dagegen den Wert 132. Mit =KONST,0,187 würden wir der Konstanten 0 den Wert 187 zuweisen. Folgendes Beispiel würde daraufhin in jedem Falle die Ausgabe von "ä" auf den Drucker bewirken, sofern es sich bei Label 0 um die Druckerausgabe handelt:

```
Druckerausgabe
LDA,#,K,0
JSR,0
```

Unmittelbares Laden eines Adreßbyte

Oft möchten wir nicht den Inhalt einer Speicherstelle verarbeiten, sondern deren Adresse. Auch hierfür haben wir uns einen entsprechenden Zusatz überlegt:

```
Adreß-Lowbyte
LDA,#,&L0,Z
Adreß-Highbyte
LDA,#,&H0,Z
```

Da eine absolute Adresse immer mit zwei Byte, nämlich Low- und Highbyte, wiedergegeben wird, haben wir die Zusätze &LO und &HO eingeführt. Den Buchstaben L und H folgt noch ein Offset O, damit wir auch die Adressen der anschließenden Speicherstellen in den Griff bekommen.

Akku-Adressierung

Sie ist vergleichbar mit der impliziten Adressierung, bei welcher nur ein Byte Operationscode ohne weitere Adreßparameter in den Speicher des Computers geschrieben wird. Der Unterschied besteht darin, daß mit dem Operations-Mnemonic bei der implizierten Adressierung bereits die ganze Sache klar ist. Schiebebefehle wie ROR, ROL, ASL und LSR können sich jedoch außer auf den Akkumulator auch auf zu adressierende Speicherstellen beziehen. Wir müssen daher dem Assembler sagen, daß wir lediglich den Akkumulatorinhalt zu manipulieren wünschen. Der Maschinenmonitor entnimmt dies bereits dem Fehlen der Adreßangaben, wir verwenden bei unserem Assembler den Buchstaben "A":

```
MONITOR ASSEMBLER

ASL     ASL,A
LSR     LSR,A
ROL     ROL,A
ROR     ROR,A
```

Erläuterung eines Assembler-Listings

Hier ein Beispiel eines Assembler-Listings, welches den Bildschirm 256 Mal mit je einem anderen Zeichen füllt:

Zeile 130:
Adresse Nummer 2 erhält den Anfang des Bildspeichers zugewiesen.
Zeile 140:
Anfangsadresse des Maschinenprogrammes ist 14336.
Zeile 150:
Wir setzen das Interruptflag, um den Systeminterrupt zu unterbinden, und hoffen darauf, daß unsere Maschine dadurch ein wenig schneller wird.
Zeile 160:
Wir markieren einen Schleifenanfang mit LAB,3. Mit JSR,0 rufen wir die Subroutine auf, die den Bildschirm mit dem in Adresse Nummer 1 lagernden Zeichen füllen soll.
Zeile 170:
Wir erhöhen den Wert des in Adresse Nummer 1 liegenden Code, um das nächste Zeichen ausgeben zu können. Bei der Inkrementierung wird auch, je nachdem, ob der neue Zeichenwert Null beträgt oder nicht, das Zeroflag entsprechend gesetzt.
Zeile 180:
Sind noch keine 256 Ausgaben erfolgt, so ist das Zeroflag nicht gesetzt. Es erfolgt

256 AUSGABEN

nun der Rücksprung nach Label Nummer 3, woraufhin die restlichen Zeichen ausgegeben werden können. Nach 256 Ausgaben wird, infolge der vorangegangen Inkrementanweisung, der Zeichenwert wieder auf Null gesetzt. Somit haben wir ein gesetztes Zeroflag vor uns. Der bedingte Sprung wird daher nicht ausgeführt.
Zeile 190:
Vor der Rückkehr in unser BASIC müssen wir den Interrupt wieder zulassen, damit Tastatureingaben möglich sind.
Zeile 200:
Rücksprung in das aufrufende Programm beziehungsweise ins BASIC.
Zeile 210:
Mit LAB,0 wird der Anfang unserer Unterroutine, die für das bildschirmfüllende Aus-

geben unseres Zeichens verantwortlich ist, gekennzeichnet. Wir besorgen uns jetzt mit LDX,#,&LO,2 das Lowbyte der Bildschirmadresse.
Zeile 220:
Wir greifen zu einem besonderen Trick, der nur in Maschinensprache möglich ist: Um uns nicht mit dem Pro-

PROGRAMM-MANIPULATION

blem herumzuschlagen, wo in der Zeropage bei den verschiedenen Rechnern ein freier Speicherplatz für die indirekte Adressierung ist, ändern wir ein Adreßbyte der in Zeile 280 stehenden STA-Anweisung. Da dem Operationscode zuerst das Lowbyte folgt, wird dieses geändert.
Zeile 230:
Wir besorgen uns das Highbyte der Bildschirmanfangsadresse.
Zeile 240:
Das Highbyte des Adreßbyte der in der Zeile 280 stehenden STA-Anweisung wird ersetzt.
Zeile 250:
Den Akku laden wir mit dem auszugebenden Zeichen aus Adresse Nummer 1.
Zeile 260:
Das Y-Register laden wir mit dem Wert 4, da der Bildschirmspeicher 4*256 Byte umfaßt. Wir verwenden das Y-Register als Zähler, der uns Aufschluß darüber geben soll, wie oft wir noch 256 Byte auszugeben haben.
Zeile 270:
Das X-Register setzen wir als Index-Register für die nachfolgende STA-Anweisung ein. Wir wollen es von 0 bis 255 hochzählen, um auf diese Weise 256 Byte zu übertragen. Den Anfangswert setzen wir auf Null.
Zeile 280:
Mit LAB,1 setzen wir eine Schleifenmarkierung. Die Adreßdefinition ADR,1 ermöglichte die in den Zeilen 220 und 240 erfolgte Manipulation der Adreßbyte. Die Nummer 0 der durch die STA-Anweisung adressierten

BEISPIEL-LISTING

```
10 rem assembler---------------cbm <kl>
20 rem (p) commodore-welt-team    <ie>
30 rem ========================== <ng>
40 rem (c) by alfons mittelmeyer  <cg>
50 rem                            <pd>
60 rem                            <ah>
70 rem programmausschnitt         <lc>
80 rem aus cw 2/88                <nd>
90 rem ========================== <jg>
100 rem ------------------------- <op>
110 rem  zu assemblieren          <cp>
120 rem ------------------------- <po>
130 data =adr,2,3072:rem screen (c
64/128 =adr,2,1024)               <nj>
140 data org,14336                <og>
150 data              sei         <kg>
160 data lab,3,       jsr,0       <hc>
170 data              inc,$,1     <bh>
180 data              bne,3       <me>
190 data              cli         <oo>
200 data              rts         <od>
210 data lab,0,       ldx,#,&10,2 <pj>
220 data              stx,$,&1,0  <gg>
230 data              lda,#,&h0,2 <dl>
240 data              sta,$,&2,0  <pp>
250 data              lda,$,1     <cg>
260 data              ldy,#,4     <ei>
270 data              ldx,#,0     <fg>
280 data lab,1,adr,0, sta,$x,0    <dh>
290 data              inx         <gg>
300 data              bne,1       <oi>
310 data              dey         <ia>
320 data              beq,2       <lk>
330 data              inc,$,&2,0  <lk>
340 data              jmp,1       <hn>
350 data lab,2,       rts         <co>
360 data adr,1,db,0               <ol>
370 data end                      <dm>
380 rem ------------------------- <kn>
390 rem       e n d e             <bk>
400 rem ------------------------- <fc>
```

Speicherstelle hat aufgrund unserer Manipulation nichts mehr zu sagen. Vielmehr steht dort die gewünschte Bildschirmspeicheradresse.
Ist der Wert des X-Registers Null, gelangt der Akkuinhalt in die so adressierte Zelle. Erhöht sich der Inhalt des X-Registers, können auch die 255 nachfolgenden Zellen angesprochen werden.
Zeile 290:
Wir erhöhen den Inhalt des X-Registers.
Zeile 300:
Es wird so oft auf die STA-Anweisung in Zeile 280 zurückgesprungen, bis der Inhalt des X-Registers schließlich wieder Null wird.
Zeile 310:
Wir erniedrigen unser als Zähler fungierendes Y-Register, um nachzuprüfen, ob bereits 4*256 Byte ausgegeben sind.
Zeile 320:
Sollte der Inhalt des Y-Registers Null sein, so ist die Arbeit unserer Subroutine beendet, und es kann die Verzweigung auf die in Zeile 350 stehenden Rückkehranweisung erfolgen.
Zeile 330:
Sind die nächsten 256 Byte auszugeben, so müssen wir das Highbyte der in Zeile 280 stehenden STA-Anweisung erhöhen.
Zeile 340:
Weiter geht es mit der STA-Anweisung in Zeile 280.
Zeile 350:
Mit LAB,2 markieren wir den Zielpunkt der in Zeile 320 erfolgenden Verzweigung. RTS veranlaßt die Rückkehr auf Zeile 170 in unserem Hauptprogramm. Dieses Byte Programmcode hätten wir uns

RÜCKSPRUNG

eigentlich schenken können, da in erreichbarer Nähe, nämlich in Zeile 200, eine RTS-Anweisung zur Verfügung gestanden hätte.
Zeile 360:
Hier nehmen wir die Definition der Adresse Nummer 1 vor und füllen bereits bei der Assemblierung diese Speicherstelle mit dem Wert Null. Das Programm selbst läßt uns hinterher diese Speicherstelle wieder mit dem Wert Null zurück, so daß wir das Programm immer wieder mit demselben Resultat aufrufen können.
Zeile 370:
Assembleranweisung zur Beendigung des Assembliervorganges.

Vorausschau

In der nächsten Folge erhalten Sie einen zusammenfassenden tabellarischen Überblick über den CPU-Befehlssatz mit Angabe der Taktzyklen, die die einzelnen Operationen zur Abarbeitung benötigen. Auf kaum erklärte Befehle soll noch näher eingegangen werden. Dann können wir uns getrost an das Programmieren in Assembler heranwagen. *a.m.* □

2500 PROGRAMMIERSPRACHEN ZUR AUSWAHL

Wie in Babylon

Von Dipl.-Ing. (FH) Oliver Rosenbaum

Der vielschichtige, schnell wachsende Markt der Programmiersprachen ist auch für Insider kaum noch überschaubar. Unsere Serie erläutert die Hauptunterschiede. In der zweiten Folge: Zwanzig weitere Programmiersprachen – von Exapt bis OPS-5.

29. Exapt

Extended Subset of **APT**.
Weiterentwicklung der Werkzeugmaschinen-Steuersprache APT (für numerische Steuerungen). Exapt ermöglicht neben der zweidimensionalen Steuerung auch die Berücksichtigung des Werkstoffes (zweieinhalbdimensionale Steuerung). Neben APT (siehe dort) ist Exapt eine der wichtigsten Sprachen zur Steuerung von NC-Maschinen.
Exapt wird ebenfalls eingesetzt für automatisches Zeichnen, ist aber problemorientiert, also maschinenunabhängig zu programmieren.
Ein sinnvoller Einsatz von Exapt setzt eine Großrechenanlage voraus.
Exapt wurde in Deutschland auf der Basis des amerikanischen APT erstellt und ist daher auch in Englisch gehalten.
1965 wurde Exapt an den Technischen Hochschulen in Aachen, Berlin und Stuttgart in Zusammenarbeit mit der Industrie entwickelt.
Man unterscheidet drei Sprachbereiche:

Exapt 1 für Bohr- und Fräsbearbeitung mit Punkt- oder Streckensteuerung,
Exapt 2 für Drehmaschinen mit Strekken- und Bahnsteuerung,
Exapt 3 für Fräsbearbeitung mit zweieinhalbdimensionaler Bahnsteuerung.

Exapt-Programme bestehen aus drei Sprachteilen. Bei der Übersetzung werden daher drei Übersetzungsprogramme benötigt.
Die jeweiligen Übersetzungen werden getrennt voneinander vorgenommen, da jeder Teil für sich sehr umfangreich ist.
Man unterscheidet:

- geometrischer Prozessor
- technologischer Prozessor
- Postprozessor.

Im geometrischen Prozessor werden im ersten Maschinendurchlauf alle geometrischen Definitionen, die das Programm enthält, über das Werkstück und die Peripherie verarbeitet.
Beim zweiten Maschinendurchlauf wird die Technologie der Bearbeitung des Werkstückes bestimmt.
Zur Auswertung kommen Schnittwerte, Werkzeuge, Stoff- und Werkzeugdaten, die meist schon vorab gespeichert wurden.
Der Postprozessor stellt die Verbindung des Programmes mit der NC-Maschine vor; dies ist notwendig, da Exapt problemorientiert ist (siehe oben).
Der Vorteil dieser Dreiteilung liegt in der größeren Beweglichkeit, der Einsparung von Speicherplatz und der besseren Fehlersuche und -korrektur.
Programmbeispiel siehe Tabelle 8.

Programmbeispiel zu Exapt
Listingausschnitt

1 PARTNO/FLANSCH RO 83-17	Die Anweisung kennzeichnet den Programmanfang. Hinter dem "/" steht der Name des Werkstückes mit der Zeichnungsnummer.
2 MACHIN/PP1,SETPOS,-150,-150,1	Hier steht die Werkzeugmaschine, auf der das Teil angefertigt werden soll.
3 TRANS/167.2,199.3,0.5	Angegeben sind die drei Koordinaten der Position, in der das Werkstück eingespannt wurde.
4 P1=POINT/100,60,25	Definition von Punkt 1
5 P2=POINT/15,20,15	Definition von Punkt 2
6 P3=POINT/185,20,15	Definition von Punkt 3
7 ZSURF/25	Z-Koordinate des Bohrplans
8 K1=CIRCLE/CENTER,P1,RADIUS,(80/2)	Definition des Teilkreises K1
9 K2=PATERN/ARC,K1,0,CLW,4	Definition der Gewindelöcher auf K1
10 ZSURF/15	Definition der Z-Koordinate von L1 und L2
11 L1=PATERN/LIN,P2,ATANGL,90,INCR,2,AT40	Definition der Bohrung L1
12 L2=PATERN/LIN,P3,ATANGL,90,INCR,2,AT40	Definition der Bohrung L2
13 PART/MATERL.27	Materialnummer
14 A1=REAM/DIAMET,30,DEPTH,25	Arbeitszyklus "Reiben" zur Herstellung einer Bohrung, Durchmesser 30 mm
15 A2=TAP/DIAMET,10,DEPTH,15,TAT,1,BLIND,1	Herstellung eines M10 Gewindes
16 A3=DRILL/DIAMET,10,DEPTH,15	Definition von 10mm Bohrungen
17 A4=SINK/SO,DIAMET,20,DEPTH,5,SPIRET,1	Senkoperation mit Durchmesser 20mm
18 CLDIST/2	Sicherheitsabstand von 2mm
19 COOLNT/ON	Kühlmittelzugabe wird gestartet
20 FROM/10,10,0	Definition der Ausgangsposition der ersten Bewegung
21 WORK/A1	Die Reibebearbeitung wird gestartet
22 GOTO/P1	bei Punkt 1
23 WORK/A2	Die Gewindeoperation wird gestartet
24 GOTO/K2	bei Punkt K2 (Teilkreis)
25 WORK/A3,A4	Die Bohrbearbeitung (10mm) und die darauffolgende Senkbearbeitung wird gestartet.
26 GOTO/L1	bei L1
27 CODLTA/12,NOCUT	Anheben des Werkzeuges beim überschweben von Hindernissen
28 GOTO/L2	Zweite Bohrbearbeitung (wie 25)
29 FINI	Ende des Programms

Tabelle 8:

30. FOCUS

Sprache für Expertensysteme, siehe auch MANTIS, oder andere DL-Sprachen wie SQL.

31. Forth

Forth ist ein stackorientiertes Programmsystem, das etwa seit 1980 einem breiteren Anwenderkreis durch die Implementierungen für Micro-Computer zugänglich gemacht wurde.
Entwickelt wurde Forth zehn Jahre früher von Charles Moore auf der legendären IBM 1130.
Forth sollte das System der „vierten Generation" werden, daher der Name (die IBM 1130 ließ maximal fünfstellige Namen zu).
Heute gibt es preiswerte Implementationen für die verschiedensten Rechnertypen und -größen:

- das Standard-Forth der Forth-Interest-Group, auch Fig Forth genannt;
- Forth 79, eine Erweiterung des Fig Forth;
- C-64 Forth für den Commodore, das speziell den C64 und seinen grafischen Eigenschaften angepaßt wurde, –

um nur die bekanntesten zu nennen.
Das Forth-System ist äußerst portabel. Implementationen gibt es für fast alle Prozessortypen. Die hier vorliegende Be-

schreibung von Forth basiert auf der C64-Version. Forth unterscheidet sich in einigen markanten Punkten von anderen Programmiersprachen:

- Forth arbeitet mit der RPN (Reverse Polish Notation, umgekehrten Polnischen Notation) hier auch Lifo (Last in first out);
- Fort ist stackorientiert und damit maschinennäher als andere höhere Programmiersprachen; daraus ergibt sich eine höhere Ausführungsgeschwindigkeit der Programme sowie eine bessere Speicherausnutzung;
- Bei der Programmierung in Forth wird zugleich mit einem Interpreter und einem Compiler gearbeitet (interpretierender Compiler).
  Forth besitzt zudem ein eigenes Betriebssystem;

- Das Forth-System ist relativ klein und benötigt nur etwa acht bis zwölf KByte;
- Es können eigene Befehle entworfen und in das System integriert werden;

- Das Forth-System ist äußerst transparent und in sich selbst definiert, was auch Änderungen am System selbst erlaubt.

DIE REGISTER IN FORTH

Forth hat grundsätzlich zwei verschiedene Stapelregister: eines für die Verarbeitung von Zahlen und ein weiteres für definierte Wörter, Befehle und anderes.
In Forth gibt es nur wenige fertig vordefinierte Befehle; sie beschränken sich auf ein unbedingt notwendiges Minimum, ähnlich wie bei Assembler-Sprachen.
Neu definierte Befehle (sie basieren auf den vorhandenen) werden in ein „Wörterbuch" abgelegt. Der TOD (Top Of Dictionary) zeigt immer auf den zuletzt definierten Befehl.
Forth-Programme werden in sogenannten Screens (Bildschirmseiten) auf Diskette abgespeichert. Ein Screen entspricht dem Inhalt eines kompletten Bildschirmes.
Mit Hilfe des im Forth-Betriebssystem enthaltenen Editors können einzelne Screens beschrieben, geändert oder gelöscht werden.

DER BEFEHLSVORRAT IN FORTH

DUP dupliziert den obersten Stackwert TOS (=Top of Stack);
SWAP vertauscht die beiden obersten Stackwerte;
DROP löscht die oberste Position im Stack;
OVER kopiert die vorletzte Zahl in den obersten Stack (TOS);
ROT vertauscht die oberen drei Stackwerte zyklisch (ROTation);
nPICK kopiert die n-te Zahl in den TOS;
n ROLL vertauscht die ersten n-Stackwerte (zyklisch);
MOD liefert den Rest einer Division;
ABS Absolutwert einer Zahl;
MIN bestimmt die kleine von zwei Zahlen;
MAX liefert den größeren von zwei Werten.

In Forth gibt es außerdem vergleichende Operatoren wie in BASIC: größer, kleiner, gleich.

DO...LOOP
begrenzt Schleifen in Forth analog der For-Next-Schleife in BASIC;

IF..THEN..ELSE
bedingte Verzweigung:

BEGIN...UNTIL
BEGIN...AGAIN
Schleifen mit bedingtem Ende;

```
----------------------------------------
Programmbeispiel zu Forth:
Listingausschnitt
----------------------------------------
3 $VARIABLE A$  3 $VARIABLE B$  10 $VARIABLE C$
: CKFILE
  ?FILE IF
    CR."FILENAME " PAD $.. "NOT FOUND"  0
  ELSE 1 THEN ,

: COPYSCR
  CR ."SCREENCOPY FROM : "  A$  INPUT$
  CR. "SCREENCOPY TO   : "  B$  INPUT$
  OPENCHN
  " SCR" A$ $+ C$ $!
  14 8 14 C$ " ,S,R" $+ COUNT OPEN
  CKFILE IF
    " SCR" B$ $+ C$ $!
    13 8 13 C$ " ,S,S" $+ COUNT OPEN
    CKWRSTAT
    14 HERE     1024 GET#     CKWRSTAT
    13 HERE  1+ 1024 PRINT#   CKWRSTAT
  THEN
  13 CLOSE  14 CLOSE  CLOSCHN  CR;
----------------------------------------
```

Tabelle 9:

BEGIN..WHILE..REPEAT
bedingte Schleife;

INPUT
Eingabe von Variablen in das laufende Programm;

GET INKEY
Eingabe eines Zeichens in das laufende Programm.
Außerdem sind Stringmanipulationen möglich wie in BASIC (siehe dort):
LEFT$, RIGHT$, MID$, LEN, ASC, CHR$

FILEHANDLING

OPEN
Öffnen eines logischen Files;

CLOSE
Schließen eines logischen Files;

ABORTIO
Schließen aller noch offenen logischen Files;

GETDEV
holt die Devicenummer eines eröffneten Files zurück;

RDSTAT
liest den Diskettenstatus von der Disk;

CKST
liest den I/O-Status (ST-Variable);

CKRDSTAT
Lesen des Fehlerkanals;

OPENCHN
eröffnet den Fehlerkanal der Diskette;

CLOSCHN
schließt den Fehlerkanal.

Programmbeispiel siehe Tabelle 9.

32. Fortran

Formular **T**ranslator (Formel-Übersetzer).
Wie aus dem Namen schon hervorgeht, ist Fortran in erster Linie zur Lösung von technisch-mathematischen Problemen entwickelt worden. Man spricht hier auch von einer „Formelsprache". Fortran ist die heute am meisten verbreitete Programmier-Sprache in diesem Anwendungsbereich.
Fortran bietet aufgrund guter Compilerversionen eine hervorragende Fehlersuche bei der Pro-

grammerstellung. Erkannt und gemeldet werden:

- Anweisungsfehler: Syntaxfehler, semantische Fehler, konstruktive Fehler;
- Programmfehler: Programmumfangsfehler, Compilerfehler, nicht verwendete Variable oder Unterprogramme;
- Laufzeitfehler: Speicherbereichsfehler, Ein-/Ausgabefehler.

Angegeben vom Compiler wird die Art des Fehlers und die Stelle im Programm, an der er auftritt *(vgl. Tabelle 10).*

Es gibt vier verschiedene Arten von *Anweisungen* in Fortran:

1. *Eingabe- und Ausgabeanweisungen.*
   Sie regeln den Datenfluß innerhalb der Rechenanlage und zu oder von den Peripheriegeräten.
2. *Steueranweisungen.*
   Hierdurch wird der Programmablauf manipuliert. Die Steueranweisungen bestimmen die Reihenfolge der Abarbeitung der Befehle und Programmteile.
3. *Spezifikationsanweisungen,* welche die Datentypen festlegen (mit oder ohne Nachkommastellen, numerische oder alphanumerische Daten, Texte . . .).
4. *Arithmetische Anweisungen.*
   Hier finden die eigentlichen Berechnungen statt.

Der Nachteil von Fortran ist hauptsächlich in der etwas schwierigen Handhabung größerer Datenmengen zu sehen, wie sie beispielsweise in kaufmännischen Anwendungsbereichen anfallen.
Fortran wurde 1957 eingeführt und ständig überarbeitet. Die heute gebräuchlichste Form ist Fortran IV mit den ANSI-Erweiterungen.
(ANSI = American National Standards Institute, amerikanischer Normenausschuß).

Tabelle 10:

```
Beispiel eines Anweisungsfehlers in Fortran

SUMME = K1 + K2 + - K3              Programmzeile
E+E+E+E+E+E+E+E+E+$+E+E             Fehlermeldung

E = Error und $ = Stelle des Fehlers / Hier ist daß Minuszeichen der Fehler.
```

Tabelle 10b:

```
Programmbeispiel zu Fortran
Listing

   REAL*8 XA,XE,D,E,XL,ZL,ZR,X1,X*,X,Y
   F(X) = 0.1 * X**3 + 0.5 * X**2 - X - 3.5
   READ (9,1) XA,XE,D,E
 1 FORMAT ($ F10.3)
   XL = XA
   XR = XL + D
   YL = F(XL)
20 YR = F(XR)
   WRITE (10.2) XL,ZL
 2 FORMAT (1H,2X,2 F10.3)
   IF (YL*YR) 10,10,30
10 X1 = XL
   X2 = XR
40 X = 0.5 * (XL + XR)
   Y = F(X)
   IF (ABS(Y) -E) 79,50,50
50 IF (Y*YL) 60,80,80
80 XL = X
   YL = Y
   GO TO 40
60 XR = X
   YR = Y
   GO TO 40
70 WRITE (10,2) X,Y
   XR = X2
   YR = F(XR)
30 XL = XR
   YL = XL + D
   IF (XL-XE) 20,90,90
90 WRITE (10,2) XL,YL
   STOP
   END
```

Zwar ist die Sprache Fortran wie viele andere Sprachen genormt (s.o.), aber im Laufe der Zeit wurden beim Erstellen der Compiler verschiedene Computerhersteller einige Anweisungen modifiziert oder anlagenspezifische Befehle mit aufgenommen, so daß man heute nicht mehr von der Sprache Fortran sprechen kann. Es existieren viele Versionen, man spricht auch von Dialekten.

Normen

- DIN 66.027 „Informationsverarbeitung; Programmiersprache Fortran" (BRD) in deutsch;
- ANSI X3.9.1978 (USA), (früher USAS X3.9-1966);
- ISO 1539–1980 „Programming Language Fortran" (international).

In neuester Zeit ist versucht worden, die Vorteile der verschiedenen Versionen in einem Komplex zusammenzufassen: Fortran 77.
Jedoch wird diese Version auch nur eine von verschiedenen Fortran-Dialekten bleiben, da sie zu spät auf den Markt kommt. Die bestehenden Versionen lassen sich kaum verdrängen und viele Studenten, die heute die Grundlagen des Programmierens erlernen sollen, studieren dies mit einer alten Fortran-Fassung. Hinzu kommt, daß Anlagen, die beispielsweise mit Fortran IV-Compilern laufen, sich in den meisten Fällen zwar auf Fortran 77 umstellen lassen, dies aber mit hohen Kosten verbunden ist.
Folgende **Befehle** gehören zum Fortran-Vokabular:

Steueranweisungen:
GO TO
IF
DO
CONTINUE
CALL
RETURN
PAUSE
STOP
END
DIMENSION
E/A-Anweisungen:
READ
WRITE
ACCEPT
DISPLAY
ENCODE
DECODE
FORMAT
REWIND
BACKSPACE
END FILE
DEFINE FILE
Operatoren:
\+ Addition
– Subtraktion
* Multiplikation
/ Division
**Potenzierung
Bibliotheks-Unterprogramme:

SQRT	Wurzelfunktion
SIN	Sinusfunktion
COS	Cosinusfunktion
ATAN	Arcustanges
EXP	Exponentialfunktion
ALOG	Logarithmusfunktion.

Variablen:
In Fortran wird streng unterschieden zwischen den sogenannten Integer-Variablen (ganze Zahlen) und den Real-Variablen (Zahlen mit Nachkommastellen).
Schon bei der Programmierung muß festgelegt werden, welche Variablen und Konstanten zum Typ Integer oder zum Typ Real gehören.

Es gibt zwei Möglichkeiten, die Variablen (und Konstanten) zu definieren:

1. *Implizit:*
   Beginnt ein Variablenname oder Konstantenname (Buchstaben A bis Z und Zahlen 0 bis 9) mit einem der folgenden Buchstaben:
   I J K L M N
   so gehört er automatisch zum Typ Integer, wenn keine anderen Vereinbarungen getroffen werden. Alle anderen Namen gehören dem Typ Real an.
   Beispiel:

SUM	= 5.11	Real
P	= 6.9	
X	= 1.2	
PI	= 3.14	
I	= 2	Integer
JA	= 1	
NUMMER	= 123	
NN	= 8.	

2. *Explizit:*
   Am Programmanfang werden die verwendeten Variablen oder Konstanten definiert. Für allen übrigen Variablen und Konstanten gilt die implizite Typ-Vereinbarung.
   Beispiel:
   ■ Anweisung:
   INTEGER SUMME, X, Y, NN
   REAL MM, Z, KA.
   ■ Ganzzahl-Variable:
   SUMME, X, Y, NN
   und alle übrigen Variablen, deren Name mit I J K L M oder N beginnt.
   ■ Gebrochene Variable:
   MM, Z, K und alle anderen Variablen mit Ausnahme der oben genannten.

In Fortran werden häufig verwendete Programmteile in sogenannte Unterprogramme geschrieben, die über das Hauptprogramm aufgerufen werden können. Es gibt hier verschiedene Unterprogrammformen, die je nach den Erfordernissen Einsatz finden.
Die sogenannten Bibliotheksunterprogramme (siehe oben) gehören standardmäßig zum Fortran-Vokabular und werden einfach mit ihrem Namen und dem Parameter aufgerufen, zum Beispiel wird der Sinus von 15.5° berechnet mit dem UP-Aufruf:

```
x = sin (15.5)
```

Der Sinuswert von 15.5 wird daraufhin berechnet und in der Variablen X abgespeichert und kann dort abgerufen werden.
Die Möglichkeit der Unterprogrammbildung gibt es auch in anderen Programmiersprachen, jedoch ist bei Fortran zu beachten, daß Variablen im Hauptprogramm nicht den gleichen Wert enthalten wie im Unterprogramm, es sei denn, es wird ausdrücklich gewünscht (Parameterübergabe). Anders ist dies beispielsweise in BASIC, wo eine einmal belegte Variable ihren Wert beibehält (auch in Unterprogrammen), bis sie durch eine Anweisung geändert wird.
Sogar nach der Beendigung des BASIC-Programmes können die Variablen noch abgefragt werden, was in Fortran nicht mehr möglich ist.

Programmbeispiel:
Nullstellenermittlung

Die Nullstellen einer vorgegebenen Funktion sollen innerhalb eines gewählten Intervalles berechnet werden. Die Funktion wird im Programm definiert mit:

```
f(X) = 0,1 * X**3 + 0,5
       * X**2 – X – 3,5
```

Die Grenzen, in denen die Funktion untersucht werden soll, werden auf einer Datenkarte angegeben, ebenso die Schrittweite, denn es soll außerdem eine Wertetabelle ausgedruckt werden.

XA Untere Grenze
= – 8,00
XE Obere Grenze
= + 5,00
D Schrittweite
= 1,00
E Genauigkeit
= 0,001.

Programmbeispiel siehe Tabelle 10b.

Der Drucker hat hier im Beispiel die Geräte Nr. 10, der Kartenleser die Nr. 9 (READ und WRITE-Anweisungen). Bei der vorgegebenen Funktion und den gegebenen Grenzen sowie der Schrittweite erhält man folgende Ausgabe auf dem Drucker:

–8.000	–14.700	
–7.000	–6.300	
–6.000	–1.100	
–5.675	–0.000	– –
–5.000	1.499	erste Nullstelle
–4.000	2.100	
–3.000	1.300	
–2.168	0.000	– –
–2.000	–0.299	zweite Nullstelle
–1.000	–2.099	
–0.000	–3.499	
0.999	–3.900	
2.843	0.000	– –
2.999	0.699	dritte Nullstelle
3.999	6.899	
4.999	16.499	

Im Rahmen der hier vorgegebenen Rechengenauigkeit ist Zahl 0.999 = 1.000, die 1.999 = 2.000 und so weiter.
Das negative Vorzeichen vor der 0.000 hat für die Auswertung keine Bedeutung.
Die Nullstellen dieser Funktion liegen demnach bei:

```
x1  = –5.675
x2  = –2.168
x3  = +2.843
```

Ob alle Nullstellen der Funktion getroffen werden, hängt bei diesem Programm im wesentlichen davon ab, welches Intervall gewählt wird. Wird die Genauigkeit erhöht, verlangsamt sich die Rechengeschwindigkeit.

33. GPL
Graphical Programming Language.
Spezialentwicklung für die Programmierung grafischer Probleme.

34. HOL
System-Implementationssprache für Prozeßrechner des US-Verteidigungsministeriums, die sich noch in der Entwicklung befindet.
HOL ist maschinenunabhängig und dürfte aufgrund dessen von großer Bedeutung sein, da eine einheitliche Systemimplementierung bisher noch nicht möglich war. Man hofft, die mit Hilfe von HOL erreichen zu können.

35. HOPE
Deklarative Sprache.
HOPE wurde entwickelt an der University of Edinburgh. Über ein sogenanntes Regelwerk werden Beziehungen zwischen den betrachteten Daten sowie der Datenfluß an sich beschrieben (siehe auch LISP oder PROLOG).

36. Lisp
List Processing Language.
Höhere Programmiersprache mit freier Operatorwahl und Meta-Compilation.
Lisp hat einen compilierenden Interpreter (siehe auch Forth).
Lisp ist eine Programmiersprache, die zur Erforschung der künstlichen

Intelligenz eingesetzt wird, das heißt, in Lisp werden unter anderem Lernprogramme für Maschinen und Systeme geschrieben. Die Programme ändern sich je nach den Anforderungen, sie können „vergessen", was nicht benötigt wird, um Speicherplatz für das „Dazugelernte" zu schaffen.
Lisp ermöglicht hochpräzise Rechnungen in jedem Zahlensystem mit einer Basis aus dem Bereich von 2 bis 36. Durch eine dynamische Speicherverwaltung wird der benutzte Speicherplatz optimiert, so daß auch extrem komplexe Fragestellungen gelöst werden können. Die erste Implementierung von Lisp auf einen Microcomputer war Lisp 80 von der Firma Microsoft.
Lisp gibt es bereits seit Ende der 50er Jahre, wurde von John McCarthy am Massachusetts Institute of Technologie geschaffen. Sie gehört seit Anfang 1970 zu den grundlegenden Werkzeugen für die Erforschung der künstlichen Intelligenz.

KÜNSTLICHE INTELLIGENZ

Lisp unterscheidet sich von anderen höheren Programmiersprachen in erster Linie darin, daß numerische Operationen nicht die Basis für die Programmausführung sind.
Lisp ist daher besonders prädestiniert für die Manipulation symbolischer Daten.
Stellt man sich Symbole als Worte vor, dann sind Listen von Symbolen nichts anderes als Sätze und Aussagen. Da die Symbolverarbeitung und die Fähigkeit zur Wissensdarstellung relativ einfach zur Repräsentation von Schablonen und zu Assoziationen des menschlichen Denkens genutzt werden können, wurde Lisp zu einem wesentlichen Werkzeug bei den

```
Programmbeispiel zu Lisp
Listing Ausschnitt
(defun exchange-minus(expression)
(selectq(expression-type expresssion)
(expression-with-1-arg)
(undefinded 'undefinded)
(0 0)     ;(-0) -0
(otherwise(let((first-arg(exchange(operrand1 expression))))
(if(equal(operrand1 expression)first arg)
expression
(exchange-minus(make-expression-with-1-arg)
operator 'minus
operand1 first-arg))))))
(expression-with2-args
(if(or(eq (operand1 expression) 'undefinded)
'undefinded
(let((first-arg (operand1 expression)))
(if equal first-arg 0)
(exchange (make-expression-with-1-erg)
operator 'minus
operand1 (operand2 expression)))
(if(equal first arg (operand2 expression))
(selectq (operand2 expression)
(0(exchange (operand1 expression)))
.
.
.
```

Tabelle 11:

Versuchen, menschliches Verhalten und Denken zu simulieren. Darüber hinaus ist Lisp allerdings auch eine Sprache für andere universelle Anwendungen.
Wenn auch die Programmierung in Lisp nicht gerade als einfach bezeichnet werden kann, so stehen dem Programmierer doch eine Vielzahl nützlicher Hilfs-Werkzeuge zur Verfügung; so sind compilierter und interpretierter Code jederzeit ausführbar, auch Teilprogramme können ausgeführt und zur Fehlersuche ausgetestet werden.
Die Konzeption von Lisp ist, wie der Name schon sagt, auf die Verarbeitung von Listen abgestellt, aber auf eine ganz andere Art wie beispielsweise bei den Listen-Generatoren (siehe RPG oder LPG).
Der Programmierer kann hier darauf verzichten, eine Programmstruktur entwickeln zu müssen.
Diese ist Bestandteil der Programmiersprache, und man kann sich bei der Programmierung ganz auf die logische Seite des Problems konzentrieren.
Siehe hierzu auch die Ausführungen zu PROLOG.
Aus Frankreich kommt die dort sehr stark verbreitete Version Le-Lisp dieser schnellen KI-Sprache. Le-Lisp gibt es seit 1986 auch in Deutschland.

Programmbeispiel siehe Tabelle 11.

37. Logo

Logo ist ein weiterentwickelter Dialekt der Programmiersprache Lisp, jedoch wesentlich weiter verbreitet als diese. Logo wurde 1967/68 von Seymour Papert für Kinder und Anfänger konzipiert.
In Europa hatte Logo einige Startschwierigkeiten, da es nur auf Großcomputern sinnvoll angewandt werden konnte, denn für diese wurde es entwickelt. Doch Kinder haben eben selten Zugang zu Großcomputern.
In den USA hatte Logo jedoch große Erfolge in einem Großversuch in den Jahren 1970/80.
Hier in Europa ist Logo erst seit etwa 1980 bekannt. Dennoch kann sich diese hervorragende Programmiersprache kaum gegen die verbreiteten konventionellen Sprachen durchsetzen, obwohl es für die meisten Microcomputersysteme Implementierungen von Logo gibt, sogar für einige Homecomputer.
Logo ist eine typische Interpretersprache wie BASIC. Besonders unterstützt werden grafische Anwendungen und vor allem die Computerperipherie bis hin zum Joystick.

GRAFISCHE ANWENDUNGEN

Logo ist leicht lesbar und kurz. Die Programme werden in Prozeduren aufgeteilt (modulare Blöcke) und sind top-down strukturiert, das ist ein entscheidender Vorteil gegenüber BASIC.
Die Nachteile von Logo liegen eindeutig in seinen Schwächen beim interaktiven Rechnen.
Mathematisch-naturwissenschaftliche Probleme bleiben Probleme in Logo.
Logo stellt folgende Grundoperationen und Funktionen zur Verfügung:

+ Addition
– Subtraktion
* Multiplikation
/ Division.

Negative Zahlen sind in Klammern zu schreiben, ansonsten gilt „Punktrechnung vor Strichrechnung".

Ganzzahloperationen:
ROUND :X

rundet die reelle Zahl X auf die nächste ganze Zahl;

INTEGER :X
schneidet von der reellen Zahl X die Nachkommastellen ab;

QUOTIENT :X :Y
ergibt den ganzzahligen Anteil der Division von X durch Y an.

Funktionen:
SQRT :X
Quadratwurzel aus der Zahl X;

SIN :W
Sinuswert des Winkels W (in Grad);

COS :W
Cosinuswert;

ATAN :X
Umkehrfunktion des Tangens;

RANDOMIZE
Initialisierung des Zufallszahlengenerators;

RANDOM :N
erzeugt eine ganze Zufallszahl zwischen 0 und N.

In Logo sind keine Variablen-Deklarationen notwendig. Logo kennt globale und lokale Variable. Eine Variable kann einen Zahlenwert, einen Booleschen Wert, ein Wort, einen Satz beliebiger Länge, eine Liste von Elementen der verschiedensten Typen und sogar Prozedurnamen enthalten. Gerade Programmieranfängern dürfte dies sehr entgegen kommen. Das Handling von Listen, eine höhere Form von Arrays, ist das geeignete Mittel für komplexere Probleme der künstlichen Intelligenz – mit Logo sogar schon auf Homecomputern (in gewissen Grenzen, versteht sich).
Logo nimmt unter den höheren Programmiersprachen eine Sonderstellung ein. Logo wurde nicht für bestimmte Anwendungen entwickelt, sondern als Trainings-Sprache zum Erlernen des Programmierens in Form einer geistigen Disziplin;

```
Programmbeispiel zu Logo
Listing
----------------------------------------------------------------
TO PRIMZAHLEN
     CLEARTEXT
     PRINT (PRIMZAHLENTABELLE) PRINT()
     PRINT (UNTERE GRENZE?) MAKE "M FIRST
     REQUEST
     PRINT (OBERE GREZE?) MAKE "N FIRST
     REQUEST
     IF (REMAINDER :M2) = 0 THEN MAKE "M :M+1
     PRINT()
END

TO PRIM :M :N
     IF :M ) :N THEN PRINT () STOP
     MAKE "WURZEL SQRT :M
     MAKE "TEILER 3
     PRIMTEST :M :WURZEL :Teiler
     PRIM :M + 2 :N
END

TO PRIMTEST :M : WURZEL : TEILER
     IF : TEILER² :WURZEL + 1 THEN PRINT1 : M PRINT (,) STOP
     IF REMAINDER :M :TEILER) = 0 STOP
     PRIMTEST :M : WURZEL : TEILER + 2
END
```

Tabelle 12:

also zum Schulen von bestimmten Denkmechanismen, die für eine moderne Programmierung in jeder Programmiersprache von Vorteil sind.
Logo sollte daher nach den Wünschen ihres Entwicklers die allererste Sprache sein, mit der ein Anfänger in die Welt der Programmierung einsteigt, um später eine andere zu lernen.
Genau dieses Konzept wurde versuchsweise an amerikanischen, englischen und französischen Schulen eingeführt, mit großem Erfolg.
Es gibt einige Vorteile von Logo, die es sinnvoll machen, den oben beschriebenen Weg einzuschlagen. Im folgenden werden die wichtigsten aufgezählt:

- In Logo sind die Prozeduren unabhängig voneinander aufrufbar und müssen nicht über ein Hauptprogramm miteinander verknüpft sein. Ein Logo-Programm kann also direkt am Bildschirm entworfen werden.
- Die kurzen Module können während der Eingabe fortlaufend überprüft und festgehalten werden. Problemlos lassen sie sich korrigieren und verfeinern. Bei Aufruf des Programms mit dessen Hauptnamen werden die Module automatisch verkettet.
- Logo eignet sich ähnlich wie BASIC für den Dialogbetrieb, da es nicht compiliert, sondern interpretiert wird, was andererseits zu Lasten der Geschwindigkeit geht. Dies ist jedoch für eine „Lernsprache" untergeordnet.
- Logo unterstützt Rekursionen. Schleifen werden nicht iterativ, sondern rekursiv dargestellt.
- Logo stellt dem Anwender standardmäßig eine hochauflösende Grafik zur Verfügung, was viele andere Sprachen nicht können.
- In den meisten Programmiersprachen ist die Syntax und deren Umfang genau vorgegeben. Auch in Logo gibt es natürlich einen festgelegten Grundwortschatz, jedoch kann der Anwender beliebig neue Grundwörter definieren, ähnlich wie in Forth (siehe dort). Dadurch kann der Sprachschatz erweitert und die Programmierumgebung den spezifischen Anwenderanforderungen angepaßt werden.

Den oben genannten Vorteilen stehen aber auch eine Reihe von Nachteilen gegenüber:

- Logo kennt keine indizierten Variablen, so daß nicht wie in anderen Sprachen bequem auf bestimmte Elemente eines Feldes zugegriffen werden kann. Logo stellt hierzu die rekursive Listenbearbeitung bereit.
- Es stehen kaum vernünftige Ausgabe-Formatierungs-Befehle zur Verfügung, so daß es in Logo schwierig werden kann, Listen und Tabellen übersichtlich auf dem Bildschirm darzustellen. Es fehlen auch einige, je nach Anwendung, wichtige mathematische Funktionen, beispielsweise zum Potenzieren, Tangensfunktionen, Umkehrfunktionen zu Sinus und Cosinus sowie logarithmische Funktionen und Exponentialfunktionen. Ohne diese sind allgemeine mathematisch-naturwissenschaftliche Programme kaum zu erstellen. Daher gibt es einige Ergänzungen zu Logo, aber nicht für jeden Rechner. Logo ist eben nicht die für diesen Bereich prädestinierte Programmiersprache.

Vor- und Nachteile dürften sich weitgehend auf-

heben, so daß wieder die oben angesprochenen Gründe für Logo als „Einsteiger-Sprache" zählen.

Programmbeispiel siehe Tabelle 12.

38. LPG

Listen Programm Generator (siehe auch RPG). Ein Programm-Generator ist ein System, welches von Dateien eingelesene Datensätze nach vorbestimmten, gleichartigen Umformungen bearbeitet und in Form einer Liste ausgibt. Haupteinsatzgebiet für solche Programmiersprachen liegen im kommerziellen Bereich.

39. MACRO 80

MACRO 80 ist ein verschiebbarer Assembler (siehe dort) für den Prozessor 8080, 8085 und Z80. Dieser Assembler unterstützt alle Macros nach dem Intel-Standard. Dabei wird die Schachtelung der Macros nur durch den vorhandenen Speicherplatz begrenzt.
Der Objektcode wird in verschiebbarer Form generiert, die durch den Lader weiterverarbeitet wird. Eine bedingte Assemblierung wird ermöglicht durch Pseudo-Operationen. Bedingungen können bis zu 255 Stufen geschachtelt werden.
MACRO-Definitionen erlauben es, mehrere Befehlszeilen als Macro in einer Bibliothek abzuspeichern und jeweils durch einen Aufruf in das Quellprogramm einzuschließen. Besonders interessant ist auch die Möglichkeit, in der Bibliothek Macros abzuspeichern, die Maschinencode für jede beliebige Acht- oder 16-Bit-Maschine unabhängig vom 8080-Code generieren können.
MACRO-80 benötigt rund 14 KByte Speicherplatz und es werden etwa 1000 Zeilen pro Minute assembliert.

```
-----------------------------------------------
Programmbeispiel zu Modula
Listing
-----------------------------------------------
  FROM TERM1 IMPORT
     WriteString, WriteCard, Writeln;

  CONST
     Size       = 7244;          (* Arraygröße *)
     Iterations = 15;            (* minimum 1  *)

  VAR
     count, i, iter, k, prime:  CARDINAL;
     flags: ARRAY(0..Size-1) OF BOOLEAN,

BEGIN
  WriteString('15 Iterations'); Writeln
  FOR iter :=1 TO Iterations DO
     count := 0;
     FOR i := 0 TO Size DO flags(i) := TRUE; END;
     FOR i := 0 TO Size DO
        IF flags(i) THEN
           prime := i * 2 + 3;
           k := i + prime;
           WHILE k = Size DO
              flags(k) := FALSE,
              INC(k, prime);
           END;
           INC(count);
        END;
     END; (* FOR *)
  END; (* FOR *)
  WriteString(' There were ');
  WriteCard(count, 0), WriteString(' primes ');
End BEISPIEL
-----------------------------------------------
```

Tabelle 13:

40. MANTIS

MANTIS ist eine höhere interpretative, prozedurale Sprache, welche die Vorteile von BASIC (leicht erlernbar, interpretativ) und Pascal (Strukturierung und String-Manipulation) in sich vereint. Hinzu kommen Elemente aus APL: Operatoren und Mächtigkeit.
Aus diesen Gründen ist MANTIS sehr flexibel.
MANTIS wurde entwikkelt von „Cincom Systems" zur Produktivitätssteigerung beim Programmieren. Es wird daher oft mit den Tools verglichen, die mit Sprachen wie Cobol zusammenarbeiten. MANTIS ist jedoch eine richtige höhere Programmiersprache.
Integrierte Help-Funktionen und Menüsteuerungen machen MANTIS sehr benutzerfreundlich.

Der vollständige Befehlsvorrat ermöglicht eine übergreifende Anwendungsbereich-Abdeckung.
Die Stärke von MANTIS liegt in seiner Effizienz.
Leistungsfähige Gleitkomma-Arithmetik und ein spezieller Interpreter, der mit speziellen Techniken arbeitet und sich deutlich von anderen Interpretern abzeichnet, sind die Vorteile dieser relativ neuen Programmiersprache.
Innerhalb kurzer Zeit gab es von MANTIS Implementationen für die gängigsten Rechnertypen.
Von IBM-Rechnern unter DOS über VM-CMS und MVS gibt es MANTIS nun auch für die VAX von DEC, WANG/VS und Honeywell-Bull.
MANTIS-Programme sind sehr flexibel und weisen eine erstaunlich hohe Portabilität auf.
Sie sind völlig unabhängig vom Betriebssystem.
MANTIS vereinigt in sich alle fortschrittlichen Technologien: vom interaktiven Bildschirmentwurf über die Darstellungsform des sogenannten „Filmablaufes" bis hin zur vollständigen Simulation von Anwendungen, einschließlich der Darstellung aller Datenflüsse und der programmierten Entscheidungslogik. Die dabei angewandte Verbindung zwischen Top-down- und Bottom-up-Lösungsansatz führt zu ausgezeichneten Programmen mit relativ geringem Wartungsaufwand.
MANTIS erlaubt auch bei komplexeren Problemen schnellere Lösungen als mit herkömmlichen Programmiersprachen.

41. Modula 2

Der Name dieser höheren Programmiersprache weist schon hin auf den modularen Aufbau der Programme.
Modula 2 entstand aus einer Studie zur Programmierung gleichzeitig ablaufender Prozesse: Modula 1.
In Modula 1 haben die Module die Aufgabe sogenannter Monitore, Gruppierungen gemeinsamer Variablen und ihrer Operatoren. Monitore sind als kritische Programmabschnitte gedacht, welche nicht gleichzeitig von verschiedenen Prozeduren durchlaufen werden dürfen.

Die allgemeine Bedeutung und letztlich wichtigere für die Gliederung von Programmsystemen wurde bei der Entwicklung erst später erkannt.
Modula ist als „Nachfolger" von Pascal von Prof. Klaus Wirth am Institut für Informatik (IfI) in Zürich entwickelt worden und erhielt sehr schnell das Prädikat: „Pascal ohne die Nachteile von Pascal".
Nicht nur der Programmaufbau ist modular, sondern auch die Compilation erfolgt modulweise. Die Sprachdefinitionen sind auf ein Minimum beschränkt, dafür wird der Aufbau von Modulbibliotheken unterstützt. Dies wird erreicht durch eine gesonderte Schnittstellenbeschreibung, welche als gesondertes Modul existiert, je nach Implementation, also Installation auf einem bestimmten Rechnertyp.
Folgende Merkmale von Modula 2 unterscheiden diese wesentlich von anderen höheren Programmiersprachen:

- Namen können beliebig lang sein;
- Groß- und Kleinschreibung können unterschieden werden;
- Konstantenarithmetik;
- Reihenfolge der Deklarationen ist nicht streng vorgeschrieben;
- Open Arrays in Prozedur-Deklarationen;
- ELSE-Variante beim CASE-Statement.

Modula 2 setzt einigen Aufwand an Hardwareeinsatz voraus: mindesten 56-Kbyte-Arbeitsspeicher, zwei Floppy-Laufwerke mit je 350-KByte-Kapazität. (Die minimale Konfiguration der notwendigen Files für den Compiler, den Linker und den Editor ist bereits 226 KByte groß.)
Modula 2 ist ursprünglich für den Z80-Prozessor entworfen worden, unter dem Betriebssystem CP/M80.
Wie bei vielen anderen compilierbaren Programmiersprachen muß für Modula 2 jeweils separat

- der Texteditor – zum Erstellen und Eingeben von Programmfiles;
- der Compiler – zum Übersetzen der einzelnen Module und
- der Linker – zum Binden der einzelnen Module und Bibliotheksunterprogramme

aufgerufen und gestartet werden.
Die Bibliothek von Modula 2 besteht ihrerseits ebenfalls aus Modulen und kann vom Anwender ergänzt werden.
Das hat zur Folge, daß im Laufe der Zeit die einzelnen Modula 2-Programme immer kürzer werden können, da auf eine wachsende Anzahl von fertigen Problemlösungen in Form von Unterprogrammen zurückgegriffen werden kann.
Die Bibliotheksmodule liegen dann bereits in übersetzter Form vor und brauchen nur noch an das neu geschriebene Programm gebunden zu werden (Link File).
Der Compiler besteht aus vier Compiler-Pässen, das heißt, es sind vier Durchläufe zur Übersetzung eines Modula 2-Programmes nötig.
Eine Aufstellung der beim Übersetzen gefundenen Fehler wird in einem gesonderten File (LISTE) abgelegt und kann dort zur Fehlerbeseitigung eingesehen werden.
Hat das Programm die vier Compilerpässe fehlerfrei durchlaufen, können die einzelnen Programm-Module untereinander und natürlich auch mit den Bibliotheksmodulen gebunden werden. Hierzu sind zwei weitere Pässe notwendig, die der Linker steuert.

Programmbeispiel siehe Tabelle 13.

```
---------------------------------------
Programmbeispiel zu Modula
Assembler-Implementation
---------------------------------------
;**************************************
;*     IMPLEMENTATION MODULE Ports     *
;**************************************
;
        name ('ports')
        title ports
        .z80
;
;***** EXPORT ************************
;
inp:    pop iy  ; get return address
        pop hl  ; address of 'data' in hl
        pop bc  ; c contains 'portAdr'
        in a, (c)
        ld (hl), a
        jp (iy) ; instead of push, ret
;
outp:   pop iy  ; get return address
        pop de  ; 'data' is in e
        pop bc  ; /portAdr' is in e
        out (c), e
        jp (iy) ; instaed of push, ret
;
;**************************************
;
        end
---------------------------------------
```

Tabelle 14:

Modula 2 bietet auch ein komfortables **Assembler-Interfacing.** Hierzu sind allerdings sehr genaue Kenntnisse über die Stacks und den Aufbau der Variablen notwendig.
Modula ist eine moderne Programmiersprache, die sich besonders zur Konstruktion von großen, modularen Software-Systemen eignet. Der modulare Aufbau der Programme hat sich als sehr vorteilhaft erwiesen und ist gemeinsam mit der separaten Übersetzung der Module als Garantie für effizienten Unterhalt und dauernde Konsistenz der Software verantwortlich.
Obwohl einfach in seiner Konzeption, hat sich das Modul als ein überraschend schwierig zu handhabendes Hilfsmittel herausgestellt.
Module treten in den verschiedensten Erscheinungsformen auf, so zum Beispiel als Package von Prozeduren, als Verwaltungsinstanz von Ressourcen, als Schnittstelle zur Hardware, als „Black Box" zur Verarbeitung von Objekten als Monitor und als Datenbasis.
Das modulare Denken eröffnet neue Horizonte und führt zu weitreichenden Konsequenzen, sogar für Betriebssystem-Designer und Computer-Architekten.

Programmbeispiel siehe Tabelle 14.

42. mu-SIMP

High-Level-Programmiersprache für symbolische und halbnumerische Datenverarbeitung.
mu-SIMP benutzt einen schnellen Interpreter, der nur wenig Speicher-

platz benötigt (7 KByte). mu-SIMP ist eine Spezialentwicklung und daher kaum verbreitet: Das Hauptanwendungsgebiet ist die wissenschaftliche Mathematik.

43. Natural

Strukturierte non-prozedurale Programmiersprache mit Schnittstellen zu Cobol und PL/1. Damit können Natural-Programme von Cobol- oder PL/1-Programmen aufgerufen werden, beziehungsweise umgekehrt.
Natural ist im Vergleich zu den oben genannten eine relativ „junge" Programmiersprache, jedoch keine Weiterentwicklung von Cobol oder PL/1.
Natural ist strukturierend (zum Beispiel kein GOTO-Befehl) und unterstützt die modulare Programmierung.
Nachteile von Natural: es können nur eindimensionale Tabellen verarbeitet werden.
Erstaunlich komfortabel hingegen ist die interaktive Programmierumgebung mit Compiler, Testprogrammen, Maskengenerator, Editor, Bibliotheks- und Datenverwaltung, DATA-Dictionary und Zugriffs-Schutzmöglichkeiten.
Einsatzgebiet für die Formuliersprache Natural sind unter anderem Datenbanken.
Natural ist mehr als eine der „neuen Programmiersprachen" (sie wird zur 4. Generation gezählt).
Natural stellt eine komplette Programmierumgebung bereit. Integriert sind Compiler, Editor, Maskengenerator, Testfunktionen, sowie Bibliotheks- und Datenverwaltung, Data-Dictionary und Zugriffsschutzmöglichkeiten. Teilweise werden also schon Betriebssystem-Aufgaben übernommen. Natural unterstützt durch seine hervorragende Menütechnik das Prototyping von Programmen.
Die Handhabung von Natural ist nicht ganz unproblematisch, daher erscheint diese Programmiersprache nicht unbedingt geeignet für den unerfahrenen Programmierer. Optimal programmiert sind Natural-Applikationen jedoch extrem schnell und benutzerfreundlich.

44. NEAT

NCR Elektronik Autocoding Technique. Maschinenorientierte Spezialentwicklung der Firma NCR. NEAT ist eine NC-Sprache (Numeric Control) für die Steuerung von Werkzeugmaschinen.

45. NELIAC

Ein Algol-Dialekt (siehe unter Algol).

46. NPL

New Programming Language.
Vorläufer der Programmiersprache PL/1 (siehe dort), und damit heute nicht mehr gebräuchlich.

48. OPS-5

OPS-5 ist eine Eigenentwicklung der Firma DEC (Digital Equipment Corporation).
Sie eignet sich besonders für die Anwendung in Expertensystemen und für Bereiche der kognitiven Wissenschaften im Rahmen der künstlichen Intelligenz. Darüber hinaus unterstützt OPS-5 auch die Lösung komplexer Probleme im kommerziellen Umfeld.
Expertensysteme sind Datenbanken, welche sich aber von den konventionellen Datenbanken durch ihr „intelligentes" Verhalten unterscheiden.
Expertensysteme sollen die fachliche Kompetenz von Experten in Form von Sach- und Erfahrungswissen bereitstellen.
Zu diesem Zweck wird das Wissen nicht nur in Form von Daten abgespeichert, sondern auch durch Regeln verknüpft, klassifiziert und bewertet. Die Systeme sollen darüber hinaus heuristische Methoden und vages Wissen verarbeiten und nach den vorgegebenen Regeln aus dem vorhandenen Datenmaterial selbständig Schlüsse ziehen, das heißt Problemlösungen anbieten.
Die Systeme können so dem Anwender bei der Problemanalyse helfen, sie dokumentieren ihre Vorgehensweise.
Ein professionelles Expertensystem besteht aus fünf Abschnitten:
1. Die „Wissensbasis" ist der Grundstock. Hier ist das Experten-Wissen in Form von Fakten und Regeln abgespeichert.
2. Die Abteilung „Wissenserwerb" beinhaltet Instrumente zur Aufnahme und Integration von neuen Expertenwissen in das System, also hier in die Wissensbasis. Diese Instrumente formulieren die neuen Informationen so um, daß erkannt werden kann, ob dieses Wissen bereits im System vorhanden ist. Lücken in der Wissensbasis können erkundet werden, usw.
3. Ein weiterer Abschnitt des Systems dokumentiert dem Anwender, welcher Lösungsweg verfolgt wird und welche Ergebnisse das System ermittelt hat und mögliche Schlußfolgerungen daraus.
4. Die wichtigste Komponente des Systems ist aber die "Inference Machine", welche die Problemlösung steuert. Sie verarbeitet das Expertenwissen der Wissensbasis und zieht hieraus Schlußfolgerungen und Ableitungen. In bestimmten Situationen kann die "Inference Machine" auch beim Anwender nachfragen, also Zusatzinformation zum erfragten Problem abrufen, um eindeutigere Schlußfolgerungen ziehen zu können.
5. Die Benutzerschnittstelle „Dialog" verbindet das System schließlich mit der Außenwelt, also dem Anwender. Hier können Eingaben in das System erfolgen und hier gibt das System auch Informationen aus.
Expertensysteme haben das Ziel, sowohl objektive Fakten, als auch menschliches Erfahrungswissen zu verarbeiten.
Sie leisten also wesentlich mehr, als einfache Datenbanken, die wie große Nachschlagewerke funktionieren.
Auf diese Weise sollen Probleme gelöst werden, die aufgrund fehlender exakter Verfahren bisher nur durch erfahrungs- und bewertungsabhängige menschliche Entscheidungen gelöst werden können. Doch an dieser Stelle wieder zurück zu einer Programmiersprache, in der solche Systeme erstellt werden können.
Der OPS-5-Compiler wurde in der code-optimierenden Sprache BLISS-32 geschrieben (siehe dort). Der Anwender erhält damit einen leistungsfähigen kompakten Code.
OPS-5 ist eine typische KI-Sprache (künstliche Intelligenz), die mit Symbolen arbeitet. Sie kann eine Vielzahl von Lösungswegen erzeugen und damit für mehrere Probleme jeweils mehrere Lösungen erarbeiten.
Ein solches KI-Werkzeug bietet sich als computergestützter Lösungsweg an, wenn sich die geplante Anwendung konventionellen, auf Algorithmen beruhenden Lösungen entzieht oder der Problembereich häufigen Veränderungen unterworfen und nur schwer oder gar nicht einzugrenzen ist.
OPS-5 ist auch eine Sprache zur Entwicklung von Regelsystemen. Sie ermöglicht Problemlösungen unter Ausnutzung menschlicher Erfahrungen und menschlichen Wissens.

Fortsetzung im nächsten Heft

Interpreter oder Compiler?

Es gibt eine große Anzahl verschiedener Programmiersprachen, die jedoch alle eines gemeinsam haben: Programme, welche in einer Programmiersprache geschrieben sind (BASIC, FORTRAN, COBOL und andere) müssen zunächst übersetzt werden, bevor sie zur Ausführung gelangen können. Sie sind zwar unterschiedlich aufgebaut, aber je nachdem, auf welcher Anlage sie eingesetzt werden, in einer ganz bestimmten Programmiersprache verfaßt. Daher kann die Übersetzung immer nach gleichen Strukturen ablaufen, bleibt man bei der Betrachtung einer Programmiersprache. Diese Arbeit wiederum kann geradezu ideal von einem Computer übernommen werden.

Nicht jedes Übersetzungsverfahren eignet sich für jede Anlage oder jeden Anwendungsbereich der Programme. Die Übersetzungsverfahren sind von verschiedenen Faktoren abhängig:

- *Die Verschiedenartigkeit von Programmiersprachen prädestiniert sie für ganz bestimmte Übersetzungsverfahren. Je nach Anwendungsbereich bieten bestimmte Übersetzungsverfahren Vor- und Nachteile zum Beispiel bei der Fehlersuche und der Korrektur. Manche sind sehr gute Hilfen bei der Programmverbesserung.*
- *Die Übersetzungsart ist auch abhängig von der Kapazität und dem Aufbau des Computers. (Das Übersetzungsprogramm benötigt wertvollen Speicherplatz). Daher sind KByte-Angaben von Computeranlagen (zumindest kleinerer) immer im Zusammenhang mit den verwendeten Übersetzungsmöglichkeiten zu sehen.*

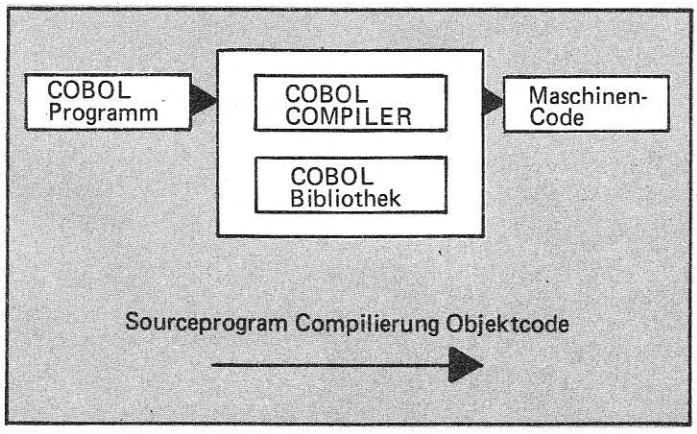
Grundsätzlich unterscheidet man zwei Übersetzungsmethoden:

- die einmalige Übersetzung durch einen Compiler
- die dauernde Übersetzung (Interpretierung) durch Interpreter.

Es gibt aber auch Mischformen hieraus.

1. DER COMPILER

Compiler sind ebenfalls Programme (in einer Programmiersprache geschrieben), mit deren Hilfe aus einem FORTRAN-Programm ein Maschinencode erzeugt wird, der vom Computer verarbeitet werden kann. Bekanntlich versteht die Maschine nur die Zustände „0" und „1".



Man unterscheidet zwei Programm-Files:

- das SOURCE PROGRAM (Quellprogramm), welches vom Programmierer in einer höheren Programmiersprache geschrieben wird. Dieses Programm ist „lesbar" für den Programmierer und bei einer strukturierten, klaren Programmierung auch von anderen Personen nachvollziehbar.
- das OBJECT PROGRAM oder den OBJECT CODE, welcher aus einer vom Benutzer nicht mehr zu interpretierenden Folge von Zeichen besteht, welche vom Computer verarbeitet werden können.

Mit SOURCE und OBJECT bezeichnet man eigentlich also das gleiche Programm in zwei verschiedenen Darstellungsformen: einmal in der Programmiersprache für den Anwender, zum anderen in einem maschineninternen Code, den der Computer versteht.

Aus diesem Grund gibt es nicht den Compiler schlechthin, sondern jeder Computertyp, jede Programmiersprache hat den spezifischen Compiler:

FORTRAN-Compiler für Computertyp A
FORTRAN-Compiler für Computertyp B
FORTRAN-77-Compiler für Computertyp A
COBOL-Compiler für Computerty A
usw.

Wenn also die Rede vom Compiler ist, so ist immer ein bestimmter Compiler gemeint. Daher nennt man den Compiler immer im Zusammenhang mit seiner Programmiersrpache, die er „versteht". Gute Compiler beschränken sich nicht ausschließlich auf die reine Übersetzung von Quellprogrammen in Objektcode, sondern bieten Hilfen etwa bei der Fehlersuche.

Hierin sind unter anderem Stärken und Schwächen von Compilern zu sehen. Es ist eben ein gewaltiger Unterschied, ob ein Compiler meldet, daß ein Fehler im Quellprogramm vorliegt, oder ob er die Art und die Lokalisation des Fehlers gleich mitliefert.

Ein weiteres Bewertungskriterium für die Leistungsfähigkeit von Compilern ist die Compilergeschwindigkeit

und die Qualität des Objectcodes, denn unterschiedliche Compiler (von verschiedenen Herstellern) liefern bei gleichem Sourceprogramm oft unterschiedlichen Code, der unterschiedlich schnell ablaufen kann. Während die Compilergeschwindigkeit in erster Linie den Programmierer interessiert, dürfte die Ausführungsgeschwindigkeit des des Codes von allgemeinem Interesse sein.

Man unterscheidet weiterhin zwei Grundtypen von Compilern: den Einpaß-Compiler und den Mehrpaß-Compiler.

1.1. Einpaß-Compiler

Dieser Compiler wandelt jedes Sprachelement des Quellprogrammes in mehrere Elemente des Maschinencodes um. Eine Umkehrung dieses Vorganges ist also kaum noch möglich.

Die Aufteilung eines Quellprogramm-Befehls in mehrere Codes ist notwendig, da die Befehle höherer Programmiersprachen sogenannte Makrobefehle sind, das heißt, sie enthalten ganze Befehlssequenzen für die Maschine. Ein Programmierer, der Programme in BASIC schreibt, muß nichts über diese Befehlsfolgen wissen, die der Rechner an das Diskettenlaufwerk gibt, um den Lesekopf auf der Diskette hin und her zu bewegen.

Der Befehl READ bewirkt also nicht ausschließlich das Lesen von Daten von der Diskette, sondern zugleich das Suchen und Kontrollieren auf Plausibilität sowie die Bewegungen des Lesekopfes. Ein Teil dieser Aufgaben wird natürlich vom Betriebssystem und von Kontrollern übernommen, aber eben der Befehl READ löst alle diese Aktivitäten aus und liefert hierzu auch die notwendigen Informationen. Diese Vorgänge werden dem normalen Anwender – wenn überhaupt – nur bei auftretenden Fehlfunktionen bewußt.

Die Aufschlüsselung beim Übersetzen hat natürlich den Nachteil, daß eine Übersetzung nur in einer Richtung stattfinden kann, da einzelne Maschinenbefehle abhängig sind von ihrer Umgebung im Programm, die Anweisungen der höheren Programmiersprache jedoch nicht, zumindest nicht unmittelbar.

Der Vorteil des Einpaß-Compilers ist in der Übertragbarkeit der Quellprogramme zu sehen. Diese Programme werden nicht maschinenorientiert, sondern können problemorientiert geschrieben werden. Dieser Vorteil gilt für alle Compiler- und Interpretertypen im Ggensatz etwa zur Assembler-Programmierung.

Aus der starken Problemorientierung der Programme (Vorteil bei der Lösung von Problemen) folgt zwangsläufig eine Maschinenorientierung des Compilers, denn an irgendeinem Punkt im System muß zwangsläufig der große Sprung stattfinden in der Reihe: Mensch-Sprache-Compiler-Maschine.

Das Quellprogramm ist nur auf das zu programmierende Problem fixiert. Die Maschinenverständlichkeit (durch Compilierung) erfolgt ausschließlich durch den Compiler des entsprechenden Gerätes. Der Programmierer muß sich also bei der Formulierung des Problems zunächst nicht mit dem Computer auseinandersetzen, er kann sich auf einer höheren Hardware-Ebene bewegen.

Da das Programm auf verschiedenen Anlagen laufen kann, spricht man von *portabler Software*.

Nachteil der Einpass-Compilierung ist jedoch, daß eine Codeoptimierung nicht stattfinden kann.

1.2. Mehrpaß-Compiler

Hier wird die Compilierung eines Programmes in mehrere Schritte aufgeteilt: das Programm in einer Programmiersprache wird zunächst in eine Zwischensprache übersetzt, bevor es ein zweites Mal übersetzt wird, diesmal in Maschinen-Code.

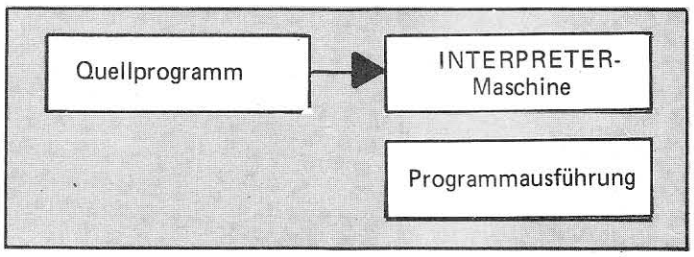
Die erste Übersetzung in die sogenannte Intermediate Language (Zwischensprache) ist maschinenunabhängig. Erst darauf folgt die maschinennahe Übersetzung der Zwischensprache.

Der Mehrpaß-Compiler hat den Vorteil, daß auch Teile des Compilers maschinenunabhängig sind (der erste Teil) und durch die zweimalige Übersetzung ein hohes Maß an Code-Optimierung erreicht werden kann.

Nachteile sind in dem größeren Aufwand zu sehen, der getrieben werden muß, sowohl in zeitlicher Hinsicht als auch in bezug auf den größeren Speicherplatz, der benötigt wird.

Die Codeoptimierung hat zur Folge, daß die Ausführungszeiten der Programme minimiert werden können.

2. DER INTERPRETER



Im Gegensatz zum Compiler wird hier nicht das gesamte Quellprogramm auf einmal in Maschinensprache übersetzt, sondern immer nur die Programmteile, die gerade durchlaufen werden.

Der Interpreter ist also ständig gleichzeitig mit dem ablaufenden Programm im Computer am arbeiten (wie ein Simultan-Dolmetscher). Beim Ablauf des Programmes werden die einzelnen Programmanweisungen in eine Zwischensprache übersetzt (entspricht dem ersten Teil eines Mehrpaß-Compilers) und diese Zwischensprache sofort für die Maschine interpretiert.

Es existiert also nie ein kompletter Objekt-Code des Programmes, im Gegenteil zum Compiler.

Da nie eine komplette Übersetzung des Quellprogrammes erfolgt, muß nicht erst ein der Programmausführung vorausgehender zeitaufwendiger Prozeß stattfinden wie bei der Compilierung; das Programm kann sofort gestartet werden. Hierunter leidet jedoch die Ausführungsgeschwindigkeit des Programmes, da beispielsweise bei zu durchlaufenden Schleifen diese bei jedem Durchlauf neu interpretiert werden müssen.

Auch ist die Ausnutzung der Kapazität der Anlage schlechter als bei Compilern. Der Interpreter muß ständig neben dem Programm im Speicher vorhanden sein, im Gegensatz zu dem Compiler, der nur für die Dauer der Übersetzung Speicherplatz belegt, da er später für die Programmausführung nicht mehr benötigt wird.

Als Vorteile bei der Arbeit mit einem Interpreter wären hier die besseren Korrekturmöglichkeiten zu nennen, da nicht nach jeder Fehlerbeseitigung im Quellprogramm eine erneute komplette Übersetzung erfolgen muß.

Auch hier ist die Portabilität der Programme gegeben, allerdings in gewissen Grenzen. Im Gegensatz zum Einpaß-Compiler hat ein Interpreter selbst eine recht gute Portabilität.

Bekannteste Beispiele guter Interpreter sind die in den Commodore-Homecomputern installierten. Hier wie auch bei vielen Rechnern dieser Klasse ist das Interpreterprogramm fest in Chips der Maschine integriert (ROM).

3. ASSEMBLER

Mit Assembler wird sowohl die Sprache als auch der dazugehörige Übersetzer bezeichnet.

Die Befehle in Assembler entsprechen genau den Maschinenbefehlen, sie sind nur der Lesbarkeit wegen als mnemotechnische Befehle verschlüsselt.

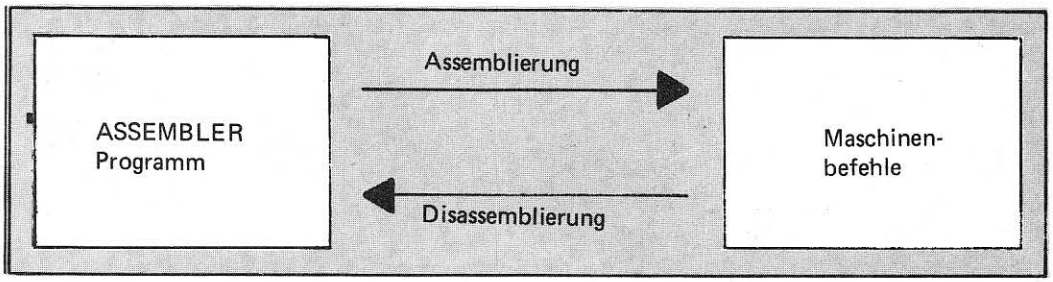
Der Assembler (hier Übersetzungsprogramm) überträgt lediglich jedem mnemotechnischen Befehl den entsprechenden Maschinenbefehl.

Vorteile des Assemblers werden in der Grafik sichtbar: Die Übersetzung ist in beiden Richtungen möglich, da jeder Assemblerbefehl genau einem Maschinenbefehl entspricht.

Diese Tatsache wirkt sich günstig aus, sowohl auf die Ausführungsgeschwindigkeit der Programme (sie ist wesentlich kürzer als bei anderen Übersetzungsverfahren) als auch auf den benötigten Speicherplatz.

Die Sprache Assembler ist allerdings streng maschinenorientiert und daher nicht portabel, was als großer Nachteil anzusehen ist. Außerdem ist sie durch fehlende Problemnähe wesentlich schwieriger zu handhaben. Assembler gehört aus diesen Gründen auch nicht zu der Familie der Hochsprachen (BASIC, FORTRAN und andere).

Dipl.-Ing. (FH) Oliver Rosenbaum ■



Nachtrag zu CW 3/88

Schalten mit dem Kassetten-Port

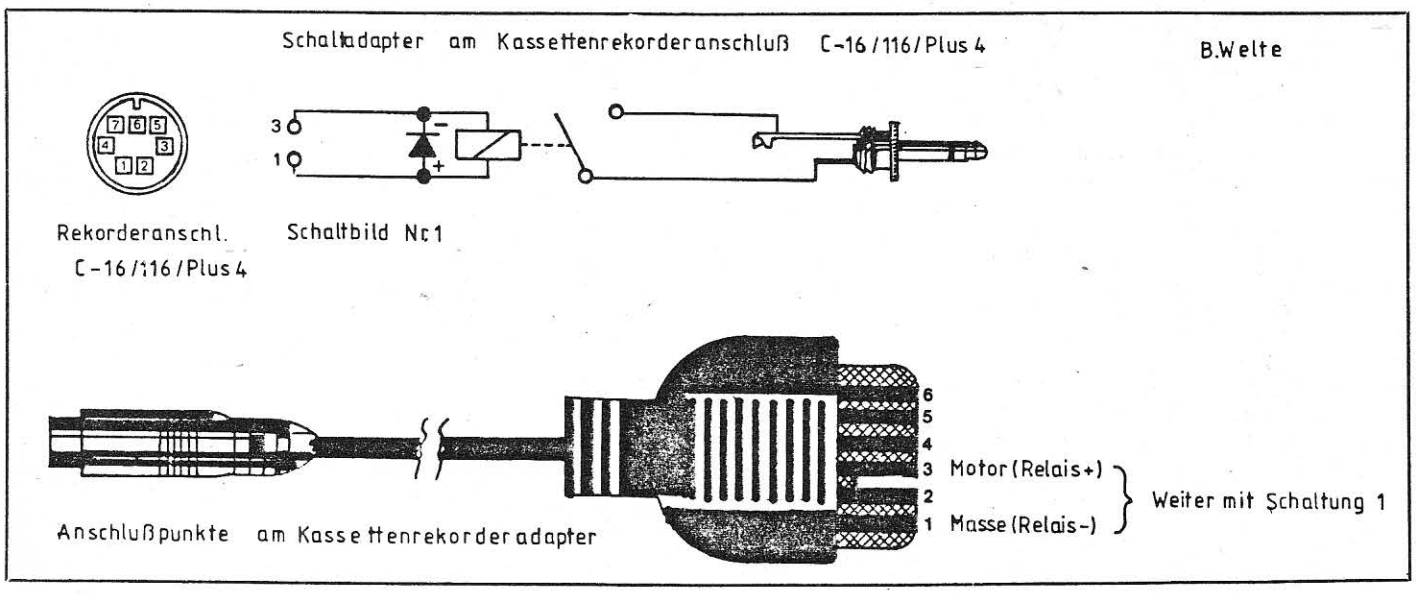
Die Schreibleitung, Pin fünf am Kassettenport, ermöglicht uns, eine Spannung von null oder drei Volt auszugeben. Die Leseleitung, Pin vier, können wir zum Messen von Signalen verwenden. Drei Volt an der Schreibleitung liegen an, wenn Bit eins in Adresse eins auf null gesetzt wird. Besitzt besagtes Bit den Wert eins, so haben wir null Volt auf der Leitung.

Zum Messen von Signalen über die Leseleitung ist Bit vier in Adresse eins zuständig. Ist dieses Bit gesetzt, so liegt High an, anderenfalls Low.

Hierzu einige PEEKs und POKEs:

```
POKE 1,PEEK(1) AND
253:REM 3V SENDEN
POKE 1,PEEK(1) OR 2
:REM 0V SENDEN
L=PEEK(1) AND 224
:REM HIGH bei L<>0
L=PEEK(1) AND 224
:REM LOW bei L=0
```

a.m. □



Jetzt perfekt: Unser Checksummer

Hatte bisher unser Checksummer an Buchstabenvertauschungen nichts auszusetzen, so zeigt er sich nun nicht mehr so kulant.

Ob Sie mit der alten Version nun eingegeben hatten:

10 print"ab"
oder
10 print "ba",

der Checksummer brachte in beiden Fällen die Prüfsumme < gk > . Leicht kann es vorkommen, daß beim schnellen Tippen, besonders im Zehnfingersystem, die Taste, die eigentlich erst als übernächste drankommen sollte, ein wenig zu früh erwischt wird. Dem Checksummer, der lediglich die Ascii-Werte der Buchstaben addierte, konnte dieses natürlich nicht auffallen. Was also tun? Ob etwas früher oder später addiert wird, ändert nichts am Resultat der Summe. Anders ist es, wenn man zwei Verknüpfungsarten kombiniert. So ist z.B. 2*30+40 etwas anderes als 2*40+30. Und genau dieses war dann die Lösung. Die Summe wird nun einfach durch eine Linksverschiebung vor jeder Addition verdoppelt. Dadurch, daß im Falle, wenn das Ergebnis größer als 255 ist, der dabei entstehende Übertrag als Wert 1 zusätzlich addiert wird, verflüchtigen die Werte der am Anfang der Zeile gefundenen Codes sich nicht nach 8 weiteren Zeichen. Damit bleibt nicht nur die Aussagekraft der Prüfsumme voll erhalten, sondern erfährt sogar eine erhebliche Steigerung. Und vor allen Dingen wird nur eine klitzekleine Änderung erforderlich, die dieses zu vollbringen, in der Lage ist. Ein einziges Byte ist nur zu ändern. Wir tun dieses mit "poke 345,10" in der Zeile 470. Dadurch wird das hier ursprünglich ansässige CLC (Clear Carry) durch ASL (Arithmetik Shift Left) ersetzt. Die nachfolgende Addition mit ADC (Addiere mit Carry) addiert den Ascii-Code des gefundenen Zeichens und den nach links herausgeshifteten Übertrag. Da einige unserer Leser beklagten, daß das Checksummerlisting nachher noch im Programmspeicher stehen würde, haben wir diesem noch mit einem "new" abgeholfen. New bzw. neu ist nun folgendes.

10 print"ab" ergibt die Prüfsumme< jd >
10 print"ba" die Prüfsumme<jf >

Sie brauchen den Checksummer nicht neu einzutippen. Alles, was Sie tun müssen, ist, die Zeile 470 anzufügen. An der Bedienung des Checksummers hat sich nichts geändert. Die Eingabehinweise bleiben daher wie gehabt.

EINGABEHINWEISE

Am rechten Rand jedes Listings, jeweils am Ende einer Eingabezeile, finden Sie zwei Buchstaben zwischen einem Kleiner- und einem Größerzeichen eingeschlossen. Diese dürfen Sie nicht mit in Ihr Listing eintippen, sondern sie dienen Ihnen zur Überprüfung Ihrer Eingabe.

Zwischen dem Kleiner- und dem Größerzeichen am rechten Rand befinden sich zwei Buchstaben. Mit einem speziellen Programm können Sie beim Eintippen Ihre Eingabe auf ihre Richtigkeit überprüfen. Dieses Programm, der Checksummer, sorgt nämlich dafür, daß nach erfolgter Zeileneingabe am linken oberen Bildschirmeck zwei Buchstaben ausgegeben werden. Wenn diese Buchstaben nicht mit den vorher erwähnten Buchstaben in unserem Listing übereinstimmen, so können Sie davon ausgehen, daß Sie sich vertippt haben und können sich so die Zeile nochmals näher ansehen, ob Sie Ihren Eingabefehler finden. Wenn Sie dann alles richtig getippt haben, so stimmen die Buchstaben überein und Sie können sich getrost der nächsten Zeile zuwenden.

ERST SICHERN, DANN AUSPROBIEREN

Das Checksummerlisting hat noch keine Prüfsummen. Seien Sie deshalb besonders aufmerksam, daß alles paßt und speichern Sie dieses Programm unbedingt ab, bevor Sie es starten! Bei einem Tippfehler würde es sich wahrscheinlich auf Nimmerwiedersehen verabschieden und Sie müßten die ganze Arbeit vermutlich nochmals machen. Wenn Sie es gestartet haben, so geschieht nichts Besonderes. Der Computer meldet sich einfach kurz darauf mit „READY", und das war auch schon alles. Alles sollte nun wie immer funktionieren, mit der kleinen Ausnahme, daß nunmehr nach jeder Eingabe im Direktmodus eine Prüfsumme erscheint. Nehmen Sie zum Testen irgendeine kurze Basiczeile aus unserem Heft her und testen sie aus. Wenn die Summen übereinstimmen, so können Sie sich freuen, denn Fehler beim Abtippen werden Ihnen nun in Zukunft viel weniger passieren, als vorher.

EINER FÜR ALLE, EIN ECHTES UNIVERSAL-PROGRAMM

Unseren Checksummer können Sie verwenden, ob Sie einen C16/116/Plus4 oder ob Sie einen C64 oder gar einen C128 haben. Nur müssen Sie beim letzteren beachten, ob Sie auch wirklich im 40-Zeichenmodus sind. Nachdem Sie den Checksummer geladen und gestartet haben, können Sie Ihr Basicprogramm eingeben wie gewohnt, Sie können es abspeichern, Sie können auch laden, Sie können Kürzel verwenden und, ob Sie ein paar Leerzeichen mehr oder weniger verwenden, der Checksummer läßt sich dadurch nicht aus der Fassung bringen. Ein bißchen Vorsicht sollte man allerdings walten lassen, wenn man Programme eingetippt hat, in denen Peeks und Pokes vorkommen. Es wird zwar nicht besonders häufig vorkommen, aber es könnte bisweilen ge-

```
10 rem =checksummer==c16 c64 c128==
20 rem (p) 05/87 commodore welt  ==
30 rem ============================
40 rem (c) alfons mittelmeyer    ==
50 rem                           ==
60 rem c16/116/plus4             ==
70 rem c64                       ==
80 rem c128 (40-zeichen)         ==
90 rem ============================
100 rem --------------------------
110 rem   grundroutine (c16)
120 rem --------------------------
130 data165,059,072,165,060,072,032
140 data086,137,104,133,060,104,133
150 data059,152,072,160,000,165,020
160 data024,101,021,170,024,144,011
170 data201,032,240,006,138,024,113
180 data059,234,170,200,177,059,234
190 data208,240,169,031,072,138,074
200 data074,074,074,072,138,041,015
205 data072,169,031,072,162,003,104
210 data024,105,129,157,000,012,202
220 data016,246,104,168,096
230 lt=peek(772):ht=peek(773)
240 fori=312to386:readx:pokei,x:nex
t
250 iflt<>124then350
260 rem --------------------------
270 rem       anpassung c64
280 rem --------------------------
290 fori=312to317:pokei,234:next
300 fori=321to326:pokei,234:next
310 fori=1to6:readad:readx:pokead,x
:next
320 poke380,4:poke319,lt:poke320,ht
:goto430
330 data346,121,347,000,348,002
340 data351,185,352,000,353,002
350 iflt<>13then430
360 rem --------------------------
370 rem anpassung c128 (40 zeichen)
380 rem --------------------------
390 restore410:poke332,22
400 poke335,23:goto310
410 data313,061,316,062,323,062
420 data326,061,347,061,352,061
430 poke772,056:poke 773,1
440 rem --------------------------
450 rem ergaenzung 10/87
460 rem --------------------------
470 poke 345,10:new
480 rem ============================
490 rem = fuer hefte cw 7/87 bis  =
500 rem = cw 9/87 sowie cw128 5/87=
510 rem = und c16 6/87 ist die    =
520 rem = poke-anweisung in zeile =
530 rem = 470 wegzulassen         =
540 rem ============================
```

schehen, daß nach dem Laufenlassen eines Programmes weder der Checksummer noch sonst etwas mehr funktioniert, auch wenn dies bisher ohne Checksummer nicht der Fall gewesen sein sollte. Also bitte sichern Sie in jedem Falle Ihre Programme, bevor Sie sie ausprobieren.

Ein paar Dinge sollten Sie noch wissen. Wir drucken in unseren Listings des öfteren Punkte statt Leerzeichen. Wenn Ihnen nun aber Leerzeichen besser gefallen, so liefert der Checksummer natürlich eine falsche Summe. Wenn Sie diese Richtigkeit überprüfen wollen, so können Sie dies tun, indem Sie sie zuerst einmal so wie im Heft abtippen, und nachher, nachdem Sie sie nachgeprüft haben, einfach wieder die Punkte durch Leerzeichen ersetzen.

A. Mittelmeyer

MONITOR

CHECKMON

```
40 rem checkmon ===============c16 <cn>
50 rem (p) commodore welt team    <ke>
60 rem ============================ <nk>
70 rem (c) by  a.mittelmeyer      <ag>
80 rem c16/116/plus4              <ki>
90 rem ============================ <jg>
100 fori=312to398:reada           <ei>
110 pokei,a:next                  <ep>
120 data 132,218,108,219,0,132,219 <oe>
130 data 164,218,76,75,236,201,62  <nk>
140 data 208,249,165,161,10,101    <jc>
150 data 162,160,7,10,113,161,136  <ej>
160 data 16,250,133,216,169,30,133 <oh>
170 data 217,169,62,160,97,208,220 <mk>
180 data 198,217,208,218,160,105   <ai>
190 data 208,212,201,13,240,4,164  <ha>
200 data 218,24,96,169,60,160,68   <lh>
210 data 32,61,1,165,216,32,16,251 <ec>
220 data 169,62,160,5,208,2,169,32 <om>
230 data 32,75,236,136,208,248,169 <ol>
240 data 13,208,176,219,68,220,1   <go>
250 data 804,56,805,1              <hn>
260 fori=1to4:reada:readb:pokea,b  <lm>
270 next:new                       <ji>
280 rem ========e=n=d=e=========== <cc>
```

"CHECKMON" ist eine unerlaessliche Hilfe zur Eingabe von Maschinenprogrammen. Laden und starten Sie "Checkmon" und gehen dann mittels MONITOR in denselben. Wenn Sie sich nun z.B. mit 'M1000' einen Speicherbereich ansehen, oder Hexzahlen eingeben, so erscheint rechts die Pruefsumme anstatt der Asciicodes.

GUT SORTIERT IST HALB GEFUNDEN

Daten in den Computer einzutippen, sie wieder auf dem Bildschirm oder Drucker auszugeben, ist für die meisten ein Kinderspiel. Schwieriger wird's allerdings, wenn's darum geht, gezielte Ordnung in eine Datei zu bringen. COMMODORE WELT stellt Ihnen die gebräuchlichsten Sortierroutinen vor.

Irgendein Schlaumeier hat einmal das geflügelte Wort von sich gegeben: „Die hälfte seines Lebens sucht der Mensch vergebens...". So banal der Spruch auch klingt, ein Körnchen Wahrheit steckt doch drin.

Jede Kartei, jedes Verzeichnis richtet sich nach gewissen Suchkriterien aus. Der gewünschte Name, die gesuchte Adresse soll ja ohne großen Zeitverlust möglichst nervenschonend gefunden werden. Bei der Dateiablage im Computer ist der Idealfall, die geforderten Daten auf Knopfdruck abrufen zu können.

Es gibt verschiedene Möglichkeiten, zum Ziel zu kommen. Da wir ziemlich sicher sind, daß der größte Teil unserer Leser in BASIC programmiert, wollen wir Ihnen hier ein paar Sortierunterprogramme vorstellen. Die Bezeichnung „Unterprogramme" sagt es eigentlich schon: Sie sind dazu gedacht, irgendwo in einer Ihrer eigenen Software-Entwicklungen eingebaut zu werden.

SORTIEREN FÜR LEICHTERES SUCHEN

Suchen und Sortieren arbeiten Hand in Hand. Es ist für den Computer ungleich einfacher, aus einer bereits sortierten Datenmenge das Richtige herauszufinden, wobei es keine Rolle spielt, ob es sich um numerische Daten (Zahlen) oder Texte (Strings) handelt. Bei einer vorsortierten Datei können zwei Suchmethoden programmiert werden:

a) Durchsuchen sämtlicher Daten (Wörter, Nummern, Texte),
b) „Halbierungs"-Methode, auch binärer Suchmodus genannt.

Keiner großen Erklärung bedarf der erste Punkt. Wenn Sie den Namen Meier suchen, so durchforstet Ihr Computer alle zur Verfügung stehenden Daten, von Anfang bis Ende, bis er ihn gefunden hat. Ihr Pech, wenn die Datenmenge recht groß ist, dann dauert es eben seine Zeit.

Ein ganzes Stück schneller geht es mit der „Halbierungsmethode" vonstatten. Hier ist aber Voraussetzung, daß die Daten bereits sortiert sind. Sucht der Computer nach einem bestimmten Wort oder einer Zahl, so müssen Sie ihm ein „Erkennungsmerkmal" angeben, das der Begriff besitzt. Am einfachsten sind für die meisten Programmierer die ASCII-Werte der Zeichen auf der Commodore-Tastatur.

Der Buchstabe "A" beispielsweise hat den ASC-Code 65, „B" liegt um eins höher, also bei 66. Am besten prüfen Sie's nach mit PRINT ASC("A"), oder welches Zeichen Sie auch immer vorziehen. Zurück zu unserem Meier: Bei einem sortierten Datenfeld stellt der Computer fest, daß dieser Name mit dem von Ihnen vorgegebenen Merkmal (ASC-Wert) erst in dem Teil des Datenfeldes beginnen kann, dessen ASC-Werte gleich oder größer 77 sind (das ist nämlich der ASC-Wert von „M"). Das Gesamtfeld wird also halbiert, das verbleibende ebenso. So geht es weiter, bis nur noch zwei zu durchsuchende Elemente übrig sind, wovon dann eines todsicher das gewünschte ist. So weit scheint der Sinn des Sortierens einleuchtend zu sein. Nur: wie stellt man's an?

Es gibt viele Arten von Sortiermethoden, wovon einige allerdings recht unterschiedliche Sortierzeiten für große Datenmengen benötigen: Einfüge-, Tausch-, Einzelbyte- und Selektierungsmethode. Diese Klassiker wurden bereits zu Anfang der BASIC-Programmierung entwickelt, haben aber bis heute von ihrer Effizienz nichts verloren.

DIE INSERT-METHODE: AM HÄUFIGSTEN ANGEWANDT

Bei der Insert-Methode macht's der Computer ganz unkompliziert: Er beginnt bei der ersten Angabe und vergleicht sie mit der nächsten. Sie ist kleiner als die erste, so wird sie vor dieser eingefügt (daher der Name). Das dritte Element wird nun mit dem zweiten verglichen und ebenso behandelt, bis alle Datensätze und -felder „gecheckt" sind.

Die Insert-Methode ist die wohl simpelste, aber nicht unbedingt langsamste Sortierroutine.

UNMITTELBARE AUSWAHL: SELECTION

Ein weiterer Sortieralgorithmus, der neben dem Einfügeprinzip zusätzlich das des Tauschens verfolgt, ist die Auswahl-Methode. Auch hier werden zwei For-Next-Schleifen verwendet. Die erste beginnt beim höchsten Element des Datenfeldes und arbeitet sich vor bis zum nächsten. Die zweite Schleife stellt die einzelnen Datenfelder, von vorne beginnend, dagegen.

Der Grundgedanke dabei ist, daß das momentan größte Element an die letzte Stelle des gesamten Arrays gebracht wird. Die Größe des Datensatzes wird jetzt um Eins reduziert, dann erneut nach dem größten Teileelement gesucht. Das wird wiederum hinten angefügt, ohne das vorher an die letzte Stelle gebrachte Element zu beeinflussen, da dieses ja von der Gesamtmenge abgetrennt wurde. Ist nach den verschiedenen Durchläufen nur noch ein Teilelement übriggeblieben, so ist dieses – logischerweise – das erste.

VORSORTIEREN BRINGT'S NICHT IMMER

Allerdings sind mit der Selektier-Methode die besten zeitlichen Ereignisse nur dann zu erzielen, wenn keine noch so grobe Vorsortierung vorliegt. Dann kann

es nämlich sein, daß sich die höchsten Elemente bereits ziemlich am Ende des Arrays befinden, die Selektier-Routine aber unverdrossen von vorne zu suchen beginnt. Bei einem zufällig entstandenen Datensatz werden die besten Ergebnisse erzielt.

HAT NICHTS MIT KAUGUMMI ZU TUN: BUBBLESORT

Bubble Sort ist wohl eine der bekanntesten und am häufigsten benutzten Sortierroutinen. Sie arbeitet mit dem Prinzip des Tauschens. Das erste Element des gesamten Arrays wird mit dem folgenden verglichen. Ist es größer, so wird ausgetauscht. Auf diese Weise gelangt das größte Element des Datensatzes automatisch ans Ende, was ja auch der Sinn der Übung war.

„Bubble" bedeutet soviel wie „Luftblase" und beschreibt leicht humorig die Art, sich – bildlich gesprochen – das jeweils größte Datenelement durch die ganze Sortierroutine hindurch bis ans Ende des Arrays bewegt. Der Unterschied zu den bereits erwähnten Einfügemethoden besteht darin, daß Bubble Sort beim größten Datenelement zu sortieren beginnt, nicht beim kleinsten. Der Zeitaufwand dürfte in etwa derselbe sein.

DIE ROUTINE NACH SHELL

Der nächste Sortier-Algorithmus, nach dessen Erfinder benannt, hat seit nahezu dreißig Jahren unverändert seine Gültigkeit. Obwohl Shell ebenfalls das Prinzip des Einfügens verfolgt, ist der Aufbau der Routine doch ein bißchen komplizierter. Es werden nämlich nicht nur unmittelbar aneinander grenzende Daten miteinander verglichen, sondern vor allem entfernt liegende, (etwa der erste mit dem sechsten). Dieser Modus ist im Programm vorzugeben.

So wird eine Art Vorsortierung erreicht. Die entsprechenden Daten werden auf neue Variablen übertragen und ebenfalls wieder sortiert, diesmal aber mit reduzierten Abständen, beispielsweise das erste mit dem vierten. Unser Sortiersieb wird auf diese Weise immer enger, so daß zum Schluß nur noch der Einser-Schritt übrig bleibt, was gleichbedeutend mit einer exakten Sortierung ist.

HILFSROUTINE: EINFÜGEN

Genau genommen besteht dieser Sortiermodus aus zwei Sortierroutinen, denn die Grobsortierung übernimmt das bereits bekannte Insert. Da jetzt dem Hauptprogramm bereits vorbearbeitete Daten vorgelegt werden können, geht die weitere Abarbeitung sehr rasch. Shell Sort bewährt sich ideal bei recht großen Datensätzen (Adressen, Text) mit vielen Einzelelementen.

TREFFENDER NAME: QUICK SORT

Der nächste Sortier-Oldie, den wir uns ansehen, heißt Quick Sort. Er wurde etwa zur selben Zeit wie Shell Sort entwickelt; sein Erfinder heißt Roare, ebenfalls ein Amerikaner. Quick Sort, die wohl gebräuchlichste Sortierroutine, vermutlich weil sie bei sehr großen Datenmengen einwandfrei die schnellste ist.

Bei kleineren Feldern haben Insert- und Selektiermethode die Nase vorn, einfach deswegen, weil sie viel simpler aufgebaut sind und der Computer nicht so viele Programmschritte abarbeiten muß. Bei großen Arrays dagegen macht so ein einfacher Algorithmus stur seine Programmschritte, auch wenn die Sortierung so gut wie abgeschlossen ist. Eine „intelligente" Routine wie Quick-Sort erkennt das aber beizeiten und ist dadurch, obwohl softwaretechnisch weit komplizierter angelegt, viel schneller fertig.

Aus dem gesamten Array wird irgendein Element herausgezogen und abgesondert, weil es ab sofort als Vergleichsmuster für alle weiteren Datenelemente fungiert. Nach dem Motto: „Kleiner als" oder „größer als" werden jetzt die übrigen Daten links oder rechts von unserem Muster plaziert. Die erste Vorsortierung ist damit erreicht.

Nun wird jedes so gewonnene Array (das mit den kleineren und das mit den größeren Werten) genauso behandelt wie das gesamte Datenfeld vorher: wieder ein Vergleichsmuster herausgezogen und wieder ein „Kleiner" und „Größer"-Datensatz geschaffen. Auf diese Weise wird die Einteilung immer ‚feiner', bis das fertig sortierte Array vor uns steht. Ideal für diesen Sortieralgorithmus erweist sich natürlich die Auswahl des Vergleichselements aus der Mitte der Teildatensätze.

AUF ANWENDUNG UND MENGE KOMMT ES AN

Welchen der vorgestellten Algorithmen Sie künftig für Ihre selbstfabrizierten Dateiverwaltungen verwenden möchten, überlassen wir Ihnen und dem dafür vorgesehenen Anwendungszweck. Beachten Sie bitte dabei immer die Menge der eingegebenen und zu sortierenden Daten sowie den Grad der bereits bei der Eingabe vorliegenden Grobsortierung.

Quick Sort beispielsweise ist bei zufällig zusammengewürfelten Werten am effektivsten. Bei einer nahezu perfekt alphabetisch geordneten Datei kann es aber ohne weiteres sein, daß Bubble- oder Insert Sort um einiges schneller sind. Wenn Sie das beachten, dann sind wir sicher, daß Sie das Optimale aus diesen Sortierroutinen herausholen können.

Alle hier besprochenen Programme sind als Listing abgedruckt, als Daten haben wir zwanzig Begriffe aus dem Umfeld des Computers gewählt. Den Sortierablauf jeder Routine können Sie am Bildschirm in groben Zügen verfolgen, das fertig sortierte Array wird als „Erfolgsmeldung" rechts am Bildschirm ausgegeben. Wollen Sie die eine oder andere Routine in eigenen Programmen verwenden, so lassen Sie bitte die Zeilen weg, die Sie nicht benötigen (Bildschirmausgabe, DATA-Werte und dergleichen). Am besten bauen Sie so ein Sortprogramm als Unterroutine ein, die mit GOSUB angesprungen werden kann.

Da es sich um reine BASIC-Routinen ohne PEEKS und POKES handelt, sind sie für jeden Commodore-Homecomputer geeignet, vom VC20 und C16 bis hin zum 128PC. *B.U.* ■

Insert-Sort — 1 — defekt

```
10 rem sortierroutine                    <ab>
20 rem insert sort                       <pk>
30 fori=1to26:cr$=cr$+chr$(29)
:next                                    <dn>
40 ci$=left$(cr$,12):rn$=chr$(
18)                                      <jn>
50 m=20:dims$(m)                         <eo>
60 scnclr:printrn$"unsortiert:
"                                        <ak>
70 fori=1tom:reads$(i):prints$
(i):next:gosub 200                       <mi>
80 fori=2tom                             <kl>
90 ifs$(i)>=s$(i-1)then160:rem
 alles so lassen                         <kb>
100 i$=s$(i):printci$;i;i$:gos
ub210                                    <mg>
110 forj=i-1to1step-1                    <db>
120 s$(j+1)=s$(j):printci$;j;s
$(j+1):gosub210                          <co>
130 ifi$<=s$(j-1)then150:rem e
infuegen                                 <pe>
140 s$(j)=i$:printci$;j;s$(j):
gosub210:goto160                         <ao>
150 nextj                                <kj>
160 nexti                                <na>
170 printchr$(19)cr$rn$"sortie
rt:"                                     <la>
180 fori=i=1tom-9step10                  <fd>
190 forj=1to1+9:printcr$s$(j):
nextj,i:end                              <ch>
200 printchr$(19)ci$rn$"sortin
g:":return                               <ii>
210 fort=1to500:next:return              <ik>
220 datapascal,diskette,floppy
,sprites,drucker,chip,ram,rom,
videochip,basic                          <fb>
230 dataassembler,maske,amiga,
datei,text,grafik,pixel,softwa
re,tastatur,monitor                      <ob>
```

Select-Sort — 1 — 3 Sekunden

```
10 rem sortierroutine                    <ab>
20 rem selection-sort                    <fg>
30 fori=1to26:cr$=cr$+chr$(29)
:next                                    <dn>
40 ci$=left$(cr$,12):rn$=chr$(
18)                                      <jn>
50 m=20:dims$(m)                         <eo>
60 scnclr:printrn$"unsortiert:
"                                        <ak>
70 fori=1tom:reads$(i):prints$
(i):next:gosub160                        <na>
80 fori=mto2step-1:m$=""                 <no>
90 forj=1toi                             <cn>
100 ifs$(j)>m$thenm$=s$(j):k=j           <cg>
110 nextj                                <ag>
120 s1$=s$(i):s$(i)=s$(k):s$(k
)=s1$:printci$;k;s$(k)                   <oo>
130 nexti                                <fi>
140 printchr$(19)cr$rn$"sortie
rt:"                                     <jl>
150 forj=1tom:printcr$s$(j):ne
xtj:end                                  <ml>
160 printchr$(19)ci$rn$"sortin
g:":return                               <ap>
170 fort=1to500:next:return              <ab>
180 datapascal,diskette,floppy
,sprites,drucker,chip,ram,rom,
videochip,basic                          <ie>
190 dataassembler,maske,amiga,
datei,text,grafik,pixel,softwa
re,tastatur,monitor                      <ok>
```

Bubble-Sort — 1 — 64 Sekunden

```
10 rem sortierroutine                    <ab>
20 rem bubble sort                       <be>
30 fori=1to26:cr$=cr$+chr$(29)
:next                                    <dn>
40 ci$=left$(cr$,12):rn$=chr$(
18)                                      <jn>
50 m=20:dims$(m)                         <eo>
60 scnclr:printrn$"unsortiert:
"                                        <ak>
70 fori=1tom:reads$(i):prints$
(i):next:gosub160                        <na>
80 fori=m-1to1step-1:rem beim
vorletzten anfangen                      <mj>
90 forj=1toi:rem gegenprobe vo
n vorne                                  <jc>
100 ifs$(j)<=s$(j+1)then120              <bl>
110 s1$=s$(j):s$(j)=s$(j+1):s$
(j+1)=s1$:printci$;j;s$(j+1):g
osub170                                  <aj>
120 nextj                                <co>
130 nexti                                <fi>
140 printchr$(19)cr$rn$"sortie
rt:"                                     <jl>
150 forj=1tom:printcr$s$(j):ne
xtj:end                                  <ml>
```

```
160 printchr$(19)ci$rn$"sortin
g:":return                    <ap>
170 fort=1to500:next:return   <ab>
180 datapascal,diskette,floppy
,sprites,drucker,chip,ram,rom,
videochip,basic               <ie>
190 dataassembler,maske,amiga,
datei,text,grafik,pixel,softwa
re,tastatur,monitor           <ok>
```

Shell-Sort 63 Sekunden

```
10 rem sortierroutine         <ab>
20 rem shell-sort             <gg>
30 fori=1to26:cr$=cr$+chr$(29)
:next                         <dn>
40 ci$=left$(cr$,12):rn$=chr$(
18)                           <jn>
50 m=20:dims$(m),s1$(m):sw=int
(m/2)                         <mp>
60 scnclr:printrn$"unsortiert:
"                             <ak>
70 fori=1tom:reads$(i):prints$
(i):next:gosub320             <na>
80 fori=1tosw                 <lc>
90 forj=1toint(m/sw)          <mc>
100 s1$(j)=s$((j-1)*sw+i)     <kf>
110 nextj                     <ag>
120 s1=j-1:gosub220:rem insert
-sort benutzen                <nd>
130 forj=1toint(m/sw)         <jd>
140 s$((j-1)*sw+i)=s1$(j):prin
tci$;j;s$((j-1)*sw+i):gosub330 <ol>
150 nextj                     <kj>
160 nexti                     <na>
170 sw=int(sw/2)              <cd>
180 ifswthengoto80            <pe>
190 printchr$(19)cr$rn$"sortie
rt:"                          <ca>
200 forj=1tom:printcr$s$(j):ne
xtj:end                       <pn>
210 rem untersortierroutine   <pb>
220 fori1=2tos1               <ki>
230 ifs1$(i1)>=s1$(i1-1)then30
0                             <bg>
240 i1$=s1$(i1)               <bg>
250 forj1=i1-1to1step-1       <hg>
260 s1$(j1+1)=s1$(j1)         <em>
270 ifi1$<=s1$(j1-1)then290   <nf>
280 s1$(j1)=i1$:goto300       <nl>
290 nextj1                    <oi>
300 nexti1                    <df>
310 return                    <pm>
320 printchr$(19)ci$rn$"sortin
g:":return                    <oo>
330 fort=1to500:next:return   <od>
340 datapascal,diskette,floppy
,sprites,drucker,chip,ram,rom,
videochip,basic               <co>
350 dataassembler,maske,amiga,
datei,text,grafik,pixel,softwa
re,tastatur,monitor           <fi>
```

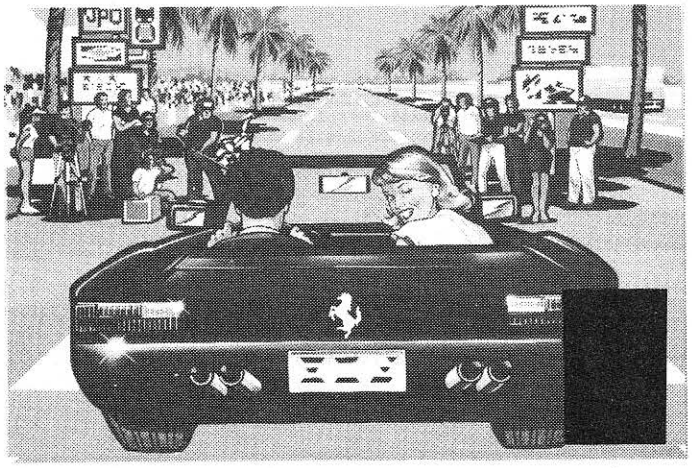
Quick-Sort 30 Sekunden

```
10 rem sortierroutine         <ab>
20 rem quick-sort             <be>
30 fori=1to26:cr$=cr$+chr$(29)
:next                         <dn>
40 ci$=left$(cr$,12):rn$=chr$(
18)                           <jn>
50 m=20:mg=100:dims$(m),le(mg)
,ri(mg)                       <gk>
60 x=0:le(1)=1:ri(1)=m        <jg>
70 scnclr:printrn$"unsortiert:
"                             <gk>
80 fori=1tom:reads$(i):prints$
(i):next:printchr$(19)ci$rn$"s
orting:"                      <jc>
90 gosub120                   <gd>
100 printchr$(19)cr$rn$"sortie
rt:"                          <mk>
110 forj=1tom:printcr$s$(j):ne
xtj:end                       <aa>
120 x=x+1:ifle(x)>=ri(x)then24
0                             <ag>
130 y=le(x):w=ri(x)           <ci>
140 s1$=s$(int((y+w)/2))      <no>
150 ify>wthen220              <mm>
160 ifs$(y)<s1$theny=y+1:goto1
60                            <ic>
170 ifs$(w)>s1$thenw=w-1:goto1
70                            <nc>
180 ify>wthen220              <ko>
190 s2$=s$(y):s$(y)=s$(w):s$(w
)=s2$:printci$;w;s$(w):gosub25
0                             <dd>
200 y=y+1:w=w-1               <hd>
210 goto150                   <na>
220 ri(x+1)=w:le(x+1)=le(x):go
sub120                        <fe>
230 le(x+1)=y:ri(x+1)=ri(x):go
sub120                        <ch>
240 x=x-1:return              <ck>
250 fort=1to500:next:return   <be>
260 datapascal,diskette,floppy
,sprites,drucker,chip,ram,rom,
videochip,basic               <ce>
270 dataassembler,maske,amiga,
datei,text,grafik,pixel,softwa
re,tastatur,monitor           <fm>
```

Road Race

**Das Geld liegt auf der Straße.
Sie brauchen es nur aufzuheben. Zumindest beim Autorennen Road-Race.**

Ziel von Road-Race ist es, möglichst viele Kilometer auf der Rennstrecke zurückzulegen und dabei Geldscheine und Diamanten einzusammeln. Hübsche Grafikeffekte und hohe Geschwindigkeit machen den Reiz dieses spannenden Spieles aus.
Wer in einen die Straße begrenzenden Strohballen oder in eine Straßensperre rast, verliert seinen Wagen. Das braucht aber noch nicht das Aus zu bedeuten. Sind 2000 Punkte erreicht, gibt es ein Extra-Auto, ebenso nach allen weiteren 10000.
Nach dem Start des Programms entsteht durch das Einlesen des Zeichensatzes und der Rennstrecke zunächst eine Pause von etwa sieben Sekunden. Bitte nicht ungeduldig werden, denn das Warten lohnt sich. Nachdem das Titelbild mit Musik erschienen ist, können Sie zwischen unterschiedlichen Fahrbedingungen wählen.



Mit der Taste F1 bestimmen Sie, ob viel oder wenig auf den Straßen los sein wird. Mit wachsender Dichte der Geldscheine und Diamanten steigt jedoch auch die Anzahl der Straßensperren. Die Geschwindigkeit wählen Sie mit der Taste F2.
Damit Sie auch bei noch ungeübter Fahrweise etwas erreichen können und nicht mit schrottreifem Wagen liegenbleiben, stehen Ihnen mehrere Autos zur Verfügung. Mit F3 wählen Sie deren Anzahl. Durch Druck der HELP-Taste bekommen Sie eine kurze Übersicht über Punktewertung und Steuerung.
Mit der Leertaste geht das Rennen los. Der Wagen wird entweder mit den Tasten A und L oder mit dem Joystick in Port eins gesteuert. Das Rennen endet, wenn Sie alle Autos zu Schrott gefahren haben.
Bei ausreichendem Punktestand dürften Sie Ihren Namen in die Highscore-Liste eintragen. Sogar deutsche Umlaute sind hierbei möglich:
Ä=[cbm] + [–] Ö=[cbm] + [–] Ü=[cbm] + [K]
Sollten Sie in der schwierigsten Stufe 100000 Punkte schaffen, sind Sie ein zweiter Nicki Lauda. □

```
10 rem road-race===============p4 <cm>
20 rem (p) commodore welt team   <ho>
30 rem ========================= <ng>
40 rem (c) by axel borchmann     <fp>
50 rem                           <pd>
60 rem                           <ah>
70 rem basic v3.5                <nl>
80 rem plus4 (c16/116 + 64kb)    <fd>
90 rem ========================= <jg>
100 poke65286,0:poke52,80:poke56,8
0:poke65298,192:poke65299,80:clr:g
osub3310                         <fl>
110 color0,1,0:color4,1,0:printhe$
he$cl$chr$(8)                    <ke>
120 neu=2000:vol7                <hg>
130 sc$="c1985 axel computing"+b4$
+"c1985 axel computing"+b4$+"c1985
 axel computing"                 <jd>
140 dimvk$(4),tp$(3),at$(5),tn(144
),hi(8),hi$(8),a$(6),b$(6),x$(11):
gosub2260                        <jj>
150 tp$(1)="langsam":tp$(2)="norma
l ":tp$(3)="schnell"             <jb>
160 at$(1)="eins ":at$(2)="zwei ":
at$(3)="drei ":at$(4)="vier ":at$(
5)="f=nf"                        <jo>
170 vk$(1)="wenig"+b4$:vk$(2)="nor
mal"+b3$:vk$(3)="viel"+b5$:vk$(4)=
"sehr viel"                      <eg>
180 restore:gosub2260            <hc>
190 fori=1to140:readtn(i):next   <cc>
200 fori=1to8:readhi(i),hi$(i):nex
t                                <gd>
210 fori=2048to2083:reada:pokei,a:
next:sys2048                     <ff>
220 reada:ifa=256then240         <fd>
230 fori=0to7:readc$:poke20480+a*8
+i,dec(c$):next:goto220          <hn>
240 pudef"@":gosub1380           <ag>
250 color0,16,2:color4,16,2      <jb>
260 printhe$he$cl$               <gd>
270 restore:gosub2260            <mb>
280 b=4032:p=0:km=0              <fi>
290 printbk$                     <lo>
300 gosub980                     <ma>
310 x=21:t=17:poke2022,3         <jg>
320 pokeb+x,66:poke3008+x,127    <jk>
330 fori=0to20:printtab(t);oe$x$(0
):next                           <pb>
340 pokeb+x,42:poke3008+x,127    <gp>
350 poke 239,0                   <na>
360 printhe$oe$chr$(27)"i":t=t+(rn
d(1)>.5)*2+1                     <cj>
370 ift=0thent=2                 <no>
380 ift=30thent=28               <dl>
390 ifrnd(1)>vkthen420           <an>
400 printtab(t)x$(0)             <bi>
410 goto430                      <gn>
420 printtab(t)x$(int(rnd(1)*12))  <oj>
430 r=peek(b+x)                  <kc>
440 ifr=42orr=86then600          <kf>
450 ifr=83then850                <ii>
460 ifr=32then540                <ah>
470 ifr=42orr=86then600:remkaputt  <gp>
480 ifr=83then850                <mk>
490 ifr=65then860                <hk>
500 ifr=90then870                <gi>
510 ifr=88then880                <gf>
520 ifr=81then890                <eg>
530 ifr=87then900                <gm>
540 geta$:l=joy(1)               <ee>
550 ifa$="a"orl=7thenx=x-1       <jc>
560 ifa$="l"orl=3thenx=x+1       <gf>
570 pokeb+x,66:poke3008+x,127    <jp>
580 km=km+1:printlg$he$he$c4$left$
(qr$,13);:printusing"######";km:po
ke2022,3                         <bb>
590 forwa=1totp:next:goto360     <ik>
600 rem                          <ed>
610 poke3008+x,127:vol7          <ae>
620 fora=0to20step2:fori=76to79  <hj>
630 pokeb+x,i:sound3,a*50,1      <dc>
640 poke3008+x,int(rnd(1)*127):nex
ti,a                             <ja>
650 poke3008+x,0:pokeb+x,80      <lp>
660 za=76:fora=24to12step-1      <bl>
670 p0=3072+a*40+(x-(25-a)):p1=307
2+a*40+(x+(25-a))                <mp>
680 pokep0,za:pokep1,za          <ad>
690 pokep0+960-a*40,za:pokep1+960-
a*40,za                          <go>
700 za=za+1:ifza=80thenza=76     <fg>
710 pokep0,32:pokep1,32:pokep0+960
-a*40,32                         <ke>
720 pokep1+960-a*40,32:volint(a/2-
6):sound3,1000,9:next            <oe>
730 sound3,0,0:vol7:ki=1:po=1    <mp>
740 km=km-ki:p=p+po              <ol>
750 printhe$he$c4$left$(qr$,13)wh$
;:printusing"######";km;         <ak>
760 printleft$(qr$,11)wh$;:printus
ing"######";p                    <ba>
770 sound1,1000,2:ifkm=0then800  <in>
780 ifint(km/10)=km/10thenki=10:po
=10                              <pi>
790 goto740                      <fd>
800 km=0:forl=1to3:sound1,500,15:f
ori=7to0step-.1:voli:nexti:forq=1t
o50:nextq,l                      <ji>
810 lb=lb-1:iflb=0then830        <ol>
820 x=21:printhe$he$c4$c4$c4$chr$(
27)"t"cl$:goto300                <jm>
830 printhe$he$c4$bk$yo$oe$fl$" en
de"fo$;bk$                       <pb>
840 goto1100                     <mm>
850 p=p+50:goto910               <gl>
```

```
860 p=p+40:goto910                                <dn>
870 p=p+30:goto910                                <cl>
880 p=p+20:goto910                                <nl>
890 p=p+1000:goto910                              <oa>
900 p=p+2000:goto910                              <mp>
910 vol7:fori=0to255step50:sound1,
i,1:next                                          <gd>
920 pokeb-40+x,32                                 <ko>
930 ifp>neuthen950                                <jm>
940 gosub980:goto360                              <ge>
950 lb=lb+1:iflb>6thenlb=6                        <kj>
960 neu=neu+10000                                 <hn>
970 goto940                                       <jo>
980 poke2021,4:poke2022,0                         <ki>
990 printhe$bk$"U";:xx=6:dd$=zv$:g
osub3560:print"IU";:xx=11:gosub356
0:print"IU";                                     <ba>
1000 xx=15:gosub3560:print"I"                     <ka>
1010 a$="":fori=1tolb:a$=a$+ym$:ne
xt                                                <ph>
1020 printyo$lg$;:printusing"#####
#";a$;                                            <ho>
1030 printbk$yo$yo$"km:"c3$lg$;:pr
intusing"######";km;                              <pm>
1040 printbk$" "yo$yo$"punkte: "lg
$;:printusing"######";p;:printbk$"
 "yo$;                                            <bo>
1050 printbk$                                     <ha>
1060 print"J";:dd$=zv$:xx=6:gosub3
560:print"KJ";:xx=11:gosub3560:pri
nt"KJ";                                           <gi>
1070 xx=15:gosub3560:print"K"                     <db>
1080 poke2021,24:poke2022,3                       <aa>
1090 return                                       <bk>
1100 rem                                          <cp>
1110 geta$:ifa$=""then1130                        <eb>
1120 goto1110                                     <pf>
1130 poke2022,3:fori=1to24:printch
r$(27)"w":next                                    <oh>
1140 gosub1150:goto1220                           <pa>
1150 poke65286,0:printcl$c4$c4$lef
t$(qr$,12)bk$"weltrangliste"c4$;c4
$                                                 <ef>
1160 color1,3,7:printfl$left$(qr$,
5);:printusing"######";hi(1);                     <hb>
1170 printb2$"-----"b2$hi$(1)fo$c4
$                                                 <ef>
1180 fori=2to6:printleft$(qr$,5);:
color1,3,9-i:printusing"######";hi
(i);                                              <dc>
1190 printb2$"-----"b2$hi$(i):next                <fk>
1200 fori=7to8:printleft$(qr$,5);:
color1,3,0:printusing"######";hi(i
);                                                <km>
1210 printb2$"-----"hi$(i):next:po
ke65286,27:return                                 <ah>
1220 ifp<hi(8)thenprintc4$c3$oe$"l
eider d=rfen sie sich nicht eintra
gen":goto1310                                     <kc>
1230 printc4$left$(qr$,4)oe$"bravo
 sie d=rfen sich eintragen"                       <dp>
1240 fori=8to1step-1:ifp>hi(i)then
x=i                                               <fk>
1250 next                                         <ep>
1260 fori=8toxstep-1:hi(i)=hi(i-1)
:hi$(i)=hi$(i-1):next                             <ne>
1270 print c4$"wie heissen sie:"ch
r$(27)"t"left$(qr$,15)chr$(27)"b"                 <fi>
1280 printcl$left$(qr$,14);:inputn
am$                                               <ao>
1290 hi(x)=p:hi$(x)=nam$                          <jf>
1300 printhe$he$:poke2022,3:gosub1
150                                               <fb>
1310 printc4$c3$"noch einmal"lg$"
(j/n) :"                                          <aa>
1320 getkeya$:ifa$="n"then1350                    <mi>
1330 ifa$<>"j"then1320                            <jh>
1340 printhe$he$cl$:color0,1,0:col
or4,1,0:forl=1to200:next:goto240                  <lk>
1350 printhe$he$cl$:poke65286,16:p
rintcl$b4$b3$"dann eben nicht"                    <pe>
1360 poke65298,196:poke65299,208                  <ad>
1370 fori=16to23:poke65286,i:forl=
1to10:nextl,i:end                                 <fg>
1380 vol7:color0,1,0:color4,1,0                   <il>
1390 a$(0)="UDDIUDDIUDDIUDDDI"                    <nd>
1400 a$(1)="GUIHGUIHGUIHJIUIH"                    <oo>
1410 a$(2)="TJKHTGHYTGHY HGHY"                    <na>
1420 a$(3)="T UK"+z4$+"TY"+z6$+z4$
+"JK"+z6$+" YTY"+z6$                              <co>
1430 a$(4)="T JITGHYTUIY HGHY"                    <pn>
1440 a$(5)="GUIHGJKHGGHHUKJKH"                    <ej>
1450 a$(6)="JKJKJFFKJKJKJFFFK"                    <lj>
1460 b$(0)="UDDIUDDIUDDIU"+zv$+zv$
+"I"                                              <ao>
1470 b$(1)="GUI"+yo$+"GUIHGUI"+yo$
+"GU"+zv$+"K"                                     <jn>
1480 b$(2)="TJK"+yo$+"TGHYTGJKTJI
"                                                 <cb>
1490 b$(3)=z4$+" UK"+z4$+"JK"+z6$+
z4$+"T"+b2$+z4$+" "+yo$+" "                       <of>
1500 b$(4)="T JITUIYTGUITUK "                     <hh>
1510 b$(5)="GUI"+yo$+"GGHHGJK"+yo$
+"GJ"+zv$+"I"                                     <gd>
1520 b$(6)="JKJKJKJKJFFKJFFK":poke
65286,27                                          <gl>
1530 fori=0to5:color 1,12,i:printh
e$left$(qd$,5)b6$"axel computing p
r"yn$"sentiert":next                              <ho>
1540 forl=0to2000:next:fori=5to0st
ep-1:color1,12,i                                  <ml>
1550 printhe$left$(qd$,5)b6$"axel
computing pr"yn$"sentiert":next:pr
intcl$                                            <mb>
1560 poke65286,0:printhe$left$(qr$
,15)"the great"c4$                                <ee>
```

```
1570 fori=0to6:color1,7,i:printb3$
a$(i)" "b$(i):next                    <bb>
1580 printre$c4$b$"c1985 axel comp
uting"                                <pa>
1590 poke2022,11:poke2023,1:poke20
24,38                                 <cn>
1600 color1,9,2:printhe$rn$z8$;:fo
ri=2to7:color1,9,i:printb3$;:next     <mk>
1610 fori=7to2step-1:color1,9,i:pr
intb3$;:next:color1,9,2:printyq$;     <dh>
1620 fori=1to5:color1,9,i:printrn$
" "left$(qr$,36)" ";:next             <ka>
1630 fori=7to2step-1:color1,9,i:pr
intrn$" "left$(qr$,36)" ";            <ip>
1640 next                             <fo>
1650 color1,9,2:printrf$yq$rn$;:fo
ri=2to7:color1,9,i:printb3$;:next     <me>
1660 fori=7to2step-1:color1,9,i:pr
intb3$;:next:color1,9,2:printrf$z8
$;                                    <of>
1670 printchr$(27)"w"he$              <jp>
1680 poke2021,23:poke2022,13:poke2
023,2:poke2024,37:printcl$            <ok>
1690 printhe$chr$(27)"t"c4$;c1$lef
t$(qd$,11)chr$(27)"b"                 <go>
1700 fori=1to8:keyi,chr$(132+i):ne
xt                                    <pn>
1710 co=0:lb=3:vk=2:tp=2:sc=1:poke
1345,1                                <oo>
1720 color1,3,6:printhe$c4$c4$c4$b
4$b3$"'f1'"b2$"verkehr"b3$":"vk$(v
k)                                    <ao>
1730 color1,3,4:printb4$b3$"'f2'"b
2$"tempo"b5$":"tp$(tp)                <ja>
1740 color1,3,3:printb4$b3$"'f3'"b
2$"autos"b5$":"at$(lb)                <co>
1750 color1,3,2:printb6$"'help' an
leitung"                              <nh>
1760 color1,3,0:printb5$"'space' s
tart";:poke65286,27                   <hp>
1770 geta$                            <la>
1780 color1,16,6:printhe$c3$mid$(s
c$,sc,34);                            <hi>
1790 sc=sc+1:ifsc=25thensc=1          <bo>
1800 co=co+1:ifco=141thenco=0         <pi>
1810 iftn(co)=0thenforl=1to10:next
:goto1830                             <lo>
1820 sound1,tn(co),11                 <ah>
1830 ifa$=f1$ then1890                <jf>
1840 ifa$=f3$ then1910                <jl>
1850 ifa$=f5$ then1930                <eh>
1860 ifa$=f8$ then1950                <db>
1870 ifa$=" " then2170                <go>
1880 forl=1to85:next:goto1770         <ml>
1890 vk=vk+1:ifvk=5thenvk=1           <no>
1900 goto1720                         <oj>
1910 tp=tp+1:iftp=4thentp=1           <di>
1920 goto1720                         <ic>
1930 lb=lb+1:iflb=6thenlb=1           <mg>
1940 goto1720                         <ag>
1950 color1,3,6:printcl$              <na>
1960 printb4$"steuerung"b4$;:color
1,9,2:printyo$;:color1,3,6:printb2
$"punktezaehlung"                     <bk>
1970 color1,9,3:printb$b4$b3$yo$      <mi>
1980 color1,9,4:printtab(19)yo$" "
;:color1,12,0:print"X "wh$" -"b3$;
:color1,11,6                          <ha>
1990 print"20 punkte"                 <el>
2000 color1,14,5:printb4$"a"wh$" -
 "oe$"Q "wh$"- ";                     <po>
2010 color1,14,5:print"l"b4$;:colo
r1,9,5:printyo$;                      <aa>
2020 color1,12,2:print" Z"b2$wh$"-
"b3$;:color1,11,5:print"30 punkte"    <pb>
2030 color1,9,6:printtab(19);yo$"
";:color1,12,3:print"A "wh$" -"b3$
;:color1,11,4                         <bg>
2040 print"40 punkte"                 <ke>
2050 color1,9,5:printtab(19);yo$"
";:color1,12,4:print"S"b2$wh$"-"b3
$;:color1,11,3                        <cg>
2060 print"50 punkte"                 <jc>
2070 color1,7,2:printb2$"links"b2$
"rechts"b2$;:color1,9,4:printyo$      <lm>
2080 color1,9,3:printtab(19)yo$" "
;:color1,12,5:print"Q"b2$wh$"- ";:
color1,11,1                           <ho>
2090 print"1000 punkte"               <eg>
2100 color1,7,0:print" oder joysti
ck 1 ";:color1,9,2:printyo$;:color
1,12,6                                <ne>
2110 print" W"b2$wh$"- ";:color1,1
1,0:print"2000 punkte"                <fh>
2120 co=co+1:ifco=141thenco=0         <ko>
2130 iftn(co)=0thenforl=1to10:next
:goto2150                             <ab>
2140 sound1,tn(co),11                 <ja>
2150 forl=1to90:nextl:geta$:ifa$="
"then2120                             <gg>
2160 printcl$:goto 1720               <ap>
2170 rem                              <je>
2180 tp=150-tp*50                     <ge>
2190 if vk=4 then vk=.1               <il>
2200 if vk=3 then vk=.25              <la>
2210 if vk=2 then vk=.6               <mm>
2220 if vk=1 then vk=.85              <pc>
2230 return                           <al>
2240 goto 2240                        <jc>
2250 rem                              <de>
2260 x$(0)="**"+b5$+"**":x$(1)="**
"+bk$+"V"+b4$+oe$+"**":x$(2)="** "
+bk$+"V"+b3$+oe$+"**"                 <cg>
2270 x$(3)="**"+b2$+bk$+"V"+b2$+oe
$+"**":x$(4)="**"+b3$+bk$+"V "+oe$
+"**"                                 <cb>
```

```
2280 x$(5)="**"+b4$+bk$+"V"+oe$+"*
*"                                  <ob>
2290 x$(6)="** "+wh$+"S"+b3$+oe$+"
**":x$(7)="** "+wh$+"A"+b3$+oe$+"*
*"                                  <io>
2300 x$(8)="**"+b2$+wh$+"Z"+b2$+oe
$+"**"                              <ni>
2310 x$(9)="**"+b3$+wh$+"X "+oe$+"
**":x$(10)="**"+b2$+wh$+"Q"+b2$+oe
$+"**"                              <gn>
2320 x$(11)="** "+wh$+"W"+b2$+oe$+
"**":return                         <ff>
2330 data169,169,306,345            <fl>
2340 data169,169,306,345            <ff>
2350 data169,169,306,345            <pf>
2360 data169,169,306,345            <hm>
2370 data262,262,383,419            <ig>
2380 data262,262,383,419            <la>
2390 data262,262,383,419            <kc>
2400 data262,262,383,419            <jm>
2410 data453,453,543,571            <be>
2420 data453,453,543,571            <bf>
2430 data453,453,543,571            <gc>
2440 data543,516,453,0              <lm>
2450 data704,704,643,596            <id>
2460 data704,704,643,596            <ij>
2470 data704,704,643,596            <fh>
2480 data704,704,643,596            <ek>
2490 data596,596,516,453            <fi>
2500 data596,596,516,453            <bf>
2510 data596,596,516,453            <hm>
2520 data596,596,516,453            <go>
2530 data453,453,345,262            <dj>
2540 data453,453,345,262            <ok>
2550 data453,453,345,262            <nm>
2560 data345,419,453,0              <jc>
2570 data169,169,306,345            <ga>
2580 data169,169,306,345            <gc>
2590 data169,516,453,0              <jl>
2600 data262,262,383,419            <fh>
2610 data262,262,383,419            <ef>
2620 data262,571,516,0              <lh>
2630 data453,453,543,571            <nj>
2640 data453,453,543,571            <da>
2650 data453,685,643,571            <bo>
2660 data453,453,345,262            <lp>
2670 data345,419,453,0              <co>
2680 data50000,amok axel            <ck>
2690 data40000,r=pel rolf           <ok>
2700 data30000,bodo blech           <oo>
2710 data20000,watz windig          <ep>
2720 data10000,mgetrder michi       <co>
2730 data 5000,erpel ernest         <gn>
2740 data 3000,w=rger walter        <gg>
2750 data 1000,rainer rosig         <im>
2760 data169,208,141,17,8,169,80,1
41,20,8,234,162,4,160,0,185,0,208,
153,0,80                            <lh>
2770 data136,208,247,202,240,8,238
,17,8,238,20,8,208,236,96           <de>
2780 data42,00,7e,d5,ab,d5,ab,7e,0
0                                   <je>
2790 data88,00,66,99,19,29,49,f6,0
0                                   <la>
2800 data90,00,66,99,29,29,99,66,0
0                                   <hg>
2810 data65,00,16,69,a9,f9,29,26,0
0                                   <nj>
2820 data83,00,f6,89,e9,19,99,66,0
0                                   <em>
2830 data66,3c,42,db,ff,7e,ff,ff,3
c                                   <ng>
2840 data76,0e,11,15,11,0e,00,00,0
0                                   <ao>
2850 data77,42,63,f7,f5,f9,e9,65,2
2                                   <jb>
2860 data78,44,28,10,10,7c,92,18,1
8                                   <eo>
2870 data79,04,0a,16,7a,72,64,48,3
0                                   <eh>
2880 data86,00,00,ff,ff,42,a5,a5,0
0                                   <bf>
2890 data80,18,18,7e,7e,18,18,3c,7
e                                   <em>
2900 data91,00,00,00,18,24,fe,fe,4
4                                   <kl>
2910 data92,c6,00,7c,fe,c6,fe,c6,0
0                                   <ag>
2920 data61,c6,00,c6,c6,c6,fe,7c,0
0                                   <de>
2930 data1,7c,fe,c6,fe,fe,c6,c6,00  <bj>
2940 data2,fc,fe,c6,fe,c6,fe,fc,00  <gg>
2950 data3,7c,fe,c6,c0,c6,fe,7c,00  <eb>
2960 data4,fc,fe,c6,c6,c6,fe,fc,00  <eg>
2970 data5,7e,fe,c0,f8,c0,fe,7e,00  <jh>
2980 data6,7e,fe,c0,f8,c0,c0,c0,00  <jp>
2990 data7,7c,fe,c0,ce,c6,fe,7c,00  <bm>
3000 data8,ee,ee,ee,fe,ee,ee,ee,00  <pp>
3010 data9,7e,7e,18,18,18,7e,7e,00  <lm>
3020 data10,7e,7e,0c,0c,6c,7c,38,0
0                                   <fn>
3030 data11,c6,cc,d8,f8,f8,cc,c6,0
0                                   <oa>
3040 data12,c0,c0,c0,c0,c0,fe,7e,0
0                                   <ci>
3050 data13,c6,ee,fe,d6,c6,c6,c6,0
0                                   <dd>
3060 data14,c6,e6,f6,fe,de,ce,c6,0
0                                   <jb>
3070 data15,7c,fe,c6,c6,c6,fe,7c,0
0                                   <jf>
3080 data16,fc,fe,66,7e,7c,60,60,0
0                                   <hc>
3090 data17,7c,fe,c6,da,dc,ee,76,0
0                                   <me>
3100 data18,fc,fe,66,7e,7c,6c,66,0
```

```
0                                          <dg>
3110 data19,7e,7e,80,7c,02,fe,fc,0
0                                          <fm>
3120 data20,7e,7e,18,18,18,18,18,0
0                                          <me>
3130 data21,c6,c6,c6,c6,c6,fe,7c,0
0                                          <ei>
3140 data22,c6,c6,c6,c6,6c,7c,38,0
0                                          <ih>
3150 data23,c6,c6,c6,d6,7e,6c,44,0
0                                          <ej>
3160 data24,c6,c6,7c,38,7c,c6,c6,0
0                                          <mc>
3170 data25,c6,c6,ec,7c,38,70,e0,0
0                                          <nm>
3180 data26,fc,fe,1c,38,70,fe,7e,0
0                                          <no>
3190 data48,7c,fe,ce,d6,e6,fe,7c,0
0                                          <pn>
3200 data49,1c,3c,7c,7c,1c,1c,1c,0
0                                          <ho>
3210 data50,7c,fe,ce,1c,38,7e,fe,0
0                                          <oo>
3220 data51,7c,fe,06,7e,06,fe,7c,0
0                                          <fm>
3230 data52,cc,cc,cc,fe,fe,0c,0c,0
0                                          <aa>
3240 data53,fe,fe,c0,fc,06,fe,fc,0
0                                          <ch>
3250 data54,7e,fe,c0,fc,c6,fe,7c,0
0                                          <gk>
3260 data55,fe,fe,0e,1c,38,70,e0,0
0                                          <ee>
3270 data56,7c,fe,c6,38,c6,fe,7c,0
0                                          <bi>
3280 data57,7c,fe,86,fe,7e,06,fc,0
0                                          <mg>
3290 data97,c6,00,7c,fe,c6,fe,7c,0
0                                          <nj>
3300 data256                               <ik>
3310 rem nachspann ==============          <ng>
3320 rem * farbcodes/steuercodes *         <pe>
3330 wh$=chr$(005):c4$=chr$(017)           <gc>
3340 rn$=chr$(018):he$=chr$(019)           <lk>
3350 re$=chr$(028):c3$=chr$(029)           <lf>
3360 oe$=chr$(129):f1$=chr$(130)           <pm>
3370 fo$=chr$(132):f1$=chr$(133)           <bh>
3380 f3$=chr$(134):f5$=chr$(135)           <fh>
3390 f8$=chr$(140):bk$=chr$(144)           <oe>
3400 rf$=chr$(146):c1$=chr$(147)           <hh>
3410 lg$=chr$(153):c1$=chr$(157)           <ll>
3420 rem *** zeichensatz/graphik *         <ej>
3430 z4$=chr$(165):z6$=chr$(167)           <mn>
3440 z8$=chr$(169):zv$=chr$(192)           <mp>
3450 ym$=chr$(219):yn$=chr$(220)           <gm>
3460 yo$=chr$(221):yq$=chr$(223)           <fn>
3470 zz$=chr$(255)                         <ai>
3480 rem ********* zeichenfolgen *         <ck>
3490 for q=1 to 40                         <bn>
3500 qd$=qd$+c4$:qr$=qr$+c3$               <gc>
3510 next q                                <gd>
3520 b$=chr$(32):b2$=b$+b$                 <ap>
3530 b3$=b2$+b$:b4$=b3$+b$                 <bk>
3540 b5$=b4$+b$:b6$=b5$+b$                 <lk>
3550 b$=b5$+b5$:return                     <ng>
3560 forzz=1toxx:printdd$;:next:re
turn                                       <ao>
3570 rem ========================          <ge>
3580 rem 60671 bytes memory                <bb>
3590 rem 10549 bytes program               <gi>
3600 rem 00420 bytes variables             <ld>
3610 rem 00983 bytes arrays                <ca>
3620 rem 05317 bytes strings               <jc>
3630 rem 01024 bytes zeichensatz           <gc>
3640 rem 39167 bytes nicht benutzt         <db>
3650 rem 03211 bytes free (0)              <db>
3660 rem ========================          <lh>
```

Wettlauf auf Leben und Tod

Außer Kontrolle geratene Atomsatelliten gilt es durch schier undurchdringlich erscheinende Meteoritenschwärme zu erreichen und sie im Raumschiff zu bergen. Ein Wettlauf mit der Zeit, denn der Sauerstoffvorrat ist nicht unbegrenzt.

Wir schreiben das Jahr 2001. Auf der Weltraumüberwachungs-Station der NASA herrscht größte Aufregung. Acht fremde Flugkörper im All entpuppen sich als Atomsatelliten, die defekt und unkontrolliert durch das All treiben. Wenn einer von ihnen auf der Erde aufschlüge, gäbe es die größte Katastrophe seit Menschengedenken.
Ihr Auftrag ist es, die defekten Satelliten einzusammeln und sie einzeln in die durch einen schwarzen Strich gekennzeichnete Ladeluke der Raumfähre zu verfrachten. Stoßen Sie mit einem Meteoriten zusammen oder sammeln Sie zwei Satelliten auf einmal ein, verlieren Sie einen Ihrer drei Astronauten. Viel Zeit steht Ihnen nicht zur Verfügung, denn Ihr Sauerstoffvorrat ist begrenzt. Schaffen Sie es, alle Satelliten einzusammeln, erreichen Sie den nächsten Level, in dem die Meteoritenschwärme noch dichter sind.

Hinweise:
1. Sie können die Satelliten nur einzeln in die Ladeluke zurückbringen.
2. Der Astronaut, der bei der Anzeige Lives aufblinkt, ist gerade in Aktion.
3. Steuerung erfolgt über Joystick eins.

Variablenliste:

HI	=	Highscore
HI$	=	Name des Highscore-Halters
NA$	=	Name des Spielers
LE	=	Level
MA	=	Leben
SC	=	Punkte
OX	=	Sauerstoff
MO$	=	Meteoritenband
PU	=	Abfrage für SA
X1–X4	=	Variablen für Spielfeld

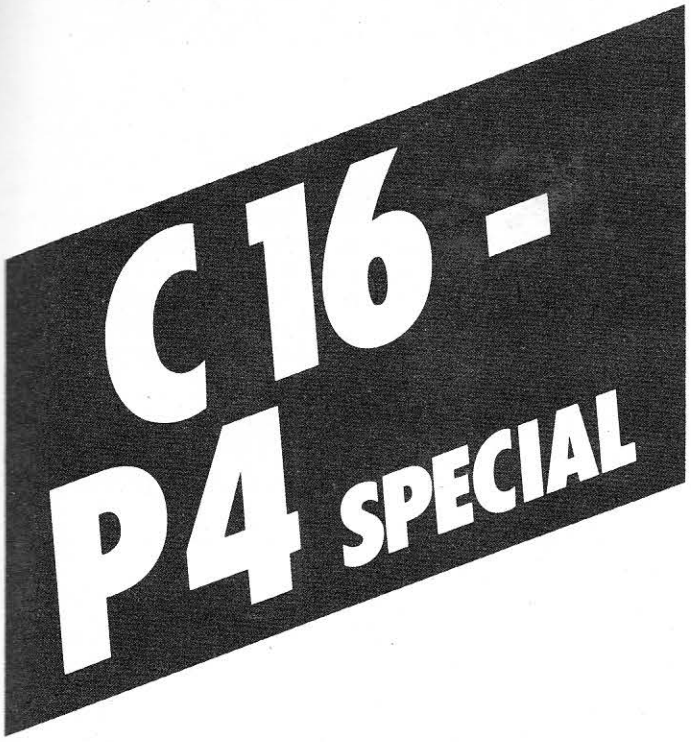
Sonstige Variablen: PR, A, T, I, TH, TL, L.

```
10 rem satellit================c16 <ae>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by martin zucker       <fi>
50 rem                            <pd>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/plus4              <ki>
90 rem ========================== <jg>
100 poke55,0:poke56,56:clr        <bm>
110 gosub2030:scnclr:vol8:hi=999:h
i$="sugar-soft"                   <bg>
120 key1,"1":key2,"2":v=65280     <lh>
130 pokev+18,peek(v+18)and251     <po>
140 pokev+19,peek(v+19)and3or56   <je>
150 fort=832to849:reada:poket,a:pr
=pr+a:nextt                       <ao>
160 sys832                        <hg>
170 fort=14848to15128:reada:ifa>-1
thenpoket,a:pr=pr+a:nextt         <mc>
180 ifpr=19509then420             <mf>
190 print"data-fehler":stop       <hh>
200 data162,0,189,0,208,157,0,56,1
89,0,209,157,0,57,202,208,241,96  <kn>
210 data000,001,007,007,001,007,00
7,001                             <ee>
220 data000,255,255,255,170,255,17
0,255                             <og>
230 data000,000,255,170,170,255,17
0,255                             <km>
240 data000,000,240,252,170,255,17
0,255                             <mm>
250 data000,000,000,000,128,224,22
4,128                             <na>
260 data060,060,016,056,086,016,04
0,108                             <lo>
270 data120,120,255,127,254,254,06
3,254                             <ik>
280 data060,126,255,126,126,060,02
4,255                             <mp>
290 data255,255,001,001,001,001,00
1,001                             <jh>
300 data255,000,255,000,255,000,25
5,000                             <nc>
310 data246,127,113,119,113,125,11
3,127                             <mn>
320 data000,142,255,241,119,145,25
3,241                             <mo>
330 data127,120,096,096,096,096,09
6,096                             <gk>
340 data255,113,000,000,000,000,00
0,000                             <pp>
350 data000,233,137,137,233,041,04
1,239                             <ap>
360 data000,123,074,074,091,074,07
4,122                             <pg>
370 data000,222,082,082,222,088,08
4,082                             <do>
380 data000,059,034,034,058,010,01
0,059                             <ob>
390 data000,221,080,092,080,080,08
0,208                             <dp>
400 data000,240,064,064,064,064,06
4,064                             <di>
410 data-1                        <hg>
420 rem ***************           <em>
430 rem ** titelbild **           <pn>
440 rem ***************           <gl>
450 scnclr                        <gp>
460 char,8,1,ye$+"sugar-soft praes
entiert"+fo$                      <ie>
470 color0,1:color4,14,0:color1,1 <ei>
480 char,3,3,"IIII"+c4$+left$(q1$,
4)+"I"+c4$+c1$+"I"+c4$+c1$+"I"+c4$
+c1$                              <da>
490 print"IIII"+c4$+c1$+"I"+c4$+c1
$+"I"+c4$+c1$+"I"+c4$+left$(q1$,4)
+"IIII"                           <hg>
500 char,9,3,"IIII"+c4$+left$(q1$,
4)+"I"+c3$+c3$+"I"+c4$+left$(q1$,4
)+"I"+c3$+c3$                     <ph>
510 print"I"c4$left$(q1$,4)"I"c3$c
3$"I"c4$left$(q1$,4)"IIII"c4$left$
(q1$,4);                          <ai>
520 print"I"c3$c3$"I"c4$left$(q1$,
4)"I"c3$c3$"I"c4$left$(q1$,4)"I"c3
$c3$"I"                           <dk>
530 char,14,3,"":print"III"c4$c1$c
1$"I"c4$c1$"I"c4$c1$"I"c4$c1$"I"c4
$c1$"I"c4$c1$"I"                  <ia>
540 char,18,3,"III"+c4$+c1$+c1$+c1
$+"I"+c4$+c1$+"III"+c4$+c1$+c1$+c1
$+"I"+c4$+c1$+"III"               <mj>
550 char,22,3,"I"+c4$+c1$+"I"+c4$+
c1$+"I"+c4$+c1$+"I"+c4$+c1$+"I"+c4
$+c1$+"III"                       <an>
560 char,26,3,"I":printc4$c1$"I"c4
$c1$"I"c4$c1$"I"c4$c1$"I"c4$c1$"I"
c4$c1$"III"                       <jp>
570 char,30,3,"I":printc4$c1$"I"c4
$c1$"I"c4$c1$"I"c4$c1$"I"c4$c1$"I"
c4$c1$"I"c4$c1$"I"                <nk>
580 char,32,3,"IIIII":printc4$c1$c
1$c1$"I"c4$c1$"I"c4$c1$"I"c4$c1$"I
"c4$;                             <og>
590 printc1$+"I"+c4$+c1$+"I"+c4$+c
1$+"I"+c4$+c1$+"I"                <hp>
600 char1,17,10,ye$+"NOPQRS"      <lc>
610 color1,2,6:char,9,12,zv$+"ABCD
 = raumfaehre"                    <jp>
620 color1,2,5:char,9,14," E"+b4$+
"= raumfahrer"                    <bf>
630 color1,7,5:char,9,16," G"+b4$+
"= satellit"                      <ec>
640 color1,10,4:char,9,18," F"+b4$
+"= meteorit"                     <go>
```

```
650 char1,8,20,bl$+"high by "+hi$+
":"+str$(hi)                          <hc>
660 char1,5,14,wh$+"JK"+c4$+c1$+c1
$+"LM":char1,29,14,"JK"+c4$+c1$+c1
$+"LM"                                <he>
670 fori=3to37:poke3191-1024+i,3*1
6+6:next:sound1,300,9                 <if>
680 fori=3to37:poke3231-1024+i,4*1
6+6:next:sound1,400,9                 <pj>
690 fori=3to37:poke3271-1024+i,5*1
6+6:next:sound1,500,9                 <ce>
700 fori=3to37:poke3311-1024+i,6*1
6+6:next:sound1,600,9                 <gi>
710 fori=3to37:poke3351-1024+i,7*1
6+6:next:sound1,700,9                 <pg>
720 fori=3to37:poke3391-1024+i,6*1
6+6:next:sound1,600,9                 <ik>
730 fori=3to37:poke3431-1024+i,5*1
6+6:next:sound1,500,9                 <if>
740 fori=3to37:poke3471-1024+i,4*1
6+6:next:sound1,400,9                 <af>
750 fori=3to37:poke3511-1024+i,3*1
6+6:next                              <ca>
760 char,9,22,rn$+ye$+"f1 = spiel"
+b$                                   <jg>
770 char,9,24,rn$+oe$+"f2 = spiela
nleitung "+rf$                        <jl>
780 restore840:vol8                   <en>
790 readth,tl:tl=tl*15:ifth=-1then
780                                   <od>
800 sound1,th,tl:sound2,th-7,tl       <in>
810 geta$:ifa$="1"ora$="2"then960     <ko>
820 fori=1to50:nexti                  <ml>
830 goto790                           <nh>
840 data596,1,704,1,739,1             <dl>
850 data770,5,596,1,704,1             <pd>
860 data739,1,770,5,810,1             <ek>
870 data810,1,810,1,810,2             <om>
880 data770,2,704,2,770,2             <md>
890 data739,5,770,1                   <he>
900 data739,1,704,3,704,1             <kg>
910 data770,2,810,1,810,1             <ki>
920 data810,1,779,5,810,1             <ba>
930 data834,1,810,2,770,2             <gi>
940 data704,2,739,2,704,4             <fj>
950 data-1,-1,-1                      <gf>
960 ifa$="2"then1910                  <om>
970 char,1,20,b$+b$+b$+b3$+b3$        <mh>
980 char,10,20,bl$+"dein name":inp
utna$                                 <gm>
990 le=1 :ma=3:sc=0                   <jf>
1000 rem ***************              <fn>
1010 rem ** spielfeld **              <ig>
1020 rem ***************              <ch>
1030 printcl$:color1,2                <ba>
1040 fori=0to24:char,0,i,rn$+"HHHH
HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
HH"+rf$:nexti                         <ah>
1050 fori=1to19:char,0,i,rn$+"H"+r
f$+b$+b$+b$+b4$+b4$+rf$:next          <bg>
1060 char,0,24,rn$+"HHHHHHHHHHHHHH
HHHHHHHHHHHHHHHHHHHHHHHHHHHH"+rf$     <lj>
1070 color1,2,6:char,13,18,zv$+"AB
CD"                                   <pb>
1080 char,2,21,g1$+"score: "+str$(
sc)                                   <jb>
1090 char,27,21,"level: "+str$(le)    <de>
1100 ifma=3thenchar,2,23,"lives: E
E"+f1$+"E"+fo$+" "                    <op>
1110 ifma=2thenchar,2,23,"lives: E
"+f1$+"E"+fo$+b2$                     <no>
1120 ifma=1thenchar,2,23,"lives: "
+f1$+"E"+fo$+b3$                      <db>
1130 char,27,23,"oxygen: 500"         <fl>
1140 fori=1tole*10                    <pe>
1150 x1=int(rnd(1)*797)+1             <dh>
1160 x2=x1+3113                       <ph>
1170 ifpeek(x2)<>32andpeek(x2)<>96
then1150                              <pg>
1180 pokex2,70:pokex2-1024,4*16+9     <bh>
1190 nexti                            <pi>
1200 forl=1to8                        <oj>
1210 x3=int(rnd(1)*797)+1             <ld>
1220 x4=x3+3113                       <gb>
1230 data704,685,2,739,643,2,704,7
04,4                                  <nk>
1240 data-1,-1,-1                     <be>
1250 ifpeek(x4)<>32andpeek(x4)<>96
then1210                              <hd>
1260 ifx4>3272andx4<3311then1210      <af>
1270 ifx4>3632andx4<3671then1210      <ea>
1280 pokex4,71:pokex4-1024,5*16+6     <cb>
1290 nextl                            <in>
1300 rem *****************            <kn>
1310 rem * hauptschleife *            <hd>
1320 rem *****************            <dn>
1330 a=3767:sa=0:pu=0:me=1:ox=500     <nn>
1340 color1,10,4                      <pj>
1350 printhe$left$(qd$,5)c3$;mid$(
mo$,me,38)                            <kc>
1360 printhe$left$(qd$,14)c3$;mid$
(mo$,40-me,38)                        <md>
1370 ifme=38thenme=1                  <lp>
1380 me=me+1                          <jp>
1390 char1,34,23,g1$+str$(ox)+wh$+
rn$+"H"+rf$:ox=ox-1:ifox=0then1680     <go>
1400 j=joy(1):ifj<>0thenpokea,32      <kn>
1410 ifj=1thena=a-40                  <lp>
1420 ifj=3thena=a+1                   <ej>
1430 ifj=5thena=a+40                  <cb>
1440 ifj=7thena=a-1                   <kc>
1450 ifpeek(a)<>32andpeek(a)<>69th
en1490                                <eo>
1460 pokea-1024,5*16+1                <gg>
1470 pokea,69:sound1,500,1:sound2,
404,1                                 <fp>
```

```
1480 goto1340                                  <cm>
1490 rem **********                            <pc>
1500 rem * punkt ?*                            <dp>
1510 rem **********                            <bc>
1520 ifpeek(a)=200orpeek(a)=70then
1680                                           <an>
1530 ifpeek(a)=66orpeek(a)<>71then
goto1590                                       <jm>
1540 pokea,69                                  <bk>
1550 pu=pu+1:ifpu=2then1680                    <pm>
1560 sc=sc+25:char,9,21,g1$+str$(s
c)                                             <ac>
1570 fori=200to600step50:sound1,i,
1:nexti                                        <cc>
1580 goto1340                                  <mf>
1590 rem *************                         <fl>
1600 rem * rueckkehr *                         <kc>
1610 rem *************                         <nh>
1620 ifpeek(a)<>66then1680                     <gf>
1630 ifpu=1thensc=sc+50:ifpu=1then
sa=sa+1                                        <mo>
1640 fori=200to600step100:sound1,i
,2:next                                        <kb>
1650 char,9,21,g1$+str$(sc)                    <kc>
1660 ifsa=8thengoto1770                        <ia>
1670 a=a-40:pu=0:goto1340                      <lp>
1680 rem ******************                    <bo>
1690 rem * zusammenstoss *                     <do>
1700 rem ******************                    <oo>
1710 ifpeek(a)=71thensa=sa+1:pokea
,69                                            <nh>
1720 fori=1to126:sound3,i*8,1                  <im>
1730 pokea-1024,i:nexti                        <dl>
1740 ma=ma-1                                   <nn>
1750 ifma=0then1860                            <fi>
1760 goto1000                                  <do>
1770 rem ********************                  <cd>
1780 rem * naechstes level *                   <gh>
1790 rem ********************                  <gd>
1800 char,13,10,ye$+"bonus:":fori=
oxto0step-1                                    <mh>
1810 char1,18,10,str$(i)+" ":sc=sc
+1                                             <pm>
1820 char,9,21,g1$+str$(sc):sound1
,i+500,1:next                                  <cj>
1830 le=le+1:char,13,10,lg$+"level
"+str$(le)+b2$                                 <np>
1840 fori=1to125step2:forl=1to7:po
ke3484-1024+l,i:nextl:nexti                    <ik>
1850 goto1000                                  <il>
1860 char,13,10,ye$+"game over"                <pl>
1870 ifsc<=hithen1890                          <kd>
1880 hi=sc:hi$=na$                             <ca>
1890 fori=1to2000:next                         <jf>
1900 goto420                                   <nm>
1910 scnclr:printbl$b5$b4$rn$"spie
lanleitung"rf$                                 <jb>
1920 printc4$ye$b4$"sie haben den
auftrag,defekte "                              <fe>
1930 printc4$b4$"satelliten einzus
ammeln und sie"                                <bb>
1940 printc4$b4$"einzeln in die la
deluke(gekenn-"                                <ca>
1950 printc4$b4$"zeichnet durch sc
hwarzen strich)"                               <he>
1960 printc4$b4$"der raumfaehre zu
rueckzubringen."                               <lo>
1970 printc4$b4$"stossen sie gegen
 einen meteorit,"                              <kb>
1980 printc4$b4$"oder sammeln sie
2 satelliten"                                  <oh>
1990 printc4$b4$"aufeinmal ein,ver
lieren sie "                                   <gb>
2000 printc4$b4$"einen ihrer 3 ast
ronauten."                                     <cf>
2010 printc4$c4$c4$b4$b4$oe$rn$b3$
"f1 = intro"b3$rf$                             <nk>
2020 geto$:ifa$="1"then420:else202
0                                              <co>
2030 b$=chr$(32):b2$=b$+b$                     <aj>
2040 b3$=b2$+b$:b4$=b3$+b$                     <bn>
2050 b5$=b4$+b$:b$=b5$+b5$                     <di>
2060 fori=1to26:mo$=mo$+"F"+b2$:ne
xt                                             <bk>
2070 rem nachspann ---------------             <jk>
2080 rem * farbcodes/steuercodes *             <kc>
2090 wh$=chr$(005):c4$=chr$(017)               <pd>
2100 rn$=chr$(018):he$=chr$(019)               <ee>
2110 c3$=chr$(029):bl$=chr$(031)               <pl>
2120 oe$=chr$(129):fl$=chr$(130)               <hb>
2130 fo$=chr$(132):rf$=chr$(146)               <me>
2140 cl$=chr$(147):g1$=chr$(151)               <nn>
2150 lg$=chr$(153):c1$=chr$(157)               <nj>
2160 ye$=chr$(158)                             <jh>
2170 rem *** zeichensatz/graphik *             <ba>
2180 zv$=chr$(192)                             <eo>
2190 rem ********* zeichenfolgen *             <oa>
2200 for q=1 to 40                             <kg>
2210 qd$=qd$+c4$:ql$=ql$+c1$                   <cl>
2220 next q                                    <ci>
2230 return                                    <al>
2240 rem ========================              <ml>
2250 rem 12277 bytes memory                    <jl>
2260 rem 06986 bytes program                   <ld>
2270 rem 00336 bytes variables                 <bh>
2280 rem 00000 bytes arrays                    <pk>
2290 rem 00574 bytes strings                   <fe>
2300 rem 02048 bytes zeichensatz               <mf>
2310 rem 02333 bytes free (0)                  <bj>
2320 rem ========================              <ma>
```

*



KOMMT REGELMÄSSIG ZU IHNEN INS HAUS

Finden Sie Ihr C16/P4 nicht am Kiosk? Weil es schon ausverkauft ist? Oder „Ihr" Kiosk nicht beliefert wurde? Kein Problem! Für ganze 70 DM liefern wir Ihnen per Post sechs Hefte ins Haus (Ausland 80 DM). Einfach den Bestellschein ausschneiden – fotokopieren oder abschreiben, in einen Briefumschlag und ab per Post (Achtung: Porto nicht vergessen). C16/P4-SPECIAL kommt dann pünktlich ins Haus.

WICHTIGE RECHTLICHE GARANTIE!
Sie können diesen Abo-Auftrag binnen einer Woche nach Eingang der Abo-Bestätigung durch den Verlag widerrufen – Postkarte genügt. Zur Wahrung der Frist genügt die rechtzeitige Absendung. Ansonsten läuft dieser Auftrag jeweils für sechs Ausgaben, wenn ihm nicht vier Wochen vor Ablauf widersprochen wird, weiter.

DAS SONDERANGEBOT: PRIVATE KLEINANZEIGEN KOSTENLOS!

Das bietet Ihnen COMMODORE-WELT: KLEINANZEIGEN SIND KOSTENLOSE FÜR PRIVATANBIETER! Suchen Sie etwas, haben Sie etwas zu verkaufen, zu tauschen, wollen Sie einen Club gründen? Coupon ausfüllen, auf Postkarte kleben oder in Briefumschlag stecken und abschicken. So einfach geht das. Wollen Sie das Heft nicht zerschneiden, können Sie den Coupon auch fotokopieren. Oder einfach den Anzeigentext uns so schicken, auf Postkarte oder im Brief. Aber bitte mit Druckbuchstaben oder in Schreibmaschinenschrift!
Und: Einschließlich Ihrer Adresse und/oder Telefonnummer sollten acht Zeilen à 28 Anschläge nicht überschritten werden.
ACHTUNG: WICHTIGER HINWEIS!
Wir veröffentlichen nur Kleinanzeigen privater Inserenten kostenlos. Gewerbliche Anzeigen kosten pro Zeile 4,80 plus Mehrwertsteuer!
Wir versenden für Privat-Inserenten keine Beleg-Exemplare!
Chiffre-Anzeigen sind nicht gestattet! Wir behalten uns vor, Anzeigen, die gegen rechtliche, sittliche oder sonstige Gebote verstoßen, abzulehnen!
Anzeigenabdruck in der Reihenfolge ihres Eingangs, kein Rechtsanspruch auf den Abdruck in der nächsten Ausgabe!
Die Insertion ist nicht vom Kauf des Heftes abhängig! Wir behalten uns vor, Anzeigen, die nicht zum Themenkreis des Heftes – Computer – gehören, nicht abzudrucken oder sie nur insoweit zu berücksichtigen, wie es der Umfang des kostenlosen Anzeigenteils zuläßt.

ANZEIGENSERVICE

Die große Börse für jeden Zweck in der CBM REVUE / COMMODORE-WELT. Kostenlos für Privat-Inserenten. Spottbillig für gewerbliche Anbieter. Einfach Coupon ausschneiden, fotokopieren o.ä., ausfüllen und ab die Post – Freimachen nicht vergessen! – Unsere Adresse steht auf dem Coupon, ebenso die Preise für gewerbliche Anbieter! Achtung! Wir weisen ausdrücklich darauf hin, daß wir offensichtlich gewerbliche Anzeigen nicht kostenlos veröffentlichen und uns jedweden Abdruck kostenloser Anzeigen vorbehalten müssen, insbesondere, wenn deren Inhalt nicht CBM-typisch ist oder gegen geltendes Recht verstößt. Private Chiffreanzeigen werden nicht aufgenommen. Für gewerbliche Anbieter DM 4,80 pro Zeile zuzüglich gesetzlicher Mehrwertsteuer.

COMMODORE WELT
ANZEIGENABTLG
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

Name _____
Vorname _____
Straße/Hausnr. _____
Plz/Ort _____

ABO-SERVICE-KARTE

16/II **COUPON**

Ich nehme zur Kenntnis, daß die Belieferung erst beginnt, wenn die Abo-Gebühr dem Verlag zugegangen ist.

Ja, ich möchte von Ihrem Angebot Gebrauch machen.
Bitte senden Sie mir bis auf Widerruf ab sofort jeweils die nächsten sechs Ausgaben an untenstehende Anschrift. Wenn ich nicht vier Wochen vor Ablauf kündige, läuft diese Abmachung automatisch weiter.

Name Buck **16/II**
Vorname Ingo
Straße/Hausnr. Danziger Ring 3
Plz/Ort 2740 Bremervörde

Ich bezahle:
☐ per beiliegendem Verrechnungsscheck
☐ gegen Rechnung
☐ bargeldlos per Bankeinzug von meinem Konto

bei (Bank) und Ort ~~Darein~~ Westbank Bremervörde
Kontonummer 57/235903
Bankleitzahl 20730157
(steht auf jedem Kontoauszug)
Unterschrift _____

Von meinem Widerspruchsrecht habe ich Kenntnis genommen.
Unterschrift _____

COMMODORE WELT
ABO-SERVICE **16/II**
POSTFACH 1161
D-8044
UNTERSCHLEISSHEIM

PROGRAMMSERVICE

Hiermit bestelle ich in Kenntnis Ihrer Verkaufsbedingungen die Listings dieses Heftes auf

☐ **Kassetten zu 40,-** ☐ **Disketten zu 40,- (16er)**

Ich zahle:
Zutreffendes bitte ankreuzen!
per beigefügtem Scheck ( ) Schein ( )

Gegen Bankabbuchung am Versandtag ( )

Meine Bank (mit Ortsname) _____ **16/II**

Meine Kontonummer _____

Meine Bankleitzahl _____ (steht auf jedem Bankauszug) _____

Vorname _____ Nachname _____

Str./Nr. _____ Plz/Ort _____

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Keine Nachnahme. Umtausch bei Nichtfunktionieren. 16/II

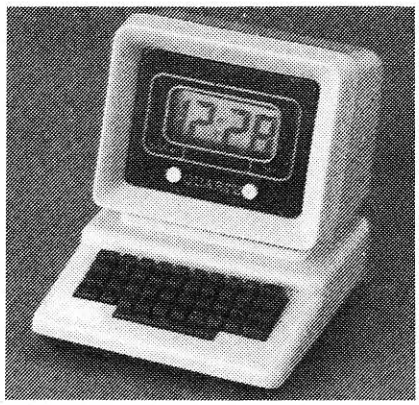
Unterschrift _____

Bitte ausschneiden und einsenden an

COMMODORE WELT
KASSETTENSERVICE **16/II**
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

LESER WERBEN LESER

GEWINNEN SIE EINE COMPUTER-UHR! Und zusätzlich eventuell noch ein großes Commodore-Buch. Oder ein Paket Disketten. ODER AUCH EINEN COMMODORE-DRUCKER – ODER EINE DISKETTENSTATION! Wie? Sie werben einen Abonnenten. Dann haben Sie auf jeden Fall schon die Computer-Uhr gewonnen. Zusätzlich verlosen wir unter allen, die mitmachen, jeden Monat vier weitere wertvolle Preise. Und alle sechs Monate gibt es einen Hauptpreis unter allen Abo-Werbern zu gewinnen. Also: Mitmachen. Mitgewinnen.



Herrn/Frau _____

_____

Straße/Hausnr. _____

_____

Plz/Ort _____

_____

Ja, ich mache mit beim Abo-Wettbewerb. Ich habe als neuen Abonnenten der COMMODORE WELT geworben.

Der neue Abonnent war bisher noch nicht Bezieher dieser Zeitschrift.
Als Prämie erhalte ich nach Eingang des Abo-Entgeltes auf jeden Fall eine Computer-Uhr, wie abgebildet, und nehme zusätzlich noch an der Verlosung des Monats sowie der halbjährlichen Hauptpreise teil. Mir ist bekannt, daß der Rechtsweg bei den Verlosungen ausgeschlossen ist.
Meinen Preis senden Sie an

Name _____

Straße/Hsnr. _____

Plz/Ort _____

(Bitte ausschneiden und zusammen mit der Abo-Bestellkarte links einsenden!) **16/II**

VERDIENEN SIE GELD MIT IHREM COMPUTER!

Haben Sie einen Commodore VC 20 oder C 64? Einen 16/116, Plus 4? Oder einen 128? Können Sie programmieren? In Basic oder Maschinensprache? Dann bietet COMMODORE-WELT Ihnen die Möglichkeit, mit diesem Hobby Geld zu verdienen!

Wie? Ganz einfach. Sie senden uns die Programme, die Sie für einen Abdruck als geeignet halten, zusammen mit einer Kurzbeschreibung, aus der auch die verwendete Hardware – eventuelle Erweiterungen – benutzte Peripherie – hervorgehen muß (Schauen Sie sich dazu den Kopf unserer Programmlistings an.)

Benötigt werden: Zwei Listings des Programms sowie eine Datenkassette oder Diskette! Wenn die Redaktion sich überzeugt hat, daß dieses Programm läuft und sich zum Abdruck eignet, zahlen wir Ihnen pro Programm je nach Umfang bis zu DM 300,–!

Sollten Sie keinen Drucker haben, genügt der Datenträger.

Sie erhalten Ihre Kassette/Diskette selbstverständlich zurück, wenn Sie einen ausreichend frankierten Rückumschlag mit Ihrer Adresse beifügen.

Bei der Einsendung müssen Sie mit Ihrer Unterschrift garantieren, daß Sie der alleinige Inhaber der Urheber-Rechte sind! Benutzen Sie bitte anhängendes Formular! (Wir weisen darauf hin, daß auch die Redaktion amerikanische und englische Fachzeitschriften liest und „umgestaltete" Programme ziemlich schnell erkennt).

Um Ihnen die Arbeit zu erleichtern, finden Sie hier ein Formular. Sie können es ausschneiden oder fotokopieren.

Name des Einsenders: _____

Straße/Hausnr./Tel.: _____

Plz/Ort: _____

Hiermit biete ich Ihnen zum Abdruck folgende(s) Programm(e) an:

_____

_____

_____

_____

Benötigte Geräte: _____

_____

Beigefügt ( ) Listings ( ) Kassette ( ) Diskette

Ich versichere, der alleinige Urheber des Programmes zu sein!
Hiermit ermächtige ich die Redaktion, dieses Programm abzudrucken und wirtschaftlich zu verwerten. Sollte es in den Kassetten-Service aufgenommen werden, erhalte ich auch dafür eine entsprechende Vergütung, das Copyright geht insoweit auf den Verlag über.

_____

Rechtsverbindliche Unterschrift

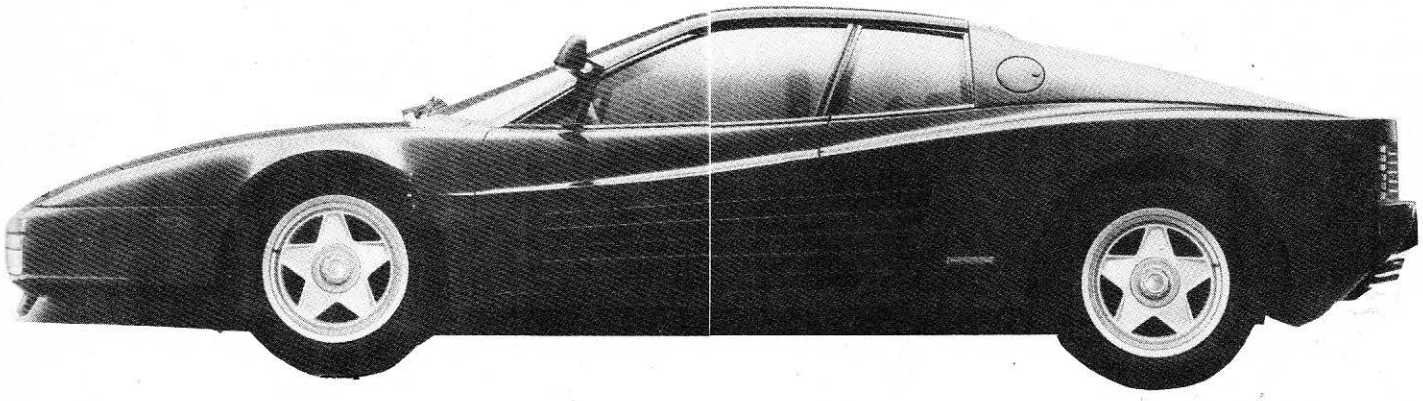
COMMODORE WELT
PROGRAMM-REDAKTION
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

Auto Dat

Auto-Dat. sammelt alle Daten, die bei einem Fahrzeug anfallen und wertet sie aus. So wird der Verbrauch sowie die Kosten pro Kilometer angezeigt. Das Programm ist in allen Funktionen menügesteuert. Fehler werden weitgehend abgefangen.

ZUM PROGRAMM

Beim Starten des Programms wird nach dem Titelbild nachgefragt, ob schon eine Datei auf der Diskette angelegt wurde. Danach wird nach dem Jahrgang gefragt. Dies ist später beim Abspeichern einer Datei wichtig, da die Daten nach der Jahreszahl abgespeichert werden. Das Programm ist 35 KByte lang. Es können bis zu 50 Dateien gespeichert werden. Pro Datei sind 150 Einträge möglich.



ZUM MENÜ

Es besteht aus elf Punkten.

1. Laden einer Datei

Man kann sich jede gewünschte Datei einladen.

2. Speichern einer Datei

Die im Speicher befindliche Datei wird unter der aktuellen Jahreszahl abgespeichert.

3. Daten eingeben

a) Datum in	Tag.Mon	01.01
b) km/stand	Tachostand	133000
c) Liter/Ben.	Benzin	56.6
d) DM/Ben.	Betrag	61.7
e) Öl/DM	Betrag	23.2
f) Reparaturen	Rechnungsbetr.	601.6

Zu diesem Punkt kann man noch einen kleinen Text eingeben.

g) Sonstiges Versicherungen 400.6
Auch hier ist eine kleine Texteingabe möglich.

4. Ändern

In diesem Punkt kann man alle Daten verändern, löschen oder sogar einen Datenblock einfügen.

5. Ausgabe

Hier werden alle Datenblöcke aufgelistet. Es besteht auch die Möglichkeit, sich einen bestimmten Datenblock anzusehen.

6. und 7. Tabelle 1 und 2

Hier werden in Tabellenform alle Daten angezeigt.

8. Gesamt-Abrechnung

In dem Menüpunkt werden alle Daten zusammengefaßt und ausgewertet.

9. Löschen

Die Funktion Löschen löscht alle Daten!

10. Datei löschen

Es werden bestimmte Dateien auf Diskette und im Directoryeintrag gelöscht.

11. Programm beenden

Wenn eine Datei vorhanden ist und man das Programm verlassen will, wird der Directory-Eintrag gesichert. Anschließend wird das Programm verlassen. Man kann aber das Programm mit F6 ohne Datenverlust wieder starten.

Leider hatte ich keinen Drucker zur Verfügung. Aber es sind die dafür vorgesehenen Stellen durch REM-Zeilen markiert. □

```
10 rem auto datei==============p4 <on>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by klaus dieter breuer <mh>
50 rem        grevenbroich        <gf>
60 rem basic v3.5                 <od>
70 rem plus4 (c16/116 + 64 kb)    <cd>
80 rem floppy                     <ak>
90 rem ========================== <jg>
100 gosub5090                     <ga>
110 bb=1                          <io>
120 dim dt$(50)                   <hi>
130 gosub550                      <ad>
140 trap3730                      <bd>
150 printc4$c4$c4$:printb2$"ist sc
hon eine datei"                   <cf>
160 printc4$b2$"vorhanden (j/n) ";
:inputa$                          <mn>
170 ifa$="n"thengoto200           <pc>
180 ifa$="j"thengosub4380 :goto200 <om>
190 goto130                       <il>
200 printhe$left$(qd$,13):printb3$
;:input"jahreszahl";ja$           <of>
210 ifa$="j"thenv=1:ifa$="n"thenv=
0                                 <og>
220 ifja$=""then200               <fi>
230 forqq=1tobb                   <hp>
240 ifdt$(qq)=ja$thengoto280      <je>
250 nextqq                        <mp>
260 dt$(bb)=ja$                   <ob>
270 bb=bb+1                       <am>
280 key6,"goto 860"+chr$(13)      <hb>
290 dim da$(150,7)                <mh>
300 dim d(150,7),zz(150),i(150)   <cc>
310 dim z(150),pt$(12),s(150)     <hl>
320 zx=134000                     <hj>
330 goto540                       <ge>
340 pt$(1)=b3$+"datei laden"+b3$  <ja>
350 pt$(2)=" datei speichern "    <ok>
360 pt$(3)="auto dat.eingeben"    <md>
370 pt$(4)="auto dat.aendern "    <hd>
380 pt$(5)="auto dat.ausgeben"    <pj>
390 pt$(6)=b2$+"auto tabelle 1 "  <he>
400 pt$(7)=b2$+"auto tabelle 2 "  <jm>
410 pt$(8)="gesamt abrechnung"    <ok>
420 pt$(9)="programm loeschen"    <pa>
430 pt$(10)=" datei loeschen"+b2$ <mc>
440 pt$(11)="programm beenden "   <ab>
450 f$(1)=b2$+"datum"+b6$         <ml>
460 f$(2)=b2$+"km/stand"+b3$      <if>
470 f$(3)=b2$+"liter/ben. "       <ao>
480 f$(4)=b2$+"dm/ben."+b4$       <kc>
490 f$(5)=b2$+"dm/oel"+b5$        <lc>
500 f$(6)=b2$+"reparatur"+b2$     <mm>
510 f$(7)=b2$+"sonstiges"+b2$     <jh>
520 f$(8)=b2$+"gefahrne km "      <fa>
530 return                        <lg>
540 goto 860                      <kl>
550 rem ========================= <pp>
560 rem  programmkopf             <mm>
570 rem ========================= <mo>
580 print cl$                     <pc>
590 for i=1 to 40:print"=";:next  <jn>
600 printb5$"auto dat.verwaltung"b
3$"jahr ";ja$                     <bb>
610 for i=1 to 40:print"=";:next  <ip>
620 print:print                   <ob>
630 return                        <ho>
640 rem ========================= <np>
650 rem  fehlermeldung            <mb>
660 rem ========================= <nj>
670 char,1,24,fe$                 <ec>
680 for i=1to1000:next            <ho>
690 char,1,24,b$+b$+b3$           <hn>
700 return                        <al>
710 rem ========================= <ne>
720 rem  koepfe programmteile     <li>
730 rem ========================= <hm>
740 gosub 550:gosub340            <hf>
750 char,10,5,"******************* <ff>
760 char,10,6,"*"+pt$(f)+"*"      <ic>
770 char,10,7,"******************* <ck>
780 return                        <km>
790 rem ========================= <lo>
800 rem  daten im recher          <mg>
810 rem ========================= <kn>
820 fe$=b2$+"keine daten im rechne
r!"                               <mh>
830 gosub 640                     <oj>
840 return                        <ce>
850 next y                        <lb>
860 gosub5340:gosub550            <op>
870 printb5$"programmfunktion:"   <ng>
880 printb5$"-----------------"   <hb>
890 print                         <jd>
900 printb6$"- 1 - datei"b4$"laden
"                                 <dh>
910 printb6$"- 2 - datei"b4$"speic
hern"                             <eh>
920 printb6$"- 3 - auto dat.eingeb
en"                               <dn>
930 printb6$"- 4 - auto dat.aender
n"                                <ch>
940 printb6$"- 5 - auto dat.ausgeb
en"                               <ig>
950 printb6$"- 6 - auto"b5$"tabell
e 1"                              <oa>
960 printb6$"- 7 - auto"b5$"tabell
e 2"                              <oo>
970 printb6$"- 8 - gesamt"b3$"abre
chnung"                           <ne>
980 printb6$"- 9 - daten"b4$"loesc
hen"                              <dl>
990 printb6$"-10 - datei"b4$"loesc
hen"                              <dk>
1000 printb6$"-11 - programm beend
```

```
en"                                          <im>
1010 print:print:f=0                         <gm>
1020 printb5$;:input"auswahl (1-11
): ";f                                       <oh>
1030 if f=0 or f>11then fe$=b2$+"u
ngueltiger wert!":gosub 640:goto 8
60                                           <ja>
1040 on f goto 1050,1330,1580,1840
,2170,2760,3030,3280,3860,3990,257
0                                            <ni>
1050 rem ======================== <mh>
1060 rem  datei laden                        <fc>
1070 rem ======================== <jo>
1080 trap3730                                <ab>
1090 ifv=0thengosub4660                      <lg>
1100 gosub 710:print:print                   <mn>
1110 printb2$"bitte diskette einle
gen"                                         <fo>
1120 print                                   <gb>
1130 print                                   <hf>
1140 print:ba$=ja$                           <pc>
1150 printb2$;:input"druecken sie
danach return";x$                            <gn>
1160 printcl$:gosub4490                      <pe>
1170 printc4$c4$c3$c3$"welche date
i soll eingelesen"                           <gn>
1180 printc4$c3$" jahrgang ";:inpu
tba$                                         <lh>
1190 open 1,8,14,ba$+",s,r"                  <fk>
1200 input#1,z                               <oc>
1210 for y=1to z                             <fi>
1220 for i=1to 7                             <kh>
1230 input#1,d(y,i),da$(y,i)                 <ni>
1240 next i                                  <lp>
1250 next y                                  <ph>
1260 input#1,ja$                             <oi>
1270 close 1                                 <fm>
1280 gosub 3600 :close15                     <ij>
1290 ifz=0thengosub790:ifz=0then86
0                                            <am>
1300 print:printb2$"daten sind gel
aden!"                                       <op>
1310 for x=1to 2000:next                     <if>
1320 goto 860                                <jk>
1330 rem ======================== <cb>
1340 rem  datei speichern                    <pg>
1350 rem ======================== <ei>
1360 gosub 710:print:print                   <mi>
1370 if z=0 then gosub 790:goto 86
0                                            <na>
1380 printb2$"bitte diskette einle
gen"                                         <fk>
1390 print:f=2                               <ha>
1400 trap3730                                <cj>
1410 print:ba$="@:"+ja$                      <go>
1420 print                                   <lm>
1430 printb2$;:input"druecken sie
danach return";x$                            <bf>
1440 gosub 3600 :close15                     <na>
1450 open 1,8,14,ba$+",s,w"                  <lj>
1460 print#1,z                               <mp>
1470 for y=1to z                             <ik>
1480 for i=1to 7                             <pc>
1490 print#1,d(y,i);",";da$(y,i)             <jf>
1500 next i                                  <nd>
1510 next y                                  <al>
1520 print#1,ja$                             <le>
1530 close 1                                 <ha>
1540 gosub4270                               <bo>
1550 print:printb2$"daten sind ges
ichert!"                                     <fb>
1560 for x=1to 2000:next                     <dm>
1570 goto 860                                <ek>
1580 rem ======================== <jo>
1590 rem  auto dat.eingeben                  <fh>
1600 rem ======================== <nb>
1610 gosub 710                               <ni>
1620 z=z+1                                   <ge>
1630 print:print                             <jc>
1640 for i=1to 7                             <ak>
1650 print f$(i);                            <fp>
1660 ifi=6ori=7 thengoto1680:elsei
nput d(z,i)                                  <ip>
1670 goto1690                                <ck>
1680 inputd(z,i): printc2$left$(qr
$,22)"text ";:inputda$(z,i)                  <bn>
1690 next                                    <md>
1700 print" nr."z                            <lk>
1710 print" daten richtig eingegeb
en (j/n)";                                   <ef>
1720 x$="":input x$                          <hh>
1730 if x$="j"then 1770                      <hl>
1740 if x$="n"then z=z-1:goto 1580           <eo>
1750 printchr$(145)b$b$b$b4$b3$              <ba>
1760 printchr$(145);:goto1710                <na>
1770 print:print                             <pf>
1780 print" weitere eingaben (j/n)
";                                           <gh>
1790 x$="":input x$                          <nn>
1800 if x$="j"then 1580                      <mj>
1810 if x$="n"then 860                       <nd>
1820 printchr$(145)b$b$b$b4$                 <mc>
1830 printchr$(145);:goto1780                <me>
1840 rem ======================== <cb>
1850 rem  auto dat.aendern                   <ce>
1860 rem ======================== <ei>
1870 zz=1                                    <bn>
1880 gosub 710                               <mj>
1890 print                                   <gl>
1900 if z=0 then gosub 790:goto 86
0                                            <ll>
1910 for i=1to7                              <mg>
1920 print i;f$(i);d(zz,i);da$(zz,
i)                                           <kh>
1930 next                                    <kf>
1940 print:print" nr."z;b2$"pl."zz           <ce>
```

```
1950 print:print " "rn$"v"rf$"orwa
erts, "rn$"r"rf$"ueckwaerts,";      <ji>
1960 print" "rn$"a"rf$"endern, "rn
$"m"rf$"enue"                       <kk>
1970 print" "rn$"e"rf$"infuegen od
er "rn$"l"rf$"oeschen von daten"    <nl>
1980 print                          <ca>
1990 get x$:if x$=""then goto 1990  <dc>
2000 if x$="m"then 860              <hn>
2010 if x$="v"and zz<z then zz=zz+
1:goto 1880                         <me>
2020 if x$="r"and zz>1 then zz=zz-
1:goto 1880                         <il>
2030 if x$="a"then goto 2070        <nh>
2040 if x$="l"then goto 4790        <lk>
2050 if x$="e"then goto 4900        <ia>
2060 fe$=" eingabefehler!":gosub 6
40:goto 1990                        <dc>
2070 char,1,21,""                   <jp>
2080 input" feldnummer(1-7)";x      <mi>
2090 if x<1 or x>7 then print chr$
(145);:goto 2080                    <gb>
2100 print                          <bb>
2110 ifx=6orx=7thengoto2140         <ab>
2120 input " neuer inhalt:";d(zz,x
)                                   <pa>
2130 goto2160                       <mg>
2140 input " neuer inhalt:";d(zz,x
)                                   <oc>
2150 printc2$left$(qr$,25)"text ";
:inputda$(zz,x)                     <ip>
2160 goto 1880                      <ji>
2170 rem ========================== <fb>
2180 rem  auto dat.ausgeben         <fc>
2190 rem ========================== <eo>
2200 rem  gosub 730:print:print     <ed>
2210 if z=0 then gosub 790:goto 86
0                                   <pf>
2220 rem input;c4$" drucker oder b
ildschirm(d/b)";g$                  <lh>
2230 rem if g$="d"then open 1,4     <pd>
2240 gosub 710:print:print          <ch>
2250 print" suchbegriffe:"          <gd>
2260 print" -------------"          <lm>
2270 char,3,22," gesuchtes datum e
ingeben"                            <go>
2280 char,3,23," oder return druec
ken.":char,0,10,""                  <gp>
2290 print                          <io>
2300 ss =0                          <lj>
2310 print f$(1);                   <im>
2320 input ss                       <ni>
2330 for y=1to z                    <ic>
2340 s=0                            <ek>
2350 if ss=0 or ss=d(y,1)then s=s+
1                                   <kg>
2360 if ss=0orss=d(y,1)then2370:el
segoto2510                          <lm>
2370 rem  if g$="d" then print#1    <kp>
2380 gosub 710:print:print:ss=0     <de>
2390 for i=1to7                     <en>
2400 if g$=""then print f$(i);d(y,
i);da$(y,i)                         <ij>
2410 rem if g$="d"then print#1,f$(
i);d(y,i;da$(y,i)                   <ei>
2420 next i                         <eb>
2430 print:print" nr."y             <dj>
2440 print:print " (v)vorwaerts,(r
)rueckwaerts,(m)menue"              <od>
2450 get x$:if x$=""then goto 2450  <bb>
2460 if x$="m"then 860:rem close 4  <ib>
2470 if x$="v"and y<z then y=y+1:g
oto 2750                            <kf>
2480 if x$="r"and y>1 then y=y-1:g
oto 2750                            <gi>
2490 fe$=b2$+"eingabefehler!":gosu
b 640:goto 2450                     <pl>
2500 print:printb2$;:input"druecke
n sie return";x$                    <oi>
2510 next y                         <ll>
2520 gosub 710:print:print          <fm>
2530 printb2$"** suchbegriff nicht
 vorhanden **"                      <ak>
2540 print:printb2$;:input"druecke
n sie return";x$                    <na>
2550 rem if g$="d"then close 1      <mb>
2560 goto 860                       <gi>
2570 rem ========================== <op>
2580 rem  programm beenden          <fg>
2590 rem ========================== <gb>
2600 gosub 710:print:print          <ol>
2610 if z=0 then 2740               <ad>
2620 trap3730                       <dp>
2630 printb2$"sind alle daten gesi
chert(j/n)";                        <nl>
2640 input x$                       <lj>
2650 if x$="n"then 860              <fn>
2660 if x$="j"then 2680             <op>
2670 print chr$(145);:goto 2630     <id>
2680 gosub 710:gosub4300            <mp>
2690 print:print:printb2$"das prog
ramm kann mit'f6'wieder"            <be>
2700 printb2$"gestartet werden,ohn
e dass daten"b5$                    <bd>
2710 printb2$"verloren gehen!"      <om>
2720 print:print                    <mh>
2730 end                            <oj>
2740 sys62359:goto2730              <ae>
2750 goto2340                       <jg>
2760 rem ========================== <ai>
2770 rem  auto tabelle 1            <kn>
2780 rem ========================== <kk>
2790 zh=1:c=9:trap3730              <lk>
2800 gosub710:print:print           <hj>
2810 if z=0then gosub790:goto860    <jp>
2820 printzf$"CCCCC"zh$"CCCCCCC"zh
```

```
$"CCCCC"zh$"CCCCCC"zh$"CCCCCC"zh$"
CCCC"zd$                                <ja>
2830 printusing"Bdat##Bkm/st"+b2$+
"Bge.kmBbenz. B dm"+b3$+"BverbB";j
a$                                      <lb>
2840 printza$"CCCCC"ym$"CCCCCCC"ym
$"CCCCC"ym$"CCCCCC"ym$"CCCCCC"ym$"
CCCC"zi$                                <jg>
2850 foru=zhtoz                         <dp>
2860 g=d(u,2)-d(u-1,2)                  <mo>
2870 printusing"B##.##B";d(u,1);        <pj>
2880 printusing" ######B";d(u,2);       <la>
2890 printusing" ####B";g;              <mp>
2900 printusing"####.#B";d(u,3);        <dj>
2910 printusing"####.#B";d(u,4);        <nk>
2920 printusing"##.#B";d(u,3)/g*10
0                                       <mn>
2930 ifu=cthengoto2950                  <eo>
2940 next                               <jb>
2950 be=0:dm=0                          <me>
2960 printzc$"=====ym$"=======ym
$"=====ym$"======ym$"======ym$"
====zs$                                 <oj>
2970 printusing" gesa"+b3$+"######
";d(z,2);:printusing" #####";d(z,2
)-d(1,2);                               <ok>
2980 forr=1tou:be=be+d(r,3):dm=dm+
d(r,4):nextr                            <ob>
2990 printusing" ####.#";be;:print
using" ####.#";dm                       <nn>
3000 print:printb2$;:input"druecke
n sie return";x$                        <pi>
3010 ifu<zthenzh=u+1:c=c+9:goto280
0                                       <md>
3020 goto860                            <dp>
3030 rem ======================== <nf>
3040 rem  auto tabelle 2                <bk>
3050 rem ======================== <ej>
3060 zh=1:c=9 :trap3730                 <jo>
3070 gosub710:print:print               <oe>
3080 if z=0then gosub790:goto860        <kd>
3090 printzf$"CCCCCCCC"zh$"CCCCCCC
C"zh$"CCCCCCCCCC"zh$"CCCCCCCCC"zd$ <hk>
3100 printusing"Bdatum ##Bol/km"+b
3$+"Breparatur BsonstigesB";ja$         <bf>
3110 printza$"CCCCCCCC"ym$"CCCCCCC
C"ym$"CCCCCCCCCC"ym$"CCCCCCCCC"zi$ <im>
3120 foru=zhtoz                         <lh>
3130 printusing"B##.##"+b3$+"B";d(
u,1);                                   <pn>
3140 printusing"#####.##B";d(u,5);      <ae>
3150 printusingb4$+"###.##B";d(u,6
);                                      <fj>
3160 printusingb3$+"###.##B";d(u,7
)                                       <hi>
3170 ifu=cthengoto3200                  <gn>
3180 ifu=cthengoto2950                  <fp>
3190 next                               <ih>
3200 be=0:dm=0:ol=0:re=0:so=0           <lh>
3210 printzc$"========ym$"=======
="ym$"==========ym$"=========zs$ <bi>
3220 foraa=1tou:ol=ol+d(aa,5):re=r
e+d(aa,6):so=so+d(aa,7):nextaa          <bh>
3230 printusing" gesamt"+b4$+"####
.##";ol;:printusingb4$+"####.##";r
e;                                      <ne>
3240 printusingb3$+"####.##";so         <ac>
3250 print:printb2$;:input"drucken
 sie return";x$                         <hp>
3260 ifu<zthenzh=u+1:c=c+9:goto307
0                                       <bl>
3270 goto860                            <oo>
3280 rem ======================== <jp>
3290 rem  auto dat.ausgeben             <nd>
3300 rem ======================== <gm>
3310 lb=0:dm=0:re=0:so=0:ef=0 :tra
p3730                                   <ap>
3320 rem gosub 730:print                <kp>
3330 if z=0 then gosub 790:goto 86
0                                       <fg>
3340 rem input;c4$" drucker oder b
ildschirm(d/b)";g$                      <ek>
3350 rem if g$="d"then open 1,4         <pb>
3360 gosub 710:print                    <li>
3370 fori=1toz                          <hg>
3380 lb=lb+d(i,3):dm=dm+d(i,4):re=
re+d(i,5):so=so+d(i,6):ef=ef+d(i,7
)                                       <oe>
3390 nexti                              <hj>
3400 print f$(1);d(z,1);"."ja$          <hk>
3410 print f$(2);d(z,2)                 <bb>
3420 print f$(3);lb                     <fg>
3430 print f$(4);dm                     <hn>
3440 print f$(5);re                     <ap>
3450 print f$(6);so                     <gh>
3460 print f$(7);ef                     <fg>
3470 printusingb2$+"verbrauchs dur
chschnitt ##.## l";lb/(d(z,2)-d(1,
2))*100                                 <pp>
3480 print:printb2$"die gesamt kos
ten sind";dm+re+so+ef;"dm"              <oo>
3490 print:printb2$"gefahrene km";
d(z,2)-d(1,2)                           <cg>
3500 print:printusingb2$+"der km k
ostet ##.##dm";(dm+rf+so+ef)/(d(z,
2)-d(1,2))                              <di>
3510 rem ifg$="d"thenclose1,4           <hl>
3520 print:printb2$;:input"druecke
n sie return";x$                        <oi>
3530 goto860                            <dp>
3540 print:printb2$;:input"druecke
n sie return";x$                        <aa>
3550 goto860                            <ho>
3560 rem ======================== <ej>
3570 rem  fehler kanal                  <fe>
3580 rem ======================== <gf>
```

```
3590 close15:close1                                    <da>
3600 open15,8,15,"i"                                   <fh>
3610 input#15,e,e$,m,n:ifo<20oro=5
0then:return                                           <cc>
3620 gosub 550:print:print                             <ol>
3630 char,10,5,"*****************
*"                                                     <gn>
3640 char,10,6,"*"+b3$+"fehler kan
al"+b2$+"*"                                            <mj>
3650 char,10,7,"*****************
*":print                                               <bk>
3660 print" "c3$c4$f1$rn$"fehler"r
f$;fo$" :";e;e$;m;n                                    <ml>
3670 close15:close1                                    <be>
3680 printc4$c3$c3$"ueberpruefe di
e diskettenstation"                                    <hl>
3690 printc4$c3$c3$"druecken sie d
anach return";:inputx$                                 <no>
3700 close15                                           <cn>
3710 iff=2orf=10orf=11thengoto860:
elseclr                                                <hj>
3720 gosub5340:goto130                                 <jo>
3730 rem ========================                      <an>
3740 rem  fehlerhafte daten                            <pc>
3750 rem ========================                      <hi>
3760 gosub550:print:print                              <dn>
3770 char,10,5,"*****************
*"                                                     <da>
3780 char,10,6,"*"+b6$+"fehler"+b5
$+"*"                                                  <cg>
3790 char,10,7,"*****************
*":print                                               <bn>
3800 printc4$c4$c3$c3$"es wurden f
alsche oder nicht"                                     <ap>
3810 printc4$c3$c3$"genug daten ei
ngeben !"                                              <hc>
3820 printc4$c4$c4$c3$c3$"oder die
 diskettenstaton ist"                                  <ea>
3830 printc4$c3$c3$"nicht in ordnu
ng"                                                    <kf>
3840 printc4$c4$c3$c3$"druecken si
e danach return";:inputx$                              <fj>
3850 iff=2orf=10orf=11thengoto860:
elseclr:gosub5340:goto130                              <cc>
3860 rem ========================                      <oo>
3870 rem  daten loeschen                               <gj>
3880 rem ========================                      <cb>
3890 gosub 710:print:print::a$=""                      <jg>
3900 print:printleft$(qd$,6)c3$c3$
c3$"es werden alle daten"                              <en>
3910 printc3$c3$c3$"geloescht !"                       <fh>
3920 printhe$left$(qd$,9)c3$c3$c3$
"sind sie "rn$"sicher"rf$" (j/n)";
:inputx$                                               <ib>
3930 ifx$="n"then goto860                              <fh>
3940 ifx$="j"thengoto 3960                             <ip>
3950 ifx$<>"j"orx$<>"n"thengoto392
0                                                      <fo>
3960 printc4$c3$c3$c3$f1$"bitte wa
rten !"fo$                                             <ak>
3970 forx=1to2000:next:clr:gosub53
40                                                     <bi>
3980 goto130                                           <gb>
3990 rem ========================                      <kp>
4000 rem  dateien loeschen                             <hb>
4010 rem ========================                      <de>
4020 ifv=0thengosub4660                                <cd>
4030 gosub4490                                         <jh>
4040 trap3730                                          <hf>
4050 print:print" welche datei sol
l geloescht                                            <ok>
4060 print" werden ";                                  <em>
4070 inputloe1$                                        <ni>
4080 printc4$" sind sie "rn$"siche
r"rf$" (j/n)";                                         <jj>
4090 inputa$                                           <ae>
4100 ifa$="n"thengoto860                               <ib>
4110 ifa$="j"thengoto4130                              <lm>
4120 ifa$<>"j"ora$<>"n"thengoto426
0                                                      <oh>
4130 forqq=1tobb                                       <dk>
4140 ifdt$(qq)=loe1$ thengoto4160                      <cn>
4150 nextqq                                            <cl>
4160 forlo=qqtobb                                      <eb>
4170 dt$(lo)=dt$(lo+1)                                 <no>
4180 nextlo                                            <ba>
4190 bb=bb-1                                           <io>
4200 print:printc3$c3$"die datei w
ird "rn$;f1$"geloescht"fo$;rf$" !"                     <no>
4210 open1,8,15                                        <ah>
4220 print#1,"s:"+loe1$                                <cb>
4230 close1                                            <mm>
4240 gosub4270                                         <pf>
4250 goto860                                           <hc>
4260 printc2$left$(qr$,24)b5$left$
(q1$,4)c2$c2$:goto4080                                 <pi>
4270 rem ========================                      <ff>
4280 rem  directory saven                              <el>
4290 rem ========================                      <ai>
4300 open1,8,14,"@:auto-datei,s,w"                     <eb>
4310 print#1,bb                                        <fd>
4320 forgg=1tobb                                       <kb>
4330 print#1,dt$(gg)                                   <eg>
4340 nextgg                                            <pp>
4350 close1:f=2                                        <ck>
4360 gosub3560:close15                                 <ei>
4370 return                                            <ne>
4380 rem ========================                      <di>
4390 rem  directory laden                              <ha>
4400 rem ========================                      <il>
4410 open1,8,14,"auto-datei,s,r"                       <pf>
4420 input#1,bb                                        <od>
4430 forgg=1tobb-1                                     <mp>
4440 input#1,dt$(gg)                                   <jf>
4450 nextgg                                            <hf>
4460 close1                                            <gg>
```

```
4470 gosub3560:close15                    <pe>
4480 return                               <lb>
4490 rem ========================        <gk>
4500 rem  directory anzeigen              <mm>
4510 rem ========================        <kh>
4520 a$="":gg=1 :ch=1                     <nm>
4530 printcl$:gosub710 :print :pri
nt                                        <eo>
4540 forqq=chto(bb-1)                     <lf>
4550 printb2$"datei jahrgang "dt$(
qq)                                       <io>
4560 ifqq=bb-1thengoto4640                <am>
4570 ifgg=4thenprintc4$+" soll wei
ter geblaettert ":elsegoto4620            <hi>
4580 print" werden"b2$"(j/n)";:ch=
ch+gg: gg=1                               <he>
4590 a$="":inputa$:ifa$="j"thengot
o4530                                     <bm>
4600 ifa$="n"thenreturn                   <ep>
4610 ifa$<>"j"ora$<>"n"thengoto465
0                                         <pi>
4620 gg=gg+1                              <lo>
4630 nextqq                               <do>
4640 return                               <pc>
4650 printhe$left$(qd$,15)left$(qr
$,14);:goto4590                           <io>
4660 rem ========================        <nb>
4670 rem  keine datei vorhanden           <ei>
4680 rem ========================        <lh>
4690 gosub710                             <bj>
4700 ifv=0thengoto4730                    <nd>
4710 ifv=1thenreturn                      <pc>
4720 print:print                          <ih>
4730 printhe$left$(qd$,11)fl$b2$"k
eine datei vorhanden !"fo$                <cc>
4740 foraw=1to2000:next                   <lh>
4750 goto860                              <nd>
4760 rem ========================        <pc>
4770 rem  daten loeschen                  <ck>
4780 rem ========================        <ff>
4790 char,3,22," "+fl$+"es wird ge
loescht "+fo$:fort=1to500:nextt           <hg>
4800 forll=zztoz-1                        <ck>
4810 fori=1to7                            <ko>
4820 d(ll,i)=d(ll+1,i)                    <lj>
4830 da$(ll,i)=da$(ll+1,i)                <li>
4840 nexti                                <dh>
4850 nextll                               <ao>
4860 ifz=zzthenzz=zz-1                    <hf>
4870 z=z-1                                <eo>
4880 char,3,22,b$+b$+b4$                  <aa>
4890 goto1880                             <ef>
4900 rem ========================        <cb>
4910 rem  daten einfuegen                 <dl>
4920 rem ========================        <ei>
4930 ifzz=zthengoto1880                   <hl>
4940 char,3,22," "+fl$+"es wird ei
ngefuegt"+fo$:fort=1to500:nextt           <cp>
4950 forll=z+1tozz+1step-1                <pn>
4960 fori=1to7                            <ek>
4970 d(ll,i)=d(ll-1,i)                    <gg>
4980 da$(ll,i)=da$(ll-1,i)                <me>
4990 nexti                                <je>
5000 nextll                               <ml>
5010 z=z+1                                <nj>
5020 fori=1to7                            <hj>
5030 d(ll,i)=0                            <in>
5040 da$(ll,i)=""                         <fj>
5050 nexti                                <id>
5060 char,3,22,b$+b$+b2$                  <jg>
5070 goto1880                             <mp>
5080 end                                  <fh>
5090 rem ========================        <ej>
5100 rem  titelbild                       <cl>
5110 rem ========================        <gf>
5120 gosub5320:printhe$he$cl$             <hh>
5130 printc4$b2$                          <md>
5140 printb4$rn$z8$b2$yq$rf$" "rn$
" "rf$b2$rn$" "rf$" "rn$b5$rf$" "r
n$z8$b3$yq$rf$" "                         <he>
5150 printb4$rn$" "rf$b2$rn$" "rf$
" "rn$" "rf$b2$;                          <kn>
5160 printrn$" "rf$b3$rn$" "rf$b3$
rn$" "rf$b3$rn$" "rf$" "                  <eb>
5170 printb4$rn$" "rf$b2$rn$" "rf$
" "rn$" "rf$b2$;                          <ig>
5180 printrn$" "rf$b3$rn$" "rf$b3$
rn$" "rf$b3$rn$" "rf$" "                  <bh>
5190 printb4$rn$b4$rf$" "rn$" "rf$
b2$rn$" "rf$b3$rn$" "rf$b3$rn$" "r
f$b3$rn$" "rf$" "                         <ii>
5200 printb4$rn$" "rf$b2$rn$" "rf$
" "rn$" "rf$b2$;                          <og>
5210 printrn$" "rf$b3$rn$" "rf$b3$
rn$" "rf$b3$rn$" "rf$" "                  <cj>
5220 printb4$rn$" "rf$b2$rn$" "rf$
" "rn$b4$rf$b3$rn$" "rf$b3$yq$rn$b
3$rf$z8$                                  <gf>
5230 printc4$c4$b$b$b4$rn$b3$yq$rf
$" "rn$z8$b2$yq$rf$" "rn$b5$rf$           <ea>
5240 printb$b$b4$rn$" "rf$b2$rn$"
"rf$" "rn$" "rf$b2$rn$" "rf$b3$rn$
" "rf$                                    <bd>
5250 printb$b$b4$rn$" "rf$b2$rn$"
"rf$" "rn$" "rf$b2$rn$" "rf$b3$rn$
" "rf$                                    <bi>
5260 printb$b$rn$b3$rf$" "rn$" "rf
$b2$rn$" "rf$" "rn$b4$rf$b3$rn$" "
rf$                                       <he>
5270 printb$b$b4$rn$" "rf$b2$rn$"
"rf$" "rn$" "rf$b2$rn$" "rf$b3$rn$
" "rf$                                    <oo>
5280 printb$b$b4$rn$b3$rf$z8$" "rn
$" "rf$b2$rn$" "rf$b3$rn$" "rf$          <bl>
5290 printleft$(qd$,4)b4$"(c) 1987
 von klaus dieter breuer"                 <nh>
```

```
5300 printc4$b$b2$f1$"taste drueck
en"fo$:getkeyq$                    <io>
5310 return                        <dd>
5320 rem nachspann ============== <km>
5330 rem * farbcodes/steuercodes * <ag>
5340 c4$=chr$(017):rn$=chr$(018)   <aa>
5350 he$=chr$(019):c3$=chr$(029)   <bl>
5360 f1$=chr$(130):fo$=chr$(132)   <id>
5370 c2$=chr$(145):rf$=chr$(146)   <fl>
5380 c1$=chr$(147):c1$=chr$(157)   <ec>
5390 rem *** zeichensatz/graphik * <hg>
5400 z8$=chr$(169):za$=chr$(171)   <gd>
5410 zc$=chr$(173):zd$=chr$(174)   <ko>
5420 zf$=chr$(176):zh$=chr$(178)   <jk>
5430 zi$=chr$(179):zs$=chr$(189)   <od>
5440 ym$=chr$(219):yq$=chr$(223)   <jl>
5450 rem ********* zeichenfolgen * <nc>
5460 qd$="":ql$="":qr$="":for q=1
to 40                              <db>
5470 qd$=qd$+c4$:qr$=qr$+c3$       <cg>
5480 ql$=ql$+c1$                   <kd>
5490 nextq:b$=chr$(32):b2$=b$+b$   <am>
5500 b3$=b2$+b$:b4$=b3$+b$         <md>
5510 b5$=b4$+b$:b6$=b5$+b$         <pj>
5520 b$=b5$+b5$:return             <lc>
5530 rem ========================= <kh>
5540 rem 60671 bytes memory        <hl>
5550 rem 14305 bytes program       <oo>
5560 rem 00392 bytes variables     <ki>
5570 rem 12976 bytes arrays        <aj>
5580 rem 01383 bytes strings       <cc>
5590 rem 31615 bytes free (0)      <af>
5600 rem (bei 4 datensaetzen)      <oo>
5610 rem ========================= <ge>
```



```
  10 rem disk-retter============c16 <nj>
  20 rem (p) commodore welt team   <ho>
  30 rem ========================= <ng>
  40 rem (c) by werner fuchs       <ec>
  50 rem                           <pd>
  60 rem                           <ah>
  70 rem basic v3.5                <nl>
  80 rem c16/116/plus4             <ki>
  90 rem ========================= <jg>
 100 gosub 800                     <bl>
 110 printchr$(14)                 <lc>
 120 gosub660:re%=0                <ah>
 130 printcl$bk$" Beginn der Suche
Spur "di%",1"c4$                   <bb>
 140 sp%=1:tr%=di%:open15,8,15,"i0 <al>
 150 open8,8,2,"#2":fi%=0          <ne>
 160 print#15,"u1:"2;0;tr%;sp%:gosu
b600                               <od>
 170 fori=0to1:get#8,a$:gosub570:p%
(i)=asc(a$):next:a=2:gosub590      <kg>
 180 get#8,a$:gosub570:a=asc(a$)and
7                                  <oi>
 190 h$="":fori=0to1:get#8,a$:gosub
570:1%(i)=asc(a$):next             <ig>
 200 fori=3to18:get#8,a$:h$=h$+a$:n
ext                                <ne>
 210 ifh$=""then230                <cb>
 220 print ki$(a);" ";co$;h$;tab(11
)co$;;1%(0);1%(1):ifa=0thengoto400 <jk>
 230 fi%=fi%+1:iffi%<8thena=fi%*32+
2:gosub590:goto180                 <eo>
 240 ifp%(0)=0then260              <eg>
 250 printc4$"Wechsel auf Spur";:tr
%=p%(0):sp%=p%(1):fi%=0:printtr%;s
p%c4$:goto160                      <gl>
 260 close8:printrn$c4$"Directory E
nde"rf$,:ifre%thenprint"Collect Dr
ive 1":print#15,"v0                <kg>
 270 gosub600:close15:end          <ef>
 280 rem**  retrieve a file **     <op>
 290 print#15,"u1:"2;0;1%(0);1%(1):
return                             <gd>
 300 d%=2:printc4$"Spur, Sektor";  <ge>
 310 gosub290:fori=0to1:get#8,a$:go
sub570:z%(i)=asc(a$)               <eb>
 320 printz%(i)c1$" ";:next:print:p
rintc2$"Spur, Sektor";             <gj>
 330 ifz%(0)=0thenprint:return     <pe>
 340 print#15,"b-f"0;z%(0);z%(1):1%
(0)=z%(0):1%(1)=z%(1):z%(d%)=z%(0) <gi>
 350 z%(d%+1)=z%(1):d%=d%+2        <nd>
 360 ifds=0then310                 <ob>
 370 print:printrn$"Das File wurde
bereits ueberschrieben"rf$"!":x%=n
ot(x%):a=fi%*32+2                  <lj>
 380 fori=2tod%-4step2:print#15,"b-
f:"0;z%(i);z%(i+1):next            <ka>
 390 1%(0)=tr%:1%(1)=sp%:gosub290:g
```

Disk-Retter

GELÖSCHTE PROGRAMME WIEDER HERHOLEN

Ein Versehen ist schnell passiert. Es ist ärgerlich, wenn ein dringend benötigtes Programm durch eine Unaufmerksamkeit beim Löschen auf Nimmerwiedersehen verschwindet. Raufen Sie sich nicht die Haare, laden Sie einfach den Disk-Rettter und holen Sie das Programm wieder zurück.

Mit unserem Programm wird das Restaurieren gelöschter Files zum Kinderspiel, doch in einigen Fällen kann auch der Disk-Retter nicht mehr helfen. Wenn ein Programm durch Überschreiben nicht mehr existiert, dann ist es hoffnungslos verloren. Wurde es allerdings nur mit dem SCRATCH-Befehl gelöscht und haben weitere Speicherungen im Anschluß daran nicht mehr stattgefunden, so dürfen Sie noch hoffen.

```
oto590                                          <ch>
400 print"Zurueckholen? (J/N)"left
$(q2$,4)                                        <on>
410 geta$:ifa$=""then410                        <ae>
420 ifa$<>"j"then230                            <im>
430 x%=0:printc4$"Blocks wiederbel
egen...":z%(0)=1%(0):gosub300:ifx%
then230                                         <cm>
440 printc4$"Ist dies ein "rn$"P"r
f$"rogramm,"rn$"S"rf$"equenielles,
 "rn$"U"rf$"ser,";                              <be>
450 print"oder "rn$"R"rf$"elatives
 File";                                         <ei>
460 get ja$:ifja$<>"p"andja$<>"s"a
ndja$<>"u"andja$<>"r"then460                    <nl>
470 print:print                                 <co>
480 print:ifja$="p"thenprintrn$+"P
rogramm"+rf$+"-":ki%=130:goto520                <ec>
490 ifja$="u"thenprintrn$+"User"+r
f$+"-":ki%=131:goto520                          <dh>
500 ifja$="r"thenprintrn$+"Relativ
es"+rf$+"-":ki%=132:goto520                     <bc>
510 printrn$"Sequentielles"rf$"-":
ki%=129                                         <dm>
520 print"File wiederholen..."                  <ad>
530 1%(0)=tr%:1%(1)=sp%:gosub290                <pe>
540 a=fi%*32+2:gosub590:print#8,ch
r$(ki%);                                        <ae>
550 bl%=((d%-2)/2)+1:printc4$bl%"B
loecke ";                                       <na>
560 print#15,"u2:"2;0;tr%;sp%:prin
t"Wiederhergestellt !"c4$:re%=1:go
to230                                           <mj>
570 ifa$=""thena$=chr$(0)                       <gg>
580 return                                      <bk>
590 print#15,"b-p:"2;a                          <mf>
600 input#15,en%,em$,et%,es%:ifen%
=0thenreturn                                    <fn>
610 printc4$c4$rn$"disk fehler"rf$              <ab>
620 printc4$"Fehler Meldung"rf$":
"en%" "em$","et%","es%                          <lb>
630 print"bitte 'CONT' geben.                   <na>
640 end                                         <if>
650 return                                      <kh>
660 dimp%(1),1%(1),z%(300),ki$(4):
co$=chr$(34)                                    <ad>
670 ki$(0)=rn$+"DEL"+rf$:ki$(1)="S
EQ":ki$(2)="PRG":ki$(3)="USR":ki$(
4)="REL"                                        <nm>
680 pg$=bk$+cl$+rn$+"       Disk Fi
le Retter VC1551/1541       "+rf$               <pp>
690 printpg$c4$:print"Dieses Progr
amm ermoeglicht das Wieder-"                    <cb>
700 print"herstellen von geloescht
en Files.                                       <hi>
710 print"Es wird vorausgesetzt,da
ss die Disk noch";                              <pb>
720 print"nicht wieder beschrieben
 wurde.                                         <dd>
730 printc4$"100% Erfolg ist nur n
ach 'SCRATCH' ge-"                              <ij>
740 print"waehrleistet.                         <ig>
750 printc4$"Beim Suchen werden nu
r die geloeschten  Dateien abgefra
gt"                                             <id>
760 printc4$c4$c4$"   Taste drueck
en !"                                           <ec>
770 get td$:if td$="" then 770                  <cj>
780 di%=18:return                               <gp>
790 printleft$(qu$,4):goto760                   <oe>
800 rem nachspann ==============                <pp>
810 rem * farbcodes/steuercodes *               <jh>
820 c4$=chr$(017):rn$=chr$(018)                 <jf>
830 bk$=chr$(144):c2$=chr$(145)                 <di>
840 rf$=chr$(146):cl$=chr$(147)                 <el>
850 c1$=chr$(157)                               <nh>
860 rem *** zeichensatz/graphik *               <ll>
870 s2$=chr$(160)                               <pn>
880 rem ********* zeichenfolgen *               <mc>
890 for q=1 to 40                               <le>
900 q2$=q2$+s2$:qu$=qu$+c2$                     <ni>
910 next q                                      <jm>
920 return                                      <mf>
930 rem ========================                <jm>
940 rem 12277 bytes memory                      <oe>
950 rem 03282 bytes program                     <mf>
960 rem 00217 bytes variables                   <ac>
970 rem 00653 bytes arrays                      <ck>
980 rem 00509 bytes strings                     <fj>
990 rem 07616 bytes free (0)                    <mk>
1000 rem ======================                <np>
```

Der Disk-Retter zeigt Ihnen den Inhalt der Diskette. Wenn er ein gelöschtes Programm findet, will er wissen, ob Sie es wiederhaben wollen. Ist Ihre Antwort positiv, folgt die Frage, ob es ein Programm-, ein sequentielles, ein User-, oder ein relatives File sein soll. Treffen Sie durch Eingabe des jeweiligen ersten Buchstabens Ihre Wahl. Der Disk-Retter beginnt jetzt mit seiner Arbeit und meldet nach einiger Zeit, daß Ihr Programm wieder existiere. Ein wenig müssen Sie sich allerdings noch gedulden, denn die Verwaltung Ihrer Diskette ist wieder auf Vordermann zu bringen, einige Blöcke, die schon zum Überschreiben freigegeben worden waren, sind wieder zu sperren. Nach getaner Arbeit geht es weiter mit der Auflistung der noch fehlenden Files. Sind keine gelöschten mehr zu finden, so übergibt der Rechner die Kontrolle an Sie. Und Sie dürfen nach Herzenslust weitermachen. Aber besser nicht gleich wieder die falschen Programme löschen. □

Universelle Datei

Dateien können auf die verschiedenste Weise verwaltet werden. Die „relative" Dateiablage ist weniger verbreitet, da sie schwieriger zu programmieren ist. Mit diesem Listing geht's aber problemlos.

Dieses Programm ermöglichst relative Dateiverwaltung aller möglichen Variationen bis 254 Byte pro Datensatz und verwaltet bis 20 unterschiedliche Dateien. Es ist möglich, pro Datensatz bis zehn Datenfelder einzurichten. Bei 40 Zeichen werden allerdings nur sechs Datenfelder zur Verfügung gestellt. Bei kürzeren Datenfeldern können je nach Recordlänge bis zu zehn Datenfelder definiert werden.
Nach dem Programmstart erscheint das Menü, dann kann mit dem Cursor der einzelne Menüpunkt angewählt werden. Wenn keine Datei vorliegt, antwortet das Programm mit entsprechenden Hinweisen. Nur der Menüpunkt „Datei einrichten" läßt sich anwählen, um eine Datei zu erzeugen!
Nach der Datums- und Dateinamen-Eingabe wird nach der Feldbezeichnung und Feldlänge gefragt. Als Feldbezeichnung sollten etwa bei Adressen, Name, Straße, Postleitzahl, Ort, Telefon, genannt werden. Der Strich bei Feldbezeichnung markiert lediglich die mögliche Eingabelänge. Die Feldlänge kann 25 Zeichen nicht überschreiten. Bei erreichter Recordlänge, Escape oder dem Erreichen des zehnten Datenfeldes wird die Funktion beendet und die Parameter werden automatisch festgestellt.
Danach erscheint die Eingabe für die Datensätze. Der Strich markiert nur die Datenfeldlänge. Es ist wichtig, daß im ersten Datenfeld die Bezeichnung steht, die für die Identifizierung maßgebend ist, also bei Adressen der Personenname oder bei Bibliotheken der Buchtitel. Sie darf sich auf keinen Fall in einem anderen Datensatz wiederholen, da dieser Datensatz sonst nicht angenommen wird. Nummern sind nicht zu empfehlen, da hier keine Kontrolle für doppelte Eingaben, die automatisch ausgeführt wird, möglich ist.
Ebenso sind Bezeichnungen, die in mehreren Datensätzen gleich sein müssen, also bei Adressen der Ort oder bei Bibliothek der Autor, im ersten Datenfeld zu vermeiden. Die Recordnummer wird vom Computer intern ermittelt und verwaltet und tritt nicht in Erscheinung. Wenn die Eingabefunktion beendet werden soll, ist die Taste Escape zu drücken. Dies gilt für alle Funktionen.

Dann erscheint das Anfangsmenü wieder. Wenn jetzt die Menüpunkte „Programm beenden", „Datei löschen" und „Datei Wahl" angewählt werden, schließen die Datenkanäle automatisch und die Parameterdatei wird als „Par.Dateiname" abgelegt. Die Dateinamen der unterschiedlichen Dateien werden unter der Datei „Rel.Files-Liste" zusammengefaßt und gespeichert. Vor dem Zugriff auf eine Datei muß „Datei Wahl" gewählt werden, denn nur hier werden die Kanäle geöffnet. Der entsprechende Dateiname wird mit dem Cursor ausgesucht und mit Return werden die entsprechenden Parameter geladen und die Kanäle geöffnet.

Nun können die Menüpunkte „Datenausgabe", „Dateneingabe" und „Daten ändern" angewählt werden. Andernfalls erscheint eine entsprechende Meldung. Da das Abbrechen des Programmes durch die Stoptaste nicht möglich ist, muß der Menüpunkt „Programm beenden" benutzt werden, um alle Kanäle zu schließen und Paramterdateien zurückzuschreiben.
Die Funktion der Suchroutine ist folgende: Nach Eingabe der Anfangsbuchstaben des Suchbegriffs werden alle Datensätze, die mit den gleichen Zeichen beginnen, hinter die Forderung Suchbegriff ausgegeben. Dann wartet das Programm drei Sekunden. Wenn in dieser Zeit nicht Return gedrückt wird, um den I.D.-Code zu übernehmen, wird an dieser Stelle der nächste Datensatz vorgeschlagen.

Dieter Kukkel □

```
10 rem relative datei =========c16 <eg>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by dieter kukkel   c128 <lk>
50 rem (v) a. mittelmeyer      c16 <la>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/plus4 + floppy     <dn>
90 rem ========================== <jg>
100 gosub110:sys818:goto160       <lg>
110 restore:fori=818to830         <hj>
120 reada:pokei,a:next:return     <bk>
130 data 120,169,102,141,038,003  <bd>
140 data 169,204,141,039,003,088  <fj>
150 data 096                      <oo>
160 gosub1690:scnclr              <cc>
170 dim n$(8),t(8),x(8),y(8),dn$(2
5),df$(20),lr(20),dx$(20),id$(500) <ig>
180 sp$=chr$(32):pu$=chr$(163):ad=
0:lf=8:rp=1:s=1:fl=0:f=0          <ol>
190 forj=0to38:s8$=s8$+sp$:nextj:r
estore210                         <ng>
200 fori=1to7:readn$(i),t(i),x(i),
y(i):nexti                        <kk>
210 data " datei-auswahl ",1,0,3,"
 daten-ausgabe ",2,0,4            <gg>
220 data " daten-eingabe ",3,0,5,"
 daten-aendern ",4,0,6            <cl>
230 data " datei einrichten ",5,20
,3," datei loeschen ",6,20,4      <ie>
240 data " programm beenden ",0,20
,5                                <kl>
250 :                             <cp>
260 gosub850:char1,1,0,rf$+"relati
ve index-datei"+b$+b3$+"c16 "+ro$ <mk>
270 char1,1,1,rf$+"(c) commodore w
elt/dieter kukkel"+ro$            <jh>
280 char1,0,8,rf$+s8$+ro$:char,1,8
,rf$+"cursor-wahl :"+ro$          <dg>
290 fori=1to7:char1,x(i),y(i),n$(i
):nexti                           <ic>
300 char,x(s),y(s),rf$+n$(s)+ro$:c
har,21,8,rf$+"funktion"+str$(t(s))
+ro$                              <hn>
310 getkey z$:char,x(s),y(s),n$(s) <ak>
320 ifz$=chr$(145)then s=s-1:ifs<1
thens=s+7                         <nd>
330 ifz$=chr$(17) then s=s+1:ifs>7
thens=s-7                         <go>
340 ifz$=chr$(29) then s=s+4:ifs>7
thens=s-7                         <nl>
350 ifz$=chr$(157)then s=s-4:ifs<1
thens=s+7                         <hc>
360 ifz$<>chr$(13)then300         <dn>
370 ifs>1ands<5then ifdn$=""then g
osub970:goto260                   <em>
380 ifs=3ors=4then gosub1060      <fl>
390 on t(s) gosub 420,590,720,590,
1120,1590:ift(s)<>0then260        <bd>
400 gosub1470:scnclr:gosub110:poke
820,101:poke825,242:sys818:end    <cd>
410 :                             <hb>
420 iffl=0thengosub1360:iffl=0then
return                            <fb>
430 s=1:x=1:y=11:fori=1tofl:ifx>30
thenx=1:y=y+1                     <ap>
440 char1,x,y,dn$(i):x=x+20:nexti:
ifdn$<>""then gosub1470           <go>
450 gosub1040:iff=0then char1,1,24
,rf$+"welche datei wird gewuenscht
 ?"+ro$                           <ak>
460 iff=1then char1,1,24,rf$+"welc
he datei soll geloescht werden ?"+
ro$                               <kl>
470 z=s-1:y=int(z/2)+11:x=((z/2)-i
nt(z/2))/.5*20                    <hn>
480 char,13,23,rf$+b3$+ro$:char,1,
23,rf$+"datei-wahl"+str$(s)+ro$   <gd>
490 char,x,y,rf$+sp$+dn$(s)+sp$+ro
$                                 <mb>
500 getkey z$:char,x,y,sp$+dn$(s)+
sp$                               <oh>
510 ifz$=chr$(145)then s=s-2:ifs<1
thens=s+20                        <kj>
520 ifz$=chr$(17) then s=s+2:ifs>2
0thens=s-20                       <dd>
530 ifz$=chr$(29) then s=s+1:ifs>2
0thens=s-20                       <cn>
540 ifz$=chr$(157)then s=s-1:ifs<1
thens=s+20                        <na>
550 ifz$<>chr$(13)then470         <no>
560 ifs=<flthen dn$=dn$(s):s=1:els
e s=1:goto470                     <mb>
570 iff=1then return:else gosub104
0:goto1410                        <pl>
580 :                             <mg>
590 gosub870:char1,0,23,rf$+">>> a
chtung <<<"+b2$+"record nr."+ro$  <la>
600 char1,0,24,rf$+"mit return ueb
ernehmen"+ro$:rn=0                <ok>
610 char1,0,2,"suchbegriff:":input
nn$                               <gc>
620 fori=1toad:char1,27,23,rf$+str
$(i)+ro$:ifleft$(id$(i),len(nn$))<
>nn$then640                       <ee>
630 char1,14,2,id$(i):gosub1680:ge
tt$:ift$=chr$(13)then660          <ni>
640 char1,14,2,left$(s8$,25):nexti
:gosub1040                        <og>
650 char1,5,24,rf$+"datensatz ist
nicht vorhanden !"+ro$:gosub1680:g
osub1040:goto830                  <gm>
660 rn=i:gosub1040:gosub900:ifrl<8
9then680                          <pm>
670 fori=1todf:forj=1tolr(i):get#1
f,eg$:dx$(i)=dx$(i)+eg$:nextj,i:go
```

```
to690                                                    <in>
680 input#lf,rc$:lr=1:fori=1todf:d
x$(i)=mid$(rc$,lr,lr(i)):lr=lr+lr(
i):nexti                                                 <mk>
690 fori=1todf:char1,0,i*2+2,df$(i
):char1,14,i*2+2,dx$(i):nexti                            <nd>
700 gosub950:ifs=4then fori=1todf:
goto730:else830                                          <co>
710 :                                                    <ml>
720 gosub870:ad=ad+1:rn=ad:fori=1t
odf                                                      <gl>
730 char1,0,i*2+2,df$(i):forj=1tol
r(i):char1,13+j,i*2+3,pu$:nextj                          <lj>
740 char1,12,i*2+2,"":inputdx$(i):
iflen(dx$(i))>lr(i)ordx$(i)=""then
730                                                      <be>
750 ifi>1then790:else ifs=4then id
$(rn)=""                                                 <kh>
760 f=0:forj=1toad:ifid$(j)<>dx$(1
)then780                                                 <im>
770 char1,1,23,rf$+"i.d.-code exis
tiert"+ro$:f=1:gosub1680                                 <ad>
780 nextj:gosub1040:iff=0then id$(
rn)=dx$(1):else730                                       <kp>
790 lr$="":forj=1tolr(i):lr$=lr$+s
p$:nextj                                                 <oi>
800 dx$(i)=left$(dx$(i)+left$(lr$,
abs(lr(i)-len(dx$(i)))),lr(i))                           <bk>
810 rc$=rc$+dx$(i):nexti:gosub900:
print#lf,rc$:gosub900:gosub950                           <io>
820 gosub1040:char1,1,24,rf$+"date
nsatz ist gesichert"+ro$:gosub1680                       <im>
830 gosub1020:ift$=chr$(27)thenret
urn:else ifs=3then720:else590                            <ji>
840 :                                                    <na>
850 scnclr:char1,0,0,rf$+s8$+ro$:c
har1,0,1,rf$+s8$+ro$:goto1040                            <no>
860 :                                                    <pi>
870 gosub850:char1,0,0,rf$+n$(s)+s
p$+dn$+ro$                                               <jn>
880 char1,1,1,rf$+"aktualisiert am
 : "+j$+ro$:return                                       <ai>
890 :                                                    <de>
900 ah=int(rn/256):al=rn-256*ah:pr
int#15,"p"+chr$(96+lf)+chr$(al)+ch
r$(ah)+chr$(rp)                                          <kc>
910 char1,1,23,rf$+ds$+ro$:ifds<20
ords=50ords=62then return                                <jd>
920 ifds=52then gosub1470:else gos
ub1520                                                   <fd>
930 run                                                  <nf>
940 :                                                    <jj>
950 fori=1todf:dx$(i)="":nexti:rc$
="":return                                               <af>
960 :                                                    <mb>
970 char1,1,23,rf$+"datei-auswahl
oder "+ro$                                               <ae>
980 char1,1,24,rf$+"datei-einricht
en waehlen"+ro$:gosub1680:return                         <ah>
990 :                                                    <pn>
1000 char1,0,2,rf$+left$(s8$,18):r
eturn                                                    <ec>
1010 :                                                   <cf>
1020 char1,1,23,rf$+"funktion wied
erholen"+ro$                                             <fn>
1030 char1,1,24,rf$+"ja=(space)"+b
2$+"/"+b2$+"nein=(escape)"+ro$:get
keyt$                                                    <cc>
1040 char1,0,23,rf$+s8$+ro$:char1,
0,24,rf$+s8$+ro$:return                                  <gm>
1050 :                                                   <hg>
1060 char1,0,10,rf$+" aktuelles da
tum "+ro$:ifj$=""thenj$="01.09.198
7"                                                       <od>
1070 char1,20,10,j$:char1,18,10,""
:inputp$:char1,0,10,s8$                                  <fe>
1080 iflen(p$)<>10orval(left$(p$,2
))>31then1060                                            <ch>
1090 ifval(mid$(p$,4,2))>12orval(r
ight$(p$,4))<1980then1060                                <mm>
1100 j$=p$:return                                        <gp>
1110 :                                                   <oo>
1120 ifdn$=""then gosub1360:else g
osub1470                                                 <fj>
1130 iffl=20thenchar1,1,23,rf$+"ma
x. dateien-anzahl ist erreicht"+ro
$:gosub1680:return                                       <io>
1140 ifad>0thenfori=1toad:id$(i)="
":nexti:ad=0                                             <lj>
1150 gosub850:char1,1,0,rf$+"relat
ive index-datei einrichten"+ro$:go
sub1060                                                  <lj>
1160 dn$="":char1,0,10,rf$+" datei
-name "+ro$:fori=16to27                                  <dg>
1170 char1,i,11,pu$:nexti:char1,14
,10,"":inputdn$:char1,0,10,s8$                           <cg>
1180 char1,0,11,s8$:iflen(dn$)>12o
rdn$=""then1160                                          <hh>
1190 fori=1tofl+1:ifdn$(i)=dn$then
char1,1,23,rf$+"datei-name existie
rt"+ro$:goto1160                                         <gh>
1200 ifdn$(i)=""then dn$(i)=dn$:fl
=i                                                       <af>
1210 nexti:gosub1040:char1,0,2,s8$
:char1,1,1,rf$+"parameter-eingabe"
+b3$+dn$+ro$                                             <kn>
1220 rl=0:b=253:i=0:do:i=i+1:lr(i)
=0:df$(i)=""                                             <ap>
1230 char1,0,i*2+2,rf$+" feld nr."
+str$(i)+sp$+ro$                                         <me>
1240 gosub1000:char1,0,2,rf$+" fel
d-bezeichnung "+ro$                                      <hd>
1250 forj=15to26:char1,j,i*2+3,pu$
:nextj:char1,13,i*2+2,"":inputdf$(
i)                                                       <ei>
1260 iflen(df$(i))>12ordf$(i)=""th
```

```
enforj=13to35:char1,j,i*2+2,sp$:ne
xtj:goto1240                          <fg>
1270 gosub1000:char1,0,2,rf$+" fel
d-laenge "+ro$:char1,27,i*2+2,sp$:
inputlr(i)                            <oe>
1280 iflr(i)<1orlr(i)>25orlr(i)>bt
hen1270:elserl=rl+lr(i):b=b-lr(i)     <dg>
1290 gosub1000:char1,0,2,rf$+" rec
ord-laenge "+ro$:char1,36,2,b3$       <bh>
1300 char1,29,2,rf$+str$(rl)+sp$+r
o$:char1,35,2,str$(b):ifi=10orb=0t
henexit                               <kh>
1310 char1,1,23,rf$+"weitere daten
felder einrichten"+ro$                <ik>
1320 gosub1030:ift$=chr$(27)then e
xit                                   <hp>
1330 loop:df=i:rl=rl+1:gosub1020:i
ft$=chr$(27)thengosub1440:s=3:goto
720                                   <nb>
1340 dn$(fl)="":fl=fl-1:goto1150      <hb>
1350 :                                <na>
1360 open2,8,3,"0:rel.files-liste,
s,r":gosub910:ifds=62thenclose2:go
to1380                                <kj>
1370 input#2,fl:fori=1tofl:input#2
,dn$(i):nexti:close2:return           <fm>
1380 char1,1,23,rf$+"dateien nicht
 existent"+ro$                        <pj>
1390 char1,1,24,rf$+"erst datei ei
nrichten"+ro$:gosub1680:return        <mf>
1400 :                                <de>
1410 open2,8,3,"0:par."+dn$+",s,r"
:gosub910                             <gb>
1420 input#2,ad,df,rl,j$:fori=1tod
f:input#2,df$(i),lr(i):nexti          <fl>
1430 fori=1toad:input#2,id$(i):nex
ti:close2                             <bd>
1440 open15,8,15:openlf,8,lf,dn$+"
,l,"+chr$(rl):gosub910                <hm>
1450 char1,1,24,rf$+"dateiarbeit k
ann begonnen werden."+ro$:gosub168
0:return                              <fp>
1460 :                                <kn>
1470 ifdn$=""thenreturn:else gosub
1540                                  <ed>
1480 print#15,"s:par."+dn$            <ad>
1490 open2,8,3,"0:par."+dn$+",s,w"
:gosub910                             <ng>
1500 print#2,ad","df","rl","j$:for
i=1todf:print#2,df$(i)","lr(i):nex
ti                                    <am>
1510 fori=1toad:print#2,id$(i):nex
ti                                    <ng>
1520 close2:cloself:close15:dn$=""
:return                               <hb>
1530 :                                <dj>
1540 char1,1,24,rf$+"daten-kanaele
 werden geschlossen "+ro$             <hl>
1550 print#15,"s:rel.files-liste"     <jc>
1560 open2,8,3,"0:rel.files-liste,
s,w":gosub910                         <np>
1570 print#2,fl:fori=1tofl:print#2
,dn$(i):nexti:close2:return           <eb>
1580 :                                <jo>
1590 f=1:gosub420:gosub1040:iffl=0
thenreturn                            <dd>
1600 char1,1,23,rf$+"loesch-vorgan
g abbrechen ?"+ro$:gosub1030          <dk>
1610 f=0:ift$<>chr$(27)then dn$=""
:return                               <aj>
1620 char1,1,24,rf$+"datei "+dn$+"
 wird geloescht"+ro$                  <ma>
1630 fori=1tofl:ifdn$=dn$(i)thendn
$(i)=""                               <eh>
1640 ifdn$(i)=""thendn$(i)=dn$(i+1
):dn$(i+1)=""                         <pm>
1650 nexti:fl=fl-1:open15,8,15:pri
nt#15,"s:par."+dn$:print#15,"s:"+d
n$                                    <hi>
1660 print#15,"s:rel.files-liste":
close15:iffl>0then gosub1560          <np>
1670 dn$="":return                    <dn>
1680 forxx=1to2000:next:return        <hd>
1690 b$=chr$(32):b2$=b$+b$            <pc>
1700 b3$=b2$+b$:b$=b3$+b3$+b2$+b2$    <dj>
1710 rf$=chr$(18):ro$=chr$(146):re
turn                                  <eb>
1720 rem ======================       <ff>
1730 rem 12277 bytes memory           <ai>
1740 rem 06348 bytes program          <pn>
1750 rem 00217 bytes variables        <hf>
1760 rem 02037 bytes arrays           <an>
1770 rem 00642 bytes strings          <cp>
1780 rem 03033 bytes free (0)         <mf>
1790 rem (bei 3 datensaetzen)         <mi>
1800 rem ======================       <dl>
```

Arbeitszeit-verkürzung für Mensch und Computer

In der letzten Ausgabe gaben wir Ihnen einige Tips, um Ihre BASIC-Programme schneller zu machen. Wie versprochen, hier die Fortsetzung. Schreiben Sie ein Buch oder eine Doktorarbeit? Dann haben wir ein sehr nützliches Hilfsprogramm für Sie. Zum Schluß noch ein Spiel zur Entspannung.

Wenn ein Programm schneller ablaufen soll, so muß alles vermieden werden, was den Computer unnötig aufhält. Bei einigen Funktionen benutzt der Computer von einer Zahl ohnehin nur den Integer-Teil, also die ganze Zahl ohne die Stellen hinter dem Komma. Daher ist es oft unnötig, die INT-Funktion einzusetzen. Ein Beispiel:

```
10 a=3077.55:b=65.5:poke int(a),int(b)
```

Hier kann in beiden Fällen der INT-Befehl weggelassen werden. Ohne ihn geht es schneller, das Ergebnis ist das gleiche. Die Funktionen, bei denen am häufigsten der INT-Befehl eingesetzt wird, sind POKE, PEEK und CHAR.

IM DUNKELN GEHT MANCHES SCHNELLER

Folgenden Trick werden regelmäßige Leser unserer Zeitschrift sicher kennen, aber der Vollständigkeit halber und für die „Neuen" sei er noch einmal erklärt. Der C16/P4 verbraucht relativ viel Zeit, um den Bildschirm aufzubauen, er ist langsamer als der C64 oder der C16. Dies hat seinen Grund in der Luminanz. Da der C16 nicht nur 16 Farben, sondern diese auch noch in sieben Helligkeitsstufen darstellen kann, braucht das Betriebssystem einige Zeit, um den entsprechenden Hardware-Bausteinen mitzuteilen, wie die einzelnen Zeichen darzustellen sind.
Wird die Bildschirmdarstellung nicht gebraucht, da nur Berechnungen durchgeführt werden, kann der Bildschirm mit

```
POKE 65286,11
```

abgeschaltet werden. Manche Programmteile laufen nun bis zu 30 Prozent schneller. Mit

```
POKE 65286,27
```

wird der Bilschirm wieder sichtbar.

Diesen POKE sollten Sie sich bei der Programmentwicklung auf eine der KEY-Tasten legen, da bei einem Programmfehler die Fehlermeldung ja unsichtbar ist und Blindschreiben ist nicht jedermans Sache.

LASSEN SIE DAS LET BLEIBEN

Viele Programmierer der „alten Schule", die schon auf dem PET oder ähnlichen Computern das Programmieren erlernt haben, verwenden immer noch den LET-Befehl:

```
10 let a=1234
```

Dieser Befehl wird von neueren BASIC-Dialekten zwar noch verstanden, aber nicht mehr benötigt. Er kostet nur zusätzlichen Speicherplatz und etwas Zeit bei der Interpretation. Schreiben Sie also besser nur

```
a=1234.
```

EIN IF KOMMT SELTEN ALLEIN

Vermeiden Sie unnötige IF-Abfragen.
Häufig finden sich mehrere hintereinander stehende IF-Zeilen, die durch eine einzige ersetzt werden können:

```
10 if a=65 then 100
20 if a=66 then 200
30 if a=67 then 300
```

Diese drei Zeilen können Sie so zusammenfassen:

```
10 on a-66 goto 100,200,300
```

Dieses Verfahren ist in der Regel ab drei aufeinanderfolgenden Zahlen sinnvoll. Besonders wirksam ist es, wenn mit dem Joystick ein Spiel gesteuert wird. JOY(1) kann die Zahlen von eins bis acht annehmen (mit Feuer 128 bis 136). Soll bei jeder Stellung in ein anderes Unterprogramm gesprungen werden, so erfolgt die Abfrage:

```
10 on joy(1) gosub 100,200,....
```

Prinzipiell muß die IF-Anweisung, die am häufigsten wirksam wird, am Anfang stehen. Dann sollten die folgenden Zeilen übersprungen werden, damit keine unsinnigen Abfragen erfolgen.

ÜBERBLICK IM BUCHSTABEN-DSCHUNGEL

Mit SCRIPT/PLUS gibt es für den C16 ein sehr gutes Textsystem, mit dem auch größere Werke wie Diplomarbeiten oder Bücher geschrieben werden können. Oft wird dafür ein umfangreiches Stichwort-Register gebraucht. Seine Erstellung ist eine langwierige und langweilige Arbeit, die uns zum Teil der Computer abnehmen soll.
Die folgende Beschreibung unseres Programms TEXTANALYSE gilt nicht nur für SCRIPT/PLUS, sondern für alle Textprogramme, die mit sequentiellen ASCII-Dateien arbeiten. Manche Programme arbeiten mit PRG-Files und Bildschirm-Code. Hier müssen die entsprechenden READ- und WRITE-Befehle abgeändert und der Bildschirm-Code in ASCII-Code umgewandelt werden.
Natürlich kann kein Programm wissen, welche Wörter wichtig sind und deshalb im Register erscheinen sollen. Sie müssen also zunächst die entsprechenden Be-

griffe herausschreiben, sinnvollerweise mit SCRIPT/PLUS. Es genügt, die Suchbegriffe in der Reihenfolge zu schreiben, wie sie gefunden werden. Dies geschieht am besten gleich beim Schreiben des Textes.
Es ist vorteilhaft, daß SCRIPT/PLUS zwei Texte parallel im Speicher halten kann (umschalten mit ESC&J bei 64 KByte), also den eigentlichen Text und das Stichwortregister. Das alphabetische Sortieren übernimmt der Computer.
Damit das Programm seine Aufgabe ordentlich verrichten kann, müssen einige Vorkehrungen getroffen werden:
a) Der Text sollte in der endgültigen Fassung vorliegen.
b) Die einzelnen Seiten müssen mit der Seiten-Ende-Kennung markiert werden (reverser Stern und fp0). Es kann jede beliebige Zeichenfolge als Trennungszeichen zwischen zwei Seiten benutzt werden, sie muß nur in Zeile 110 in u$ geändert werden und darf nicht im Text vorkommen.
c) Alle Textfiles müssen hintereinander in der richtigen Reihenfolge auf einer Diskette stehen und mit TEXT... anfangen (siehe Zeile 150). Das Stichwortverzeichnis muß REGISTER heißen (Zeilen 370 und 650).
d) Sind mehr als 100 Begriffe zu suchen, so muß die Variable mx in Zeile 100 erhöht werden.
Sind diese Bedingungen erfüllt, so kann das Programm gestartet werden. Zunächst wird nachgefragt, ob das Register sortiert werden soll. Dies ist nur nötig, wenn die Begriffe noch nicht in alphabetischer Reihenfolge vorliegen. Auf den Programmablauf hat es keinen Einfluß.

DIRECTORY VOM BILDSCHIRM LESEN

Damit das Programm erkennen kann, wie die einzelnen Textfiles heißen, müssen die Namen aus dem Directory übernommen werden. Sehr aufwendig ist es, die Namen direkt von SPUR 18 auf der Diskette zu lesen. Einfacher geht es, wenn das Programm die Namen zuerst mit DIRECTORY anzeigt (Zeile 150) und sie dann aus dem Bildschirmspeicher liest.
Mit OPEN3,3 wird der Bildschirm als Eingabegerät definiert und der Cursor mit dem CHAR-Befehl an die richtige Stelle gebracht. Das Programm liest mit 'GET#3,a$' das Zeichen, das sich in Cursorposition befindet (Zeilen 160 bis 190).
Noch ein wichtiger Tip: Eine FOR-NEXT-Schleife sollte immer ordnungsgemäß verlassen werden, da, bei einem Sprung aus der Schleife, die Schleife im Stapel (Stack) noch als offen steht. Dabei kann es passieren, daß der für den Stapel verfügbare Speicherplatz irgendwann nicht mehr reicht und ein 'OUT OF MEMORY ERROR' erscheint. Das kann leicht zu Verwirrungen führen, da mit FREE(8) immer noch freier Speicherplatz angezeigt wird.

ERKENNEN VON GANZEN WÖRTERN

Um die Begriffe im Stichwortregister mit denen im Text zu vergleichen, muß sie das Programm im Text erkennen. Gewöhnlich dienen als Trennung zwischen einzelnen Wörtern die Satzzeichen, das Leerzeichen oder das RETURN-Zeichen (CHR$(13)). Daher wird aus den einzelnen Zeichen, die mit GET#2,A$ eingelesen werden, eine Zeichenkette gebildet (Zeilen 250 bis 300), bis A$ mit einem Zeichen aus TR$ (Zeile 140) übereinstimmt (Zeile 260). Jetzt muß dieses Wort mit den Begriffen aus dem Register verglichen werden. Durch die Verwendung eines bereits sortierten Registers läßt sich die Suche beschleunigen. Sie kann mit der Zeile

```
565 if sw$(t)<w$ then t=mx:goto600
```

abgebrochen werden, wenn die noch folgenden Begriffe im Register weiter hinten im Alphabet stehen.
Ein logischer Vergleich mit dem <- oder >-Zeichen erscheint manchem seltsam. Dem Computer ist es aber egal, ob Zahlen oder Buchstaben verglichen werden sollen. Bei einem Größenvergleich von zwei Zeichenketten wird Zeichen für Zeichen verglichen, ob der jeweilige ASCII-Code größer oder kleiner ist. Zum Beispiel ist abc kleiner als abd und ABC größer als abc.
Diese ganzen Vergleiche sind sehr zeitaufwendig, daher sollte das Programm nach dem Eintippen zunächst mit einem kleineren Text getestet werden.
Schneller wird das Programm, wenn alle PRINT-Anweisungen entfernt werden und der Bildschirm abgeschaltet wird. Dies ist jedoch erst beim ausgetesteten Programm ratsam.
In Zeile 300 wird abgefragt, ob das File schon zu Ende ist. Dann ist nämlich die Statusvariable ST größer Null. Wird ein Begriff gefunden, so wird die entsprechende Seitenzahl an bereits früher gefundene Seitenzahlen in SZ$(I) angehängt. Auch das nötige Komma zwischen den Seitenzahlen wird nicht vergessen. Da ein Begriff auf einer Seite oft mehrmals vorkommt, muß überprüft werden, ob diese Seite bereits registriert wurde (Zeile 580).

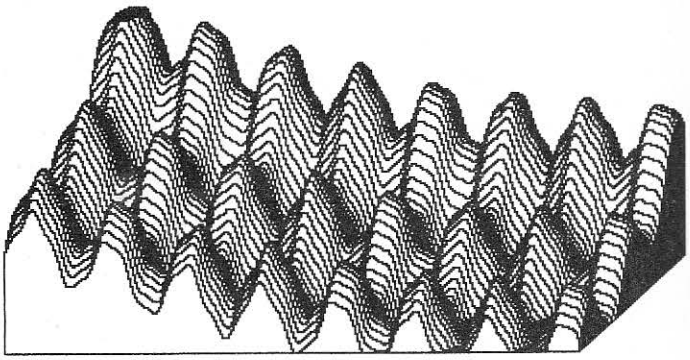
SICHERHEIT IST TRUMPF

Beim Abspeichern wichtiger Daten ist es immer sinnvoll, eine Sicherheitskopie der alten Daten anzufertigen. In Zeile 670 wird der Name des alten Files geändert und das neue File in den Zeilen 680 bis 690 unter dem alten Namen abgespeichert. In den Zeilen 700 bis 710 kann die Sicherheitskopie gelöscht werden. Die bequemen Befehle RENAME und SCRATCH lassen sich hier nicht verwenden, da sie nicht mit Variablen funktionieren.

NACHTRAG ZU C16-P4-SPECIAL 1/88

Im letzten Heft wurde ein kleines Spiel als Beispiel für die Möglichkeit, mit Strings Spielfiguren auf dem Bildschirm darzustellen, besprochen. Leider hat der Druckfehlerteufel das dazugehörige Programm (Würfelstop) unterschlagen. Sie finden es in dieser Ausgabe.
Noch eine kurze Anleitung: Wählen Sie am Anfang eine höhere Zahl für die Geschwindigkeit (zum Beispiel 100). Es erscheinen dann zwei Würfel. Der linke ist unbeweglich, der rechte verändert sich schnell. Stimmen beide Würfelaugen überein, muß eine Taste gedrückt werden. Ziel ist es, möglichst viele richtige Stoppversuche zu erreichen und möglichst wenige Übereinstimmungen zu verpassen.
Sie brauchen ein flinkes Auge, um selbst bei geringer Geschwindigkeit noch Treffer zu erzielen. □

```
10 rem text-analyse ==========c16 <de>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by  r. schmid-fabian   <jp>
50 rem              heidelberg    <kp>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/p4 + floppy        <jh>
90 rem ========================== <jg>
100 gosub720:mx=100:rem trap310   <oi>
110 u$=chr$(128)+"*fp0"           <ob>
120 dim ff$(24),sw$(mx),sz$(mx)   <pp>
130 gosub370                      <pp>
140 tr$=" .,"+chr$(13):rem trennze
ichen                             <md>
150 scnclr:directory"text*":open3,
3                                 <ji>
160 fory=1to23:forx=6to23         <ln>
170 char,x,y,"":get#3,a$:ifa$<>chr
$(34)thenff$(y)=ff$(y)+a$:goto180 <he>
175 x=24:char,0,y,chr$(27)+"q"    <co>
180 if instr(ff$(y),"free.") then
ff$(y)="":hlt=y-1:y=24            <on>
190 nextx,y:printchr$(27)"o"chr$(1
47):close3                       <ll>
200 fori=1to23:printff$(i):next   <ak>
210 for y=1 to hlt:p$=ff$(y)      <nb>
220 sz=sz+1:z$=str$(sz):lz=len(z$) <in>
230 open2,8,2,p$+",s,r"           <oj>
240 printchr$(18)"Analyse von "+p$ <do>
250 fori=1to50000                 <oa>
260 get#2,a$:if instr(tr$,a$)=0 th
en w$=w$+a$:goto300               <nm>
270 if w$=u$ then sz=sz+1:z$=str$(
sz):lz=len(z$):printchr$(18)"Seite
"sz                               <po>
280 rem printw$:rem nur in testpha
se                                <ni>
290 gosub560:w$=""                <lo>
300 ifst=0then nexti              <kf>
310 close2:nexty                  <lh>
320 goto650                       <ml>
330 rem ------------------------- <dk>
340 rem  einlesen des             <co>
350 rem  stichwortregisters       <pp>
360 rem ------------------------- <gg>
370 open2,8,2,"register"+",s,r"   <lk>
380 fori=1tomx:input#2,sw$(i):prin
tsw$(i)                           <ie>
390 if st=0 then nexti            <cl>
400 mx=i:close2                   <mg>
410 scnclr:print"Stichwortregister
 sortieren? (J/N)":getkey a$:if a$
="n" then return                  <pb>
420 rem ------------------------- <gp>
430 rem  sortieren des            <kl>
440 rem  stichwortregisters       <oi>
450 rem ------------------------- <ni>
460 fori=1tomx:a$=sw$(i)          <co>
470 fork=1tomx:b$=sw$(k)          <op>
480 ifa$<b$thensw$(i)=b$:sw$(k)=a$
:a$=b$                            <ld>
490 nextk:printi,sw$(i):nexti     <bc>
500 fori=1tomx:printsw$(i):next   <en>
510 return                        <in>
520 rem ------------------------- <pg>
530 rem  durchsuchung des         <ml>
540 rem  stichwortregisters       <dj>
550 rem ------------------------- <ok>
560 for t=1 to mx:if sw$(t)="" the
n t=mx:goto600                    <kf>
570 if sw$(t)<>w$ then 600        <pa>
580 if sz$(t)="" then sz$(t)=z$:pr
intw$+sz$(t)                      <id>
590 ifright$(sz$(t),lz)<>z$ then s
z$(t)=sz$(t)+","+z$:printw$+sz$(t) <le>
600 nextt:return                  <af>
610 rem ------------------------- <op>
620 rem  abspeichern des          <hp>
630 rem  stichwortregisters       <ik>
640 rem ------------------------- <hh>
650 p$="register"                 <bm>
660 printchr$(147)"Jetzt wird "+ch
r$(34)+p$+chr$(34)+" abgespeichert
 (Taste)":getkeya$                <mn>
670 open1,8,15,"r:"+left$(p$+d$,12
)+".bak+"+"="+p$:close1           <gg>
680 open2,8,2,p$+",s,w"           <dg>
690 fori=1tomx:print#2,sw$(i),sz$(
i):next:close2                    <gk>
700 printleft$(p$+d$,12)+".bak";:i
nput" loeschen (j)";q$:ifq$<>"j"th
enstop                            <el>
710 open1,8,15,"s:"+left$(p$+d$,12
)+".bak"+"="+p$:close1:end        <la>
720 d$=chr$(32):d$=d$+d$+d$       <lj>
730 d$=d$+d$+d$+d$:return         <jb>
740 rem ========================= <jc>
750 rem  p r o g r a m m e n d e  <nf>
760 rem ========================= <jf>
```



WUERFEL-REAKTION

```
10 rem wuerfel-reaktion =======c16 <cj>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by schmid-fabian       <lo>
50 rem                            <pd>
60 rem basic v3.5                 <od>
70 rem ========================== <po>
80 d$=chr$(32):d2$=d$+d$:d3$=d2$+d
$                                 <ek>
90 d5$=d3$+d2$:d$=d5$+d5$         <fp>
100 bs$=chr$(157)+chr$(157)+chr$(1
57)+chr$(17)                      <pj>
110 p1$="Q"+d2$+bs$:pm$=" Q "+bs$:
pr$=d2$+"Q"+bs$                   <ij>
120 p2$="Q Q"+bs$: n$=d3$+bs$     <ek>
130 rem *** wuerfel definieren *** <gj>
140 w$(0)=n$+n$+n$                <ai>
150 w$(1)=n$+pm$+n$               <nm>
160 w$(2)=pr$+n$+p1$              <on>
170 w$(3)=pr$+pm$+p1$             <kc>
180 w$(4)=p2$+n$+p2$              <ea>
190 w$(5)=p2$+pm$+p2$             <pb>
200 w$(6)=p2$+p2$+p2$             <nj>
210 rem ****** bei-spiel ********* <nl>
220 rem ****** reaktion ********** <ek>
230 printchr$(142)chr$(147):char,7
,5,""                             <ia>
240 input"geschwindigkeit";v:gosub
390                               <im>
250 printchr$(147):char,7,5,"taste
 bei uebereinstimmung"            <pl>
260 gosub360:char,15,10,w$(a):wz=a <he>
270 gosub 360:char,20,10,w$(a)    <gc>
280 for t=1 to v:next             <kn>
290 get a$                        <ee>
300 if (a=wz) and (a$="") then nv=
nv+1:gosub390                     <bi>
310 if a$="" then 270             <pn>
320 if a=wz then nr=nr+1:else nf=n
f+1                               <pf>
330 gosub 390                     <ig>
340 getkeya$:goto250              <np>
350 rem ** zufallszahl wuerfeln ** <gc>
360 a=int(rnd(1)*6)+1:if a=wn then
 360                              <po>
370 wn=a:return                   <cb>
380 rem **** anzeigen ************ <lf>
390 char,2,2,d$+d$+d5$            <id>
400 char,2,2,"richtig"+str$(nr)   <ee>
410 char,15,2,"falsch "+str$(nf)  <dn>
420 char,27,2,"verpasst "+str$(nv) <hp>
430 return                        <on>
440 rem ========================== <kf>
450 rem  p r o g r a m m e n d e  <pc>
460 rem ========================== <jk>
```

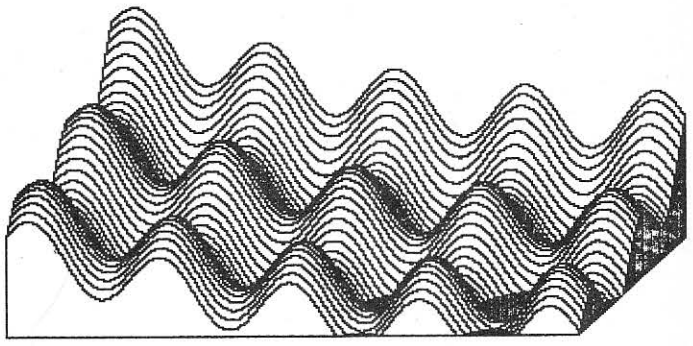
LISTING

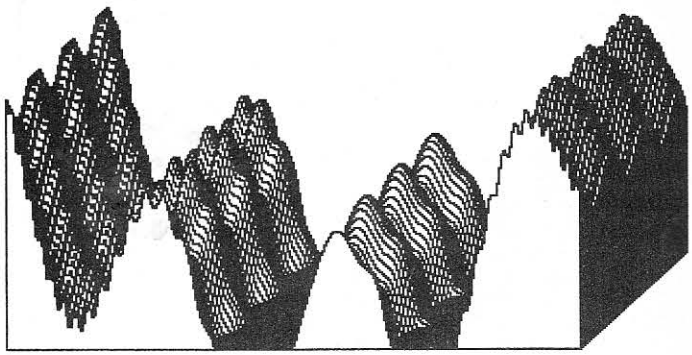
DIR-Printer

Unser DIR-PRINTER hilft Ihnen, die Übersicht über Ihre Programmsammlung zu behalten. Jetzt brauchen Sie nicht mehr lange zu forschen, auf welcher Diskette was zu finden ist.

Inhaltsverzeichnis per Programm, das erübrige sich doch, mag manch einer sagen. Sicher, mit dem CMD-Befehl, gefolgt von DIRECTORY, geht es auch. Nur ergibt dies mitunter ellenlange, und recht unübersichtliche Listen. Auf einer Diskettenhülle finden sie schon gar nicht Platz.
Unser Programm DIR-PRINTER druckt das Inhaltsverzeichnis dreispaltig aus. Blockzahlen werden weggelassen, so daß die Druckbreite etwa der einer Diskettenhülle entspricht. Wenn Sie die Übersicht über den Inhalt Ihrer Disketten behalten wollen, kommt Ihnen dieses Programm sicher gerade recht. Einfach das Directory ausdrucken, auf die Diskettenhülle kleben, und in Zukunft brauchen Sie nicht mehr erst die Diskette ins Laufwerk stecken, um zu sehen, was darauf ist. □

```
10 rem dir-printer============c16 <jd>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by detlef lokay        <hg>
50 rem                            <pd>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/plus4              <ki>
90 rem ========================== <jg>
100 gosub 550                     <bj>
110 printcl$left$(qd$,5)          <bc>
120 printleft$(qr$,10)"**********
****"                             <nf>
130 printleft$(qr$,10)"*
   *"                             <ig>
140 printleft$(qr$,10)"* dir-print
er *"                             <hk>
150 printleft$(qr$,10)"*  copyrigh
t  *"                             <bb>
160 printleft$(qr$,10)"*   d.lokay
   *"                             <bd>
170 printleft$(qr$,10)"*
   *"                             <ia>
180 printleft$(qr$,10)"**********
****"                             <eo>
190 forx=1to1000:next:printcl$    <ba>
200 printc4$c4$c4$c3$c3$"der dir-p
rinter druckt die"                <pm>
210 print"  directory 3-spaltig au
s."                               <ga>
220 printc4$c4$c4$"  bitte entspre
chende disk "                     <kh>
230 print"  einlegen und drucker e
inschalten."                      <pk>
240 printc4$c4$c4$"  zum start bel
iebige taste."                    <fp>
250 geta$:ifa$=""then250          <nn>
260 printcl$                      <og>
270 printleft$(qd$,11)left$(qr$,7)
"ausdruck laueft."                <fa>
280 open2,4:                      <ab>
290 open1,8,0,"$0"                <dj>
300 get#1,a$,b$                   <fn>
310 gosub470:b=b+1                <ff>
320 get#1,b$:if st<>0 then 360    <cm>
330 if b$<>chr$(34) then 320      <ng>
340 get#1,b$:if b$<>chr$(34)then p
rint#2,b$;:goto340                <of>
350 if st=0 then 300              <kl>
360 print#2,chr$(13);chr$(13)     <gj>
370 print#2:close1:close2         <ma>
380 printcl$:printleft$(qd$,5)left
$(qr$,6)"fertig."                 <pc>
390 printc4$c4$c3$c3$c3$"weitere a
usdrucke j/n ?":print:print       <kg>
400 geta$:ifa$=""then400          <ij>
410 ifa$<>"j"thenend              <hp>
420 ifa$="j"thenprint"   disk wech
seln."                            <co>
430 print"   =============="      <ip>
440 printc4$"   weiter mit tastend
ruck."                            <il>
450 geta$:ifa$=""then450          <lk>
460 run 280                       <ja>
470 ifb<1thenreturn               <kb>
480 ifb=1thenprint#2,chr$(13)"====
============";                   <ml>
490 a=a+1                         <jo>
500 ifa=1thenprint#2,chr$(16);"17"
;"* ";                            <kd>
510 ifa=2thenprint#2,chr$(16);"35"
;"* ";                            <kh>
520 ifa=3thenprint#2              <ne>
530 ifa=3thena=0                  <gb>
540 return                        <mk>
550 rem nachspann ================ <gh>
560 rem * farbcodes/steuercodes  * <og>
570 c4$=chr$(017):c3$=chr$(029)   <gj>
580 cl$=chr$(147)                 <ld>
590 rem ********* zeichenfolgen  * <jp>
600 for q=1 to 40                 <eh>
610 qd$=qd$+c4$:qr$=qr$+c3$       <mp>
620 next q                        <an>
630 return                        <ho>
640 rem ========================== <np>
650 rem 12277 bytes memory        <dk>
660 rem 01787 bytes program       <ln>
670 rem 00056 bytes variables     <fj>
680 rem 00000 bytes arrays        <ga>
690 rem 00408 bytes strings       <pp>
700 rem 10026 bytes free (0)      <bn>
710 rem ========================== <ne>
```



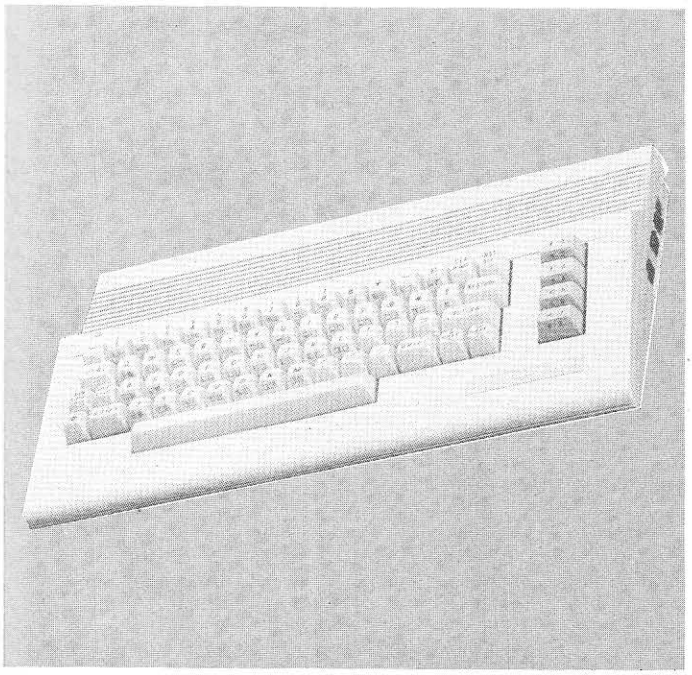


DER COMPUTER ALS LERNINSTRUMENT

Zehnfinger-system

Im Gegensatz zur Schreibmaschine kann ein Computer Aufgaben stellen und Leistungen auswerten. Was liegt also näher, als ihn zum Erlernen des Maschinenschreibens zu benützen?

Dem Lernenden werden verschiedene Übungen angeboten. Er kann sich an Fingerübungen heranmachen, an längere Buchstabenfolgen, an sinnvolle Wörter, oder sich an schwierigen Texten versuchen. Die Fingerübungen gehen von kleinem Buchstabenvorrat zu größerem über. In den komplizierteren Versuchen wird also auch der Buchstabensatz der vorangegangenen gefordert. Dadurch kommen aber die schwierigeren, weil seltener vorkommenden, Zeichen etwas zu kurz.
Ein weiteres Manko ist, daß keine deutsche Tastatur mit vertauschtem Y und Z simuliert wurde. Das ist aber nicht tragisch, es ist mit unserer Tastaturanpassung leicht zu beheben. Verbesserungsvorschläge und Anregungen zu diesem Programm werden gerne entgegengenommen.



SCHREIBMASCHINENKURS

```
10 rem schreibmaschinenkurs ===c16 <gp>
20 rem (p) 09/87 commodore welt    <bf>
30 rem ========================== <ng>
40 rem (c) by klaus freitag    2.0 <ea>
50 rem klaus freitag               <lh>
60 rem                             <ah>
70 rem (v) by bernd welte      3.5 <ae>
80 rem c16/116/plus4               <ki>
90 rem ========================== <jg>
100 cl$=chr$(147):wh$=chr$(005)    <pl>
110 rn$=chr$(018):rf$=chr$(146)    <hm>
120 c4$=chr$(017):s1$=chr$(032)    <ag>
130 a2$=s1$+s1$:a3$=a2$+s1$        <gj>
140 a4$=a3$+s1$:a5$=a4$+s1$        <oe>
150 d2$=c4$+c4$:d3$=d2$+c4$        <cf>
160 d4$=d3$+c4$:d5$=d4$+c4$        <al>
170 ka$=chr$(064):c3$=chr$(029)    <mb>
180 w2$=c3$+c3$:w3$=w2$+c3$        <kj>
190 w4$=w3$+c3$:w5$=w4$+c3$        <nd>
200 he$=chr$(019):s5$=chr$(101)    <da>
210 key 1,chr$(133):key2,chr$(134) <mf>
220 key 3,chr$(135):key4,chr$(136) <bl>
230 key 5,chr$(137):key6,chr$(138) <fd>
240 key 7,chr$(139):key8,chr$(139) <jm>
250 dimc(15),d$(40),w$(50),r$(50)  <ie>
260 aa$=cl$+wh$+rn$+a5$+"c16 schre
ibmaschinenkurs"+a5$+a5$+s1$       <pp>
270 poke65305,0:poke65301,0        <nd>
280 printaa$                       <aj>
290 printd2$rn$"f1"rf$" - fingerue
bung"                              <ia>
300 printc4$rn$"f2"rf$" - schreibu
ebung"                             <ca>
310 printc4$rn$"f3"rf$" - wortuebu
ng"                                <al>
320 printc4$rn$"f4"rf$" - wortuebu
ng fuer fortgeschrittene"          <cb>
330 printc4$rn$"f5"rf$" - schwieri
ger text"                          <ph>
340 printc4$rn$"f6"rf$" - ende"    <ck>
350 printd3$rn$"f8"rf$" - erklaeru
ngen"                              <nh>
360 geta$:ifa$=""goto360           <gd>
361 onasc(a$)-132goto450,810,900,1
780,1200,1650,2160                 <he>
440 goto360                        <en>
450 printaa$                       <km>
460 printc4$"bitte buchstabenkombi
nation waehlen"                    <el>
470 printd2$rn$"1"rf$"  - asdejkl" <bl>
480 printrn$"2"rf$"  - eidk"       <nl>
490 printrn$"3"rf$"  - wosl"       <ie>
500 printrn$"4"rf$"  - fghj"       <gb>
510 printrn$"5"rf$"  - fjvm"       <ca>
520 printrn$"6"rf$"  - fjbn"       <lp>
530 printrn$"7"rf$"  - tfj"        <ch>
540 printrn$"8"rf$"  - dkc"        <ie>
550 printrn$"9"rf$"  - slx"        <dp>
560 printrn$"10"rf$" - qpa"        <eb>
570 printrn$"11"rf$" - rufj"       <nf>
580 inputa:ifa<1ora>11goto580      <jp>
590 op=a*10:ru=0:ri=0              <gl>
600 restore                        <ea>
610 ad=int(op*rnd(1)+1)            <jc>
620 fori=1toad:readfi$:next        <kl>
630 printaa$                       <ph>
640 printhe$d5$d5$w5$w5$fi$        <cf>
```

```
650 printd5$;:inputan$                                  <ei>
660 iffi$=an$thenri=ri+1                                <ae>
670 ru=ru+1:ifru=16thengoto690                          <hl>
680 goto600                                             <fo>
690 m=16:goto1700                                       <ml>
700 data"fsda","jlk","fdsa","asdf"
,"lkj","asdf","fdsa","jkl","dsa","
sdf"                                                    <ek>
710 data"ded","def","des","kik","k
ij","kil","ded","kik","ede","iki"                       <lc>
720 data"sws","swd",swa","lol","lo
k","loo","sws","lol","wsw","olo"                        <pi>
730 data"fgf","fgd","fgs","jhj","j
kh","jhl","fgf","jhj","gfg","hjh"                       <je>
740 data"fvf","fvd","fvs","jmj","j
mk","jml","fva","mjj","dvs","mlj"                       <bb>
750 data"fbf","fbd","fbs","jnj","j
nk","jnl","fbf","jnj","fbs"                             <oc>
760 data"ttf","tft","ftg","ujj","h
jh","hjj","ftf","jtj","tft","tfg"                       <ai>
770 data"dcd","dcf","dcs","kck","k
cl","ckk","dcf","kcj","dcv","kcm"                       <cg>
780 data"sxs","sxd","sxa","lxl","l
xx","lxk","sxs","lxl","xsx",xlx"                        <lb>
790 data"aqa","aqs","aqd","qas","q
af","aqa","ppl","ppk","kpj","kpl"                       <bd>
800 data"frf","frd","frs","juj","j
uk","jul","drf","juj","frd","juk"                       <pd>
810 printaa$:an=90                                      <ai>
820 forii=1to10:printaa$                                <be>
830 fori=1to15                                          <le>
840 c(i)=int(an*rnd(1)+1)                               <hj>
850 ifc(i)<65goto840                                    <gn>
860 f$=f$+chr$(c(i)):nexti                              <ah>
870 printhe$d5$d5$w5$w5$f$                              <jj>
880 printd4$:inputan$:ifan$=f$then
ri=ri+1                                                 <cp>
890 f$="":nextii:m=10:goto1700                          <na>
900 printaa$                                            <pa>
910 d$(1)="fad":d$(2)="all":d$(3)=
"als"                                                   <nk>
920 d$(4)="las":d$(5)="das":d$(6)=
"seid"                                                  <no>
930 d$(7)="riff":d$(8)="rede"                           <da>
940 d$(9)="erde":d$(10)="felge"                         <ka>
950 d$(11)="hilfe":d$(12)="orgel"                       <hk>
960 d$(13)="fuhre":d$(14)="klage"                       <il>
970 d$(15)="narbe":d$(16)="nobel"                       <ae>
980 d$(17)="leber":d$(18)="sonne"                       <gj>
990 d$(19)="pappe":d$(20)="quart"                       <cb>
1000 d$(21)="pulver":d$(22)="yard"                      <dh>
1010 d$(23)="hyaene":d$(24)="mysti
k"                                                      <lh>
1020 d$(25)="hypothek":d$(26)="cit
y"                                                      <pe>
1030 d$(27)="loyal":d$(28)="typhus
"                                                       <fo>
1040 d$(29)="analysen":d$(30)="spu
le"                                                     <hp>
1050 d$(31)="quarze":d$(32)="empor
"                                                       <nh>
1060 d$(33)="wispern":d$(34)="paul
"                                                       <ha>
1070 d$(35)="olymp":d$(36)="symbol
"                                                       <bg>
1080 d$(37)="xylophon":d$(38)="eic
ht"                                                     <lo>
1090 d$(39)="wirkt":d$(40)="wichti
g"                                                      <hf>
1100 printd2$"wieviele worte moech
ten sie schreiben"                                      <pl>
1110 printd2$:inputwi                                   <al>
1120 printaa$:fori=1towi:printaa$                       <kc>
1130 as=int(rnd(1)*40)+1                                <ci>
1140 printhe$d5$d5$w5$w5$d$(as)                         <gb>
1150 printd5$                                           <je>
1160 inputan$                                           <li>
1170 ifan$=d$(as)thenri=ri+1                            <kf>
1180 next                                               <md>
1190 m=wi:goto1700                                      <dk>
1200 w$(1)="wie eine":w$(2)="leben
strotzende"                                             <lo>
1210 w$(3)="gluckhenne":w$(4)="sta
nd die mutter"                                          <am>
1220 w$(5)="mitten unter ihrer"                         <fl>
1230 w$(6)="bluehenden":w$(7)="kin
derschar."                                              <ko>
1240 w$(8)="sie hatte ein brot"                         <mn>
1250 w$(9)="gegen den dicken leib
gepresst"                                               <cb>
1260 w$(10)="und schnitt":w$(11)="
mit einem"                                              <he>
1270 w$(12)="sichelartigen messer"                      <fo>
1280 w$(13)="grosse scheiben ab,"                       <fm>
1290 w$(14)="die jedes mal"                             <cc>
1300 w$(15)="sofort"                                    <if>
1310 w$(16)="von den kindern"                           <cj>
1320 w$(17)="weggeschnappt und"                         <pp>
1330 w$(18)="in die taschen"                            <cc>
1340 w$(19)="gestopft wurden."                          <bh>
1350 w$(20)="horieneken"                                <pb>
1360 w$(21)="nahm ihr koerbchen"                        <fo>
1370 w$(22)="mit dem strickzeug"                        <jm>
1380 w$(23)="und den schulbuechern
"                                                       <ah>
1390 w$(24)="-erst zog sie noch"                        <dm>
1400 w$(25)="fons die struempfe in
 die hoeh',"                                            <kg>
1410 w$(26)="knuepfte sarel die ho
se zu,"                                                 <dp>
1420 w$(27)="putzte ludwig die nas
e-"                                                     <eo>
1430 w$(28)="und nachdem"                               <gj>
1440 w$(29)="die mutter gemahnt ha
tte:"                                                   <dh>
```

```
1450 w$(30)="dass ihr mir"                                    <jg>
1460 w$(31)="schnurstracks in die
schule geht,"                                                 <fp>
1470 w$(32)="hoert ihr,"                                      <gh>
1480 w$(33)="ihr bengels!"                                    <li>
1490 w$(34)="schoss die ganze band
e"                                                            <gi>
1500 w$(35)="zur tuer hinaus,"                                <gd>
1510 w$(36)="durch den blumengarte
n,"                                                           <aj>
1520 w$(37)="ueber den breiten fel
dweg,"                                                        <bi>
1530 w$(38)="geradewegs auf die gr
osse goldige sonne zu,"                                       <gg>
1540 w$(39)="die drueben hinter de
n erlenstaemmen"                                              <pc>
1550 w$(40)="in einem maechtigen s
trahlenfeuer"                                                 <ia>
1560 w$(41)="heraufgestiegen kam." <eh>
1570 printaa$:fori=1to41:printaa$  <ej>
1580 printhe$d5$d5$w$(i)                                      <np>
1590 forii=1to1000:nextii:printaa$ <kh>
1600 printhe$d5$d5$                                           <lg>
1610 inputab$                                                 <ng>
1620 ifab$=w$(i)thenri=ri+1                                   <oe>
1630 next                                                     <ek>
1640 m=41:goto1700                                            <dn>
1650 printaa$                                                 <ph>
1660 printd5$c4$w5$w5$"bitte daran
 denken:"                                                     <ab>
1670 printd2$w5$"nur uebung macht
den meister!"                                                 <if>
1680 printw5$;:fori=0to26:prints5$
;:next:prints5$                                               <ao>
1690 end                                                      <mb>
1700 printaa$                                                 <bo>
1710 printd3$"sie haben";ri                                   <mb>
1720 printd3$"von";m                                          <ch>
1730 printd3$"moeglichen punkten." <fb>
1740 printd5$d5$w5$w5$w3$rn$"menue
->taste"                                                      <ng>
1750 geta$:ifa$=""goto1750                                    <dp>
1760 ri=0                                                     <ld>
1770 goto270                                                  <lg>
1780 r$(1)="abbreviation":r$(2)="b
abylonien"                                                    <pd>
1790 r$(3)="caballero":r$(4)="dada
istisch"                                                      <ef>
1800 r$(5)="eau de cologne":r$(6)=
"facetenauge"                                                 <pc>
1810 r$(7)="gabardinemantel":r$(8)
="habeaskorpusakte"                                           <na>
1820 r$(9)="iberoamerikanisch":r$(
10)="kabinettformat"                                          <ol>
1830 r$(11)="labiovelar":r$(12)="m
achination"                                                   <oi>
1840 r$(13)="nebukadnezar":r$(14)=
"objektivation"                                               <jj>
1850 r$(14)="pachulke":r$(15)="qua
dragesima"                                                    <cj>
1860 r$(16)="rachmaninow":r$(17)="
sabbatstille"                                                 <in>
1870 r$(18)="tachygraphie":r$(19)=
"ultima ratio"                                                <mk>
1880 r$(20)="vakuumverpackt":r$(21
)="waffenstillstandslinie"                                    <ep>
1890 r$(22)="xanthin":r$(23)="yama
shita"                                                        <ko>
1900 r$(24)="zapfenzieher":r$(25)=
"allochthon"                                                  <mi>
1910 r$(26)="ballyhoo":r$(27)="cha
uffieren"                                                     <ei>
1920 r$(28)="couture":r$(29)="couv
ert"                                                          <ba>
1930 r$(30)="derogation":r$(31)="d
ialysieren"                                                   <fn>
1940 r$(32)="differenziertheit":r$
(33)="distributionsformel"                                    <ci>
1950 r$(34)="dreiviertel":r$(35)="
dystrophiker"                                                 <lb>
1960 r$(36)="egozentriker":r$(37)=
"endothel"                                                    <ke>
1970 r$(38)="enzephalitis":r$(39)=
"explosionssicher"                                            <fm>
1980 r$(40)="folgendergestalt":r$(
41)="galanthomme"                                             <nf>
1990 r$(42)="garnisonieren":r$(43)
="glyptik"                                                    <ep>
2000 r$(44)="guerillakrieg":r$(45)
="hypothese"                                                  <eb>
2010 r$(46)="ikonostase":r$(47)="i
mpressionabel"                                                <hp>
2020 r$(48)="in dulci jubilo":r$(4
9)="interlinearglosse"                                        <gm>
2030 r$(50)="isochromasie"                                    <ii>
2040 printaa$:printd2$"wieviele wo
rte moechten sie schreiben"                                   <kl>
2050 printd2$:inputwi                                         <dl>
2060 printaa$                                                 <lj>
2070 fori=1towi:printaa$                                      <mm>
2080 as=int(rnd(1)*40)+1                                      <bb>
2090 printhe$d5$d5$w5$w5$r$(as)                               <kh>
2100 forii=1to2500:next:printaa$                              <ep>
2110 printd5$                                                 <mi>
2120 inputan$                                                 <oo>
2130 ifan$=r$(as)thenri=ri+1                                  <fh>
2140 next                                                     <ek>
2150 m=wi:goto1700                                            <ig>
2160 printaa$:print                                           <ol>
2170 printrn$"fingeruebung"rf$" es
 ist eine kombination"                                        <nl>
2180 print"aus der vorgewaehlten b
uchstabenreihe"                                               <cn>
2190 print"nachzuschreiben.die ueb
ungen sind auf-"                                              <kp>
2200 print"einander aufgebaut,d.h.
:wenn sie z.b."                                               <dl>
```

SCHREIBMASCHINENKURS

```
2210 print"uebung 5 waehlen,sollte
n sie die"                          <cn>
2220 print"uebungen 1 - 4 beherrsc
hen.                                <pn>
2230 printd2$rn$"schreibuebung"rf$
" ich zeige ihnen hier"             <fj>
2240 print"eine buchstabenkombinat
ion aus 15 zu-"                     <ic>
2250 print"faellig ausgewaehlten z
eichen,die sie"                     <ne>
2260 print"nachschreiben sollen.    <ng>
2270 printd3$a5$a5$s1$rn$"bitte ta
ste"rf$                             <hn>
2280 getp$:ifp$=""goto2280          <og>
2290 printaa$:print                 <ij>
2300 printrn$"wortuebung"rf$" es s
ind einfache worte"                 <mn>
2310 print"nachzuschreiben."        <en>
2320 printd2$rn$"wortuebung fuer f
ortgeschrittene"rf$                 <aa>
2330 print"es sind schwierige wort
e nachzu-"                          <ef>
2340 print"schreiben,die ausserdem
 nur kurz zu"                       <hb>
2350 print"sehen sind."             <la>
2360 printd2$rn$"schwieriger text"
rf$" wie wortuebung "               <cc>
2370 print"fuer fortgeschrittene,j
edoch mit"                          <bk>
2380 print"zusammenhaengendem text
"                                   <kd>
2390 printd3$a5$a4$rn$"bitte taste
"rf$                                <jo>
2400 getp$:ifp$-""goto2400          <oc>
2410 printaa$:print                 <jm>
2420 print"sie sollte versuchen,al
le uebungen"                        <fl>
2430 print"blind nach dem 10-finge
r-system zu"                        <oh>
2440 print"schreiben."              <bb>
2450 printc4$"es ist ueberigens be
sser,jeden tag"                     <gj>
2460 print"ein paar minuten zu ueb
en,als einmal"                      <hn>
2470 print"in der woche einige stu
nden."                              <pj>
2480 printd5$d3$w5$w5$rn$"bitte ta
ste"                                <nl>
2490 getp$:ifp$=""goto2490          <mf>
2500 goto270                        <ig>
2510 rem ========================== <el>
2520 rem 12277 bytes memory         <fj>
2530 rem 08223 bytes program        <kj>
2540 rem 00168 bytes variables      <ob>
2550 rem 00537 bytes arrays         <hc>
2560 rem 00456 bytes strings        <el>
2570 rem 02893 bytes free (0)       <fn>
2580 rem ========================== <mh>
```

FUNKTIONS-GRAPHEN

```
10 rem funktionen ============c16 <mh>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by michael grobe        <ie>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ========================== <km>
100 scnclr:graphic1,1:graphic0:got
o420                               <fm>
110 scnclr:print:print:print       <ed>
120 printtab(8)"funktionsgraphen c
16"                                <mk>
130 printtab(8)"-----------------
--"                                <im>
140 print:print:print              <ha>
150 printtab(9)"von michael grobe
"                                  <gk>
160 print"-----------------------
----------------":print            <mc>
170 print"genauigkeit"spc(29)"(0.1
 oder 0.01 oder 0.001):";:inputk   <fn>
180 ifk<>0.1andk<>0.01andk<>0.001t
hen170                             <jn>
190 print"start x"spc(2)"(-16 bis
16)"spc(5)":";:inputxs:ifxs<-16orx
s>16then190                        <fo>
200 print"ziel"spc(2)"x"spc(2)"(-1
6 bis 16)"spc(5)":";:inputzz:ifzz<
-16orzz>16then200                  <hj>
210 ifzz<xsthenprint"ziel x muss g
roesser sein als start x !":goto20
0                                  <jm>
220 print"loeschen (0=n 1=j)"spc(8
)":";:inputl:ifl<>1andl<>0thengoto
220                                <nj>
230 x=xs:y=0                       <cd>
240 graphic1,1                     <gh>
250 draw1,0,100to319,100           <ld>
260 draw1,160,0to160,199           <mj>
270 fori=1to20                     <oi>
280 draw 1,159,200-10*ito161,200-1
0*i:next                           <bd>
290 fori=0to31                     <nn>
300 draw1,10*i,99to10*i,101:next   <ji>
310 ifx>zzthen410                  <ep>
320 rem ========================== <il>
330 y=5*sin(x)                     <dc>
340 rem ========================== <oc>
350 a=160+(x*10)                   <bh>
360 b=100-(y*10)                   <ig>
370 ifb<=0then390                  <oe>
380 draw 1,a,b                     <ah>
390 x=x+k                          <ad>
400 goto310                        <me>
410 char 1,1,1,"ok":getkeym$:char1
,1,1,"  ":graphic0                 <pn>
```

FUNKTIONS-GRAPHEN

```
420 scnclr:list 330:printchr$(145)
"run110":poke239,2:poke1319,19:pok
e1320,17:end                          <ch>
430 rem ======================== <kh>
440 rem  p r o g r a m m e n d e  <la>
450 rem ======================== <ia>
```

Funktions-Graphen

Mit Hilfe dieses Programms können Graphen mathematischer Funktionen auf dem Bildschirm innerhalb eines Koordinaten-Systems dargestellt werden.

Wenn Sie das Programm (hoffentlich fehlerfrei) abgetippt und mit RUN gestartet haben, erscheint kurzfristig die Kernaussage auf dem Bildschirm, nämlich die mathematische Formel in Zeile 330:

Y = SIN (X) * 6

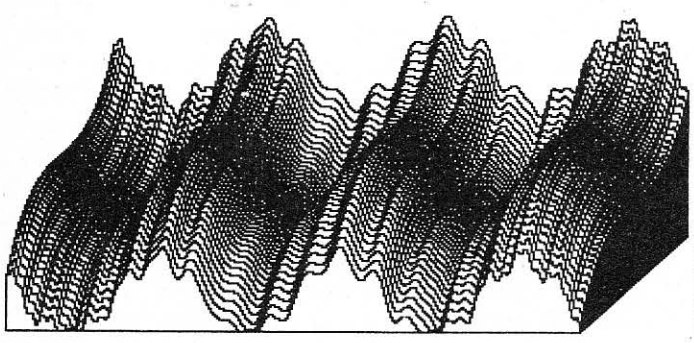
Diese Funktion bewirkt, daß Sie solche Kurven auf Ihrem Monitor überhaupt darstellen können.
Diese Formel können Sie natürlich nach Belieben ändern, indem Sie den Cursor zu dieser Zeile bewegen.
Danach starten Sie das Hauptprogramm mit dem auf dem Bildschirm angezeigten RUN 110, indem Sie das mit der RETURN-Taste bestätigen.
Die folgenden INPUT-Eingaben fragen Sie nun nach gewissen Optionen wie:

Genauigkeit (zwischen 0.1 bis 0.001)

Start X-Position (horizontal), *Ziel X-Position* (bei diesen Eingaben sind Werte zwischen −16 bis +16 möglich).

Löschen (0 bedeutet, daß der vorher aufgezeigte Bildschirm erhalten bleibt, damit können mehrere übereinanderliegende Graphen angezeigt werden), 1 löscht den Monitor, bevor ein Graph gezeichnet wird.

Das aufgeteilte Koordinatennetz im Hires-Modus hat den Mittelpunkt X=160, Y=100.
Eine Einheit enthält zehn Bildschirmpunkte und



reicht in der horizontalen (X-)Richtung von −16 bis +16, in der vertikalen (Y-)Richtung von −10 bis +10.
Aus Ihrer Eingabe bei Start X wird nun das erste Y errechnet, danach der X- und Y-Wert mit 10 multipliziert (wegen der Einteilung der Einheiten). Der zu den Werten gehörige Bildpunkt wird nun gezeichnet.
Die angegebene Genauigkeit wird jetzt zu dem aktuellen X-Wert addiert, zur neuen Berechnung springt jetzt das Programm zum entsprechenden Listing-Teil zurück. Danach wird der Graph auf den Bildschirm gezeichnet, den Abschluß dieser Tätigkeit zeigt Ihnen der Computer mit der Meldung OK an.
Nun müssen Sie irgendeine Taste betätigen, um wieder an den Programmbeginn zu gelangen. Damit haben Sie erneut die Möglichkeit, die besagte Zeile 330 mit einer anderen Funktion zu beschreiben oder einen Graphen mit anderen Werten zu definieren.

EINGABE-HINWEISE

Bei kleiner Genauigkeit (0.1) wird der Graph „löcherig" gezeichnet, dafür aber runder, weil weniger Werte berechnet werden müssen.
Bei der nächsten Steigerung (0.01) präsentiert sich der Graph schon weniger „durchlöchert", dafür aber eckiger.
Die Genauigkeit „0.01" sollte nur bei sehr steil ansteigenden Graphen (beispielsweise TAN) eingegeben werden, da die Berechnung hierbei sehr lange dauert.

Michael Grohe/hb □

Gallows-Galgenraten

Es ist schon erstaunlich, was ein Zwölfjähriger auf seinem Rechner zustande bringt. Die Spielidee ist zwar nicht neu, jedoch wurde sie hier mit guter Grafik und mit gutem Sound realisiert.
Gallows ist eine Version des bekannten Spieles „Galgenraten". Der Computer versucht, Sie an den Galgen zu bringen, indem er sich ein Wort ausdenkt, von dem er glaubt, daß Sie es nicht erraten können. Ein Kampf um Bildschirm-Leben oder -Tod beginnt.
Nach dem Starten will der Computer Ihren Namen wissen. Jetzt können Sie zwischen drei Schwierigkeitsstufen wählen: 10, 5 oder 3 Fehlversuchen.
Entscheiden Sie dann, ob Sie die Spielregeln sehen wollen. Danach beginnt das eigentliche Spiel: Sie sehen eine Hintergrundgrafik mit Musikunterlegung.
Im unteren Teil des Bildes erscheinen so viele Striche, wie das zu erratende Wort Buchstaben besitzt. Geben Sie einen Buchstaben ein, der im Wort vorkommt, so meldet sich der Computer mit einem aufmunternden, hohen Ton und setzt den Buchstaben an die richtige Stelle. Haben Sie jedoch falsch geraten, ertönt ein tiefes Knurren und Ihr Gerät zeichnet einen Teil vom Galgen. Wenn Sie schließlich und endlich das ganze Wort herausbekommen haben, teilt Ihnen die Zeitung „Gallows News" mit, daß Sie dem Tode entronnen sind. Im andern Fall erwartet Sie nur noch die Nachricht von Ihrem schrecklichen Ende. Also geben Sie acht, Sie hängen schneller, als Sie glauben!

Daniel Hanelt

```
10 rem gallows================c16 <jg>
20 rem (p) commodore welt team  <ho>
30 rem ======================== <ng>
40 rem (c) by daniel hanelt     <ep>
50 rem                          <pd>
60 rem                          <ah>
70 rem basic v3.5               <nl>
80 rem c16/116/plus4            <ki>
90 rem ======================== <jg>
100 poke55,0:poke56,56:gosub2220:u
e=1:r=20                        <id>
110 gosub 1290:gosub1310:gosub1320 <gj>
120 color 4,7,2                 <ac>
130 gosub 1660                  <ch>
140 printleft$(qd$,7)b4$b4$wh$" ih
r name ";:inputna$              <ip>
150 gosub 1660                  <kk>
160 printc4$c4$b4$b3$ye$b2$"hallo
"wh$na$                         <kb>
170 printbr$c4$b$" waehle bitte !!
!"                              <la>
180 printc4$c4$b2$cy$rn$"1."rf$wh$
" profi i"lg$" (verbesserungsfaehi
g)"                             <jn>
190 printc4$b2$pu$rn$"2."rf$wh$" p
rofi ii"bl$" (schon ganz gut)   <bf>
200 printc4$b2$re$rn$"3."rf$wh$" p
rofi iii"lb$b2$"(mensch super !!!)
"                               <ge>
210 getkey sch :ifsch <1orsch >3th
en150                           <op>
220 gosub 1660:printwh$         <no>
230 printc4$c4$b3$b3$"kennen sie d
ie"                             <mi>
240 printc4$c4$b$b2$"spielregeln s
chon (j/n)";:getkeygf$          <if>
250 ifsch=1thenhj=10:elseifsch=2th
enhj=5:elseifsch=3thenhj=3      <ka>
260 ifgf$="j"then 580           <mp>
270 gosub1660:gosub1330:printc4$c4
$" "wh$"gallows ist eine version d
es bekannten"                   <af>
280 printc4$" spieles 'galgenraten
'. sie muessen"                 <kj>
290 printc4$" ein vom computer erd
achtes wort durch"              <ac>
300 printc4$" eingeben von einzeln
en buchstaben"                  <in>
310 printc4$" erraten. kommt ein b
uchstabe im wort"               <gm>
320 printc4$" nicht vor, so zeichn
et der computer"                <bj>
330 printc4$" eins von"hj "teilen
vom galgen."                    <go>
340 printc4$c4$b5$b4$"ABCD"rn$" ta
ste "rf$"DCBE"                  <bo>
350 getr$:ifr$=""then350        <ld>
360 printcl$:gosub 1660:printwh$c4
$c4$" haben sie also"hj "fehlversu
che gemacht"                    <on>
370 printc4$" teilt ihnen der comp
uter mit, dass"                 <ji>
380 printc4$" sie verloren haben.
es kommt fuer"                  <da>
390 printc4$" sie nun darauf an, m
it weniger als"hj               <ib>
400 printc4$" fehlversuchen, falls
 sie nicht am "                 <hd>
410 printc4$" galgen landen wollen
, das wort zu "                 <ic>
420 printc4$" erraten.          <fp>
430 printc4$c4$b5$b4$"ABCD"rn$" ta
ste "rf$"DCBE"                  <do>
440 getr$:ifr$=""then440:elsegoto
580                             <dl>
450 rem *** music ***           <md>
460 restore530:vol 8            <pn>
470 for te=1to18                <kd>
480 read w,e:poke 239,0         <cm>
490 ifw=0then510:else sound 1,w,e:
sound 2,w+1,e                   <ln>
500 next                        <gn>
510 return                      <in>
520 rem *** data ***            <mi>
530 data 854,10,1016,20,854,20,770
,20                             <pm>
540 data 810,20,1016,20,810,20,770
,20                             <pb>
550 data 854,20,810,20,770,20,1016
,20                             <fk>
560 data 854,10,1016,7,854,50,881,
10                              <mb>
570 data 854,10,0,1             <hl>
580 scnclr:gosub1330:gosub 1660:pr
int:gosub 1750:gosub 1910       <hj>
590 char 1,0,23,"": print" wortX "
;                               <dm>
600 restore1160:qe=int(rnd(1)*100)
+1                              <pe>
610 for t=0toqe:readf$:next:printc
4$;                             <nc>
620 forp=1to len(f$)            <bl>
630 print"Y"c3$;                <pe>
640 next:gosub 460              <ge>
650 char 1,21,23,"(c) daniel hanel
t"+wh$                          <gm>
660 s=0:getj$:ifj$=""then660    <pd>
670 h(ue)=asc(j$)-64:fortr=0to10:i
fue=trthen680:elseifh(tr)=h(ue)the
n1950:elsenext                  <nc>
680 if j$=mid$(f$,1,1)thens=0.5  <cd>
690 if j$=mid$(f$,2,1)thens=2    <bg>
700 if j$=mid$(f$,3,1)thens=4    <co>
710 if j$=mid$(f$,4,1)thens=6    <hn>
720 if j$=mid$(f$,5,1)thens=8    <pj>
730 if j$=mid$(f$,6,1)thens=10   <cn>
```

```
740 if j$=mid$(f$,7,1)thens=12 <ce>
750 if j$=mid$(f$,8,1)thens=14 <ln>
760 if j$=mid$(f$,9,1)thens=16 <lj>
770 if j$=mid$(f$,10,10)thens=18 <hd>
780 if s=0then1950 <fj>
790 poke 3999+s,h(us) <mo>
800 ue=ue+1 <pa>
810 foryp=700to715:sound 1,yp,1:ne
xt <ef>
820 q=q+1:if p-1=qthen840 <ec>
830 goto 660 <mn>
840 vol8:for yu=1to500:next:fort=2
00to1000step+15:sound1,t,5:sound 2
,t+10,5:next <ek>
850 foryq=1to2000:next:scnclr:gosu
b 1660:color 4,1:poke 239,0 <fg>
860 printb$b5$"D"f1$"n e w s"fo$"D
" <nk>
870 printb$b5$"DDDDDDDDD" <ig>
880 printc4$wh$b2$"und wieder entr
ann ein mensch namens " <bl>
890 jl=len(na$):jl=20-jl/2:char 1,
jl,10,na$:char 1,0,11,"" <cb>
900 printc4$b$" dem schrecklichen" <mh>
910 printc4$b$b5$"galgentod" <dn>
920 printc4$b$"da er auf das wortX
"b2$ <cc>
930 printc4$b$b5$b2$rn$f1$f$rf$b3$ <kl>
940 printc4$b$b4$"gekommen ist" <hk>
950 printc4$;:d$="-":gosub2200 <kb>
960 printb5$"neues spiel" <la>
970 printb$b4$"neuer spass (j/n)"l
eft$(qu$,4); <gc>
980 getkey a$:ifa$<>"j"thenscnclr:
end:elserun <im>
990 forrok=1to500:for t=800to1step
-15:sound 1,t,5:sound2,t+10,5:next <om>
1000 forymn=1to2000:next:graphic0:
scnclr:gosub 1660:color 4,1:poke 2
39,0 <pl>
1010 printb$b5$"D"f1$"n e w s"fo$"
D <hi>
1020 printb$b5$"DDDDDDDDD" <bd>
1030 printc4$wh$b2$"und wieder erl
itt ein mensch namens " <mo>
1040 jl=len(na$):jl=20-jl/2:char 1
,jl,10,na$:char 1,0,11,"" <ch>
1050 printc4$b$" den schrecklichen
" <ac>
1060 printc4$b$b5$"galgentod" <ij>
1070 printc4$b4$b4$"da er nicht au
f das wortX"b2$ <dm>
1080 printc4$b$b5$b2$rn$f1$f$rf$b3
$ <hd>
1090 printc4$b$b4$"gekommen ist" <jf>
1100 printc4$;:d$="-":gosub2200 <ih>
1110 printb5$"neues spiel" <ci>
1120 printb$b4$"neues glueck (j/n)
"left$(qu$,4); <og>
1130 getkey a$:ifa$<>"j"thenscnclr
:end:elserun <pe>
1140 scnclr <ni>
1150 end <if>
1160 data garten,ziel,oma,opa,mens
ch <be>
1170 data rasen,buch,holz,spiel,kl
o,geld <id>
1180 data hemd,hose,baum <hh>
1190 data gras,stange,acker,auto,s
tuhl,tisch,lampe,hund <if>
1200 data tiger,vogel,maus,haus,kr
ebs,kind,herz <dd>
1210 data radio,tasche,katze,tiger
,bock,bogen,pilz,post,dose,bild <in>
1220 datagewalt,abend,milch,bahn,b
ank,fisch,kabel,firma,korb <jf>
1230 data blumen,buch,hase,pauke <hl>
1240 data pokal,symbol,krug,vater,
onkel <kd>
1250 data lied,kuh,schau,faust,tag
,nacht,karton,frau,set,video <jh>
1260 data osten,mai,musik,tage,pos
t,fach,bart,kaiser,bord,woche,stun
de,mauer <bn>
1270 data preis,film,wald,pilz,blu
me,tiger,herd,gepard,ast,motiv,fic
hte,berg,knospe <ko>
1280 data kino,wand,band,loch,knop
f,lupe,schnur,flug,april,juni,freu
nd <md>
1290 poke 65298,129 and251 <ee>
1300 poke 65299,56and3or56:return <mf>
1310 restore1340:fory=0to17:readа:
poke 819+y,a:next:sys 819:return <ck>
1320 restore1390:forpo=14856to1506
3:readа:pokepo,a:next:return <cl>
1330 restore1350:forkl=14784to1484
8:readu:poke kl,u:next:return <pe>
1340 data162,0,189,0,208,157,0,56,
189,0,209,157,0,57,202,208,241,96 <mj>
1350 data1,3,3,7,15,15,7,15,128,19
2,192,224,240,240,224,240,31,7,15,
31,63,7,15,31 <dd>
1360 data248,224,240,248,252,224,2
40,248 <db>
1370 data63,127,7,15,31,63,127,255
,252,254,224,240,248,252,254,255 <ki>
1380 data3,3,3,3,3,3,3,3,192,192,1
92,192,192,192,192,192,192,192 <ap>
1390 data0,0,0,7,63,255,255,255 <fl>
1400 data0,0,255,255,255,255,255,2
55 <an>
1410 data0,255,255,255,255,255,255
,255 <lm>
1420 data255,255,255,255,255,255,2
55,255 <ll>
```

```
1430 data0,0,0,224,252,255,255,255 <ng>
1440 data56,28,56,28,56,28,56,28   <fd>
1450 data60,66,165,129,153,102,66,
60                                 <nd>
1460 data60,66,153,129,153,129,153
,255                               <dh>
1470 data102,102,102,102,102,231,2
31,0                               <fe>
1480 data7,15,30,60,120,240,224,19
2                                  <cp>
1490 data128,0,0,0,0,0,0,0         <ca>
1500 data0,0,0,0,0,0,1,3           <ke>
1510 data224,240,120,60,30,15,7,3  <aa>
1520 data0,0,0,0,0,0,128,192       <gc>
1530 data0,0,7,31,63,127,123,127   <fe>
1540 data0,0,240,248,124,222,118,2
54                                 <od>
1550 data219,255,255,118,127,125,6
3,15                               <fm>
1560 data255,183,255,254,238,254,1
88,240                             <hl>
1570 data7,7,7,7,7,7,7,7           <ji>
1580 data224,224,224,224,224,224,2
24,224                             <im>
1590 data7,7,7,7,15,15,31,255      <dl>
1600 data224,224,224,224,240,240,2
48,255                             <nm>
1610 data 255,255,0,0,0,0,0,0      <jb>
1620 data0,24,24,0,0,24,24,0       <cd>
1630 data 255,255,0,0,0,0,255,255  <pi>
1640 data 0,0,0,0,0,0,255,255      <nn>
1650 return                        <ho>
1660 color0,1:scnclr:printbl$"DDDD
"rn$"BBB"rf$;                      <li>
1670 d$="D":g=25:gosub2210:printrn
$"BBB"rf$"DDDDD"                   <hg>
1680 printbl$"DDDD"rn$"D"rf$"D"rn$
"F"rf$"DDDD"rn$"GGGGGGGGGGGGGGGGG"
;                                  <ob>
1690 printrf$"DDDD"rn$"F"rf$"D"rn$
"D"rf$"DDDDD"                      <pf>
1700 print"DDDD"rn$"D"rf$"DDDDDD"r
n$"G"rf$"-g a l l o w s-"rn$"G";   <fg>
1710 printrf$"DDDDDD"rn$"D"rf$"DDD
DD"                                <md>
1720 print"DD"rn$"ABDBE"rf$"DDDD"r
n$"GGGGGGGGGGGGGGGGG"rf$"DDDD"rn$"
ABDBE"rf$"DDD"                     <cl>
1730 d$="D":gosub2200              <da>
1740 return                        <dd>
1750 printgr$c2$b$b4$b4$"89"b3$b3$
"89"b5$"89"b2$"89 OP"b5$;          <ha>
1760 print"OP"b2$"OP89 OP:;89OP"b2
$":;OP 89:;"b2$":;"                <pl>
1770 print"QR"b3$"89QR"b2$"QR:; QR
<=:;QR89<=QR :;<=OP<="             <hc>
1780 printbr$"ST"gr$" OP:;"br$"ST"
gr$"89"br$"ST"gr$"<= "br$"ST>?"gr$
"<="br$"ST"gr$":;";                <gn>
1790 printbr$">?ST "gr$"<="br$">?"
gr$"QR"br$">?"gr$                  <oh>
1800 print"89 QR<=OP:;89"br$">? UV
"gr$"OP"br$">?UV"gr$"<= "br$" UV >
?"b2$"ST"gr$"OP"                   <co>
1810 print":; "br$"ST>?"gr$"QR<=:;
OP89 QR"b4$br$">?"gr$" OP 89 OP"br
$"UV"gr$"QR"                       <lo>
1820 print"<="br$" UV"gr$"OP"br$"S
T>?"gr$"<=QR:; ";                  <ok>
1830 printbr$"ST"b3$b3$gr$" QR :;
QR"b2$br$"ST"gr$                   <mg>
1840 printbr$">?"gr$"89 "gr$"QR"br
$"UV"b2$">?ST"gr$"<= ";            <id>
1850 printbr$"UV"b4$b3$"ST "gr$"<=
 "br$"ST"gr$"89"br$"UV"gr$         <ca>
1860 printb2$gr$":; "br$"ST"b3$b3$
"UV>?"b$"UV >? UV"gr$":;"          <lk>
1870 print" "gr$" <= "br$"UV"gr$b$
b$b4$b4$"<="                       <dd>
1880 printbr$b2$">?"b$b$b$" >?"    <bh>
1890 print"                        <pi>
1900 return                        <he>
1910 color1,2:char 1,0,22,"":d$="W
":gosub2200                        <kl>
1920 char 1,19,23,"F":char 1,0,24,
""                                 <pa>
1930 print"ZZZZZZZZZZZZZZZZZZZFZZZ
ZZZZZZZZZZZZZZZZZ"c2$wh$:char1,39,2
4,"":color1,2:                     <ei>
1940 poke 4071,90:return           <bn>
1950 ui=ui+1:for ta=15to0step-1:so
und1,ta,1:next                     <pg>
1960 ifhj =10then1990              <ll>
1970 ifhj =5thenonuigosub 2000,201
0,2020,2030,2040                   <gb>
1980 ifhj =3thenonuigosub 2050,206
0,2070                             <lf>
1990 onuigosub2080,2090,2100,2110,
2120,2130,2140,2170,2180,2190      <ib>
2000 gosub2080:goto660             <dk>
2010 gosub 2090:gosub 2100:gosub 2
110:goto660                        <cf>
2020 gosub 2120:gosub 2130:goto 66
0                                  <do>
2030 gosub 2140:gosub 2150:gosub 2
160:goto 660                       <bf>
2040 gosub 2170:gosub 2180:gosub 2
190                                <ap>
2050 gosub 2080:gosub 2090:gosub 2
100:gosub 2110:goto 660            <lf>
2060 gosub 2120:gosub 2130:goto 66
0                                  <po>
2070 gosub 2140:gosub 2150:gosub 2
160:gosub 2170:gosub 2180:gosub 21
90                                 <lf>
2080 char1,0,21,gr$+b$+b4$+"ABCDDD
```

```
DDDDCBE":return                          <jp>
2090 foryt=0to8 :char 1,19,20-yt,b
r$+"D":next:return                       <kj>
2100 char1,20,19,"N":char1,20,20,b
r$+"MN":return                           <gk>
2110 char1,18,19,"L":char1,17,20,b
r$+"LJ":return                           <bp>
2120 char1,19,11,br$+"DDDDDDD":ret
urn                                      <og>
2130 char1,20,12,br$+"LJ":char1,20
,13,br$+"JK":return                      <bk>
2140 color1,2,4:char1,24,12," F          <kc>
2150 color1,2,4:char1,24,13," F          <kc>
2160 color1,2,4:char1,24,14," F":r
eturn                                    <jp>
2170 char1,24,15,lr$+" G":return         <jf>
2180 char1,24,16,re$+" H":return         <ia>
2190 char1,24,17,br$+" I":goto 990       <oe>
2200 forx=1to39:printd$;:next:prin
td$:return                               <gb>
2210 forx=1tog:printd$;:next:retur
n                                        <fo>
2220 b$=chr$(32):b2$=b$+b$               <oc>
2230 b3$=b2$+b$:b4$=b3$+b$               <hf>
2240 b5$=b4$+b$:b$=b5$+b5$               <ia>
2250 rem nachspann ===============       <ke>
2260 rem * farbcodes/steuercodes *       <go>
2270 wh$=chr$(005):c4$=chr$(017)         <fd>
2280 rn$=chr$(018):re$=chr$(028)         <ne>
2290 c3$=chr$(029):gr$=chr$(030)         <hn>
2300 bl$=chr$(031):fl$=chr$(130)         <dk>
2310 fo$=chr$(132):c2$=chr$(145)         <jl>
2320 rf$=chr$(146):cl$=chr$(147)         <af>
2330 br$=chr$(149):lr$=chr$(150)         <bj>
2340 lg$=chr$(153):lb$=chr$(154)         <gg>
2350 pu$=chr$(156):ye$=chr$(158)         <nb>
2360 cy$=chr$(159)                       <na>
2370 rem ********* zeichenfolgen *       <bm>
2380 for q=1 to 40                       <ma>
2390 qd$=qd$+c4$:qu$=qu$+c2$             <jo>
2400 next q                              <pj>
2410 q=0:return                          <mc>
2420 rem ========================        <pj>
2430 rem 12277 bytes memory              <pk>
2440 rem 08344 bytes program             <jj>
2450 rem 00392 bytes variables           <ma>
2460 rem 00062 bytes arrays              <mb>
2470 rem 00507 bytes strings             <jo>
2480 rem 02048 bytes zeichensatz         <lc>
2490 rem 00924 bytes free (0)            <bm>
2500 rem ========================        <ko>
```

LISTING

AUF SCHATZSUCHE IM LABYRINTH

Räuber

Suchen Sie den Schatz, der in diesem 3-D-Labyrinth verborgen ist. In der linken oberen Ecke wird die Entfernung angezeigt. Die Zeit läuft mit. Wer ist der Schnellste?

Sie stehen in einem Labyrinth. Wohin Sie auch schauen, überall nur Gänge und Wände. Doch halt, etwas gibt Ihnen Aufschluß darüber, wo Sie sich ungefähr befinden. In der oberen Ecke sehen Sie eine Zahl. Das ist die Entfernung zum verborgenen Schatz. Ihn zu finden, ist nur eine Frage der Zeit, denn keine Monster oder sonstige Feinde bedrohen Sie. Sie sind mutterseelenallein.

Oft glauben Sie, gleich müßten Sie den Schatz erreicht haben. Doch dies ist ein Trugschluß, denn urplötzlich ist der Weg zu Ende und eine undurchdringliche Wand behindert Ihr Fortkommen. Es gibt dann nur eines: wieder zurückzugehen und in einem der anderen zahlreichen Gänge Ihr Glück zu versuchen.

Haben Sie endlich den Schatz erreicht, wird Ihnen mitgeteilt, wie lange Sie dazu gebraucht haben. Am Anfang dürften Sie es schwer haben. Später, mit etwas Übung, wird es schneller gehen. Sie brauchen jedoch nicht zu glauben, daß Sie dann das Labyrinth schon auswendig kennen, denn jedesmal wird es neu aufgebaut.

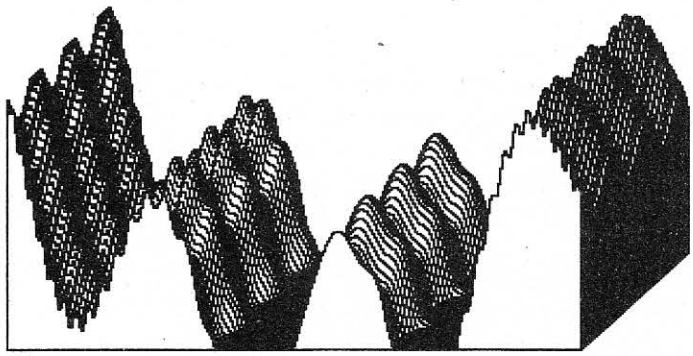
Nach erfolgreicher Schatzsuche dürfen Sie das Labyrinth von oben betrachten, um einen Überblick zu bekommen. Sichtbar sind der Lageplan, Ihr Startpunkt, der Schatz und der zurückgelegte Weg. Wenn Sie wollen, dürfen Sie es jetzt von Neuem versuchen. □

```
10 rem raeuber=================c16 <kn>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by eckhard schulz      <no>
50 rem                            <pd>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/plus4              <ki>
90 rem ========================== <jg>
100 gosub2860                     <gc>
110 key8,"?":color0,1:color4,1:pri
ntwh$                             <pp>
120 printchr$(142)chr$(8):gosub274
0                                 <di>
130 h=12:v=7:h1=h+1:w=818:v1=1630  <pn>
140 i=rnd(-ti)                    <nj>
150 dimxl(4),n2(4),yl(4),xr(4)    <fo>
160 fori=0to4:readxl(i),n2(i),yl(i
),xr(i):next                      <ji>
170 data8,20,-4,31,8,12,1,27,12,6,
5,24,15,2,8,22,17,0,10,21         <ji>
180 cx=int(rnd(1)*h)+1:cy=int(rnd(
1)*v)+1                           <ci>
190 c=cx:r=cy:s=0                 <mf>
200 fori=0to(v+1)*h1:pokew+i,0:pok
ev1+i,0:next                      <gh>
210 printcl$c4$c4$"aufbau des laby
rinthes"                          <hf>
220 da=0:n=0:av=v1+c+r*h1:aw=w+c+r
*h1:pokeaw,1:s=s+1:ifs>=h*vthen480 <jo>
230 printmid$(rf$,(sand1)+1,1)"V"; <om>
240 ifc>1andpeek(aw-1)-0thenda=da+
2:n=n+1                           <hh>
250 ifc<handpeek(aw+1)=0thenda=da+
1:n=n+1                           <ei>
260 ifr>1andpeek(aw-h1)=0thenda=da
+8:n=n+1                          <pd>
270 if(r<vandpeek(aw+h1)=0)thenda=
da+4:n=n+1                        <nj>
280 n=int(rnd(1)*n)+1             <mi>
290 onda+1goto300,470,460,330,450,
340,350,360,440,370,380,390,400,41
0,420,430                         <fm>
300 s=s-1                         <dm>
310 c=c+1:ifc>hthenc=1:r=r+1:ifr>v
thenr=1                           <bn>
320 onpeek(w+c+r*h1)+1goto310,220  <mo>
330 onngoto460,470                <ai>
340 onngoto450,470                <ig>
350 onngoto450,460                <im>
360 onngoto450,460,470            <bf>
370 onngoto440,470                <np>
380 onngoto440,460                <po>
390 onngoto440,460,470            <gk>
400 onngoto440,450                <ip>
410 onngoto440,450,470            <ej>
420 onngoto440,450,460            <go>
430 onngoto440,450,460,470        <mh>
440 r=r-1:av=av-h1:pokeav,peek(av)
or1:goto220                       <fg>
450 pokeav,peek(av)or1:r=r+1:av=av
+h1:goto220                       <fd>
460 c=c-1:av=av-1:pokeav,peek(av)o
r2:goto220                        <ab>
470 pokeav,peek(av)or2:c=c+1:av=av
+1:goto220                        <eg>
480 i=rnd(1)>.5:j=rnd(1):k=rnd(1)>
.5                                <od>
490 ifithenx=int(j*h)+1:y=v+(v-1)*
k                                 <oj>
500 ifi=0theny=int(j*v)+1:x=h+(h-1
)*k                               <mj>
510 if(cx-x)^2+(cy-y)^2<(h^2+v^2)/
9then480                          <ip>
520 sx=x:sy=y                     <ae>
530 ti$="000000"                  <ei>
540 :                             <hg>
550 printcl$                      <be>
560 fori=wtow+(v+1)*h1:pokei,0:nex
t                                 <fj>
570 d=int(rnd(1)*4)               <gn>
580 t4=ti:poke(w+x+y*h1),1        <fl>
590 iff=0thengosub950:printhe$;int
(sqr((x-cx)^2+(y-cy)^2)*10+.5)/10 <ok>
600 ifcx=xandcy=ythen2590         <pi>
610 getq$:ifq$=""goto610          <be>
620 ifq$="?"thenk=ti:goto2620     <pa>
630 jq=joy(2)                     <hi>
640 ifq$=c4$orjq=5thend=d+2       <nm>
650 ifq$=c1$orjq=7thend=d-1       <cf>
660 ifq$=c3$orjq=3thend=d+1       <nk>
670 ifd<0thend=d+4                <gc>
680 ifd>3thend=d-4                <pn>
690 ifq$=c2$orjq=1then700:else590 <eg>
700 av=v1+x+y*h1:aw=w+x+y*h1      <gb>
710 ondgoto740,760,780            <na>
720 ify>1and(peek(av-h1)and1)theny
=y-1:pokeaw-h1,1:goto590          <ek>
730 goto790                       <jb>
740 ifx<hand(peek(av)and2)thenx=x+
1:pokeaw+1,1:goto590              <ga>
750 goto790                       <mo>
760 ify<vand(peek(av)and1)theny=y+
1:pokeaw+h1,1:goto590             <li>
770 goto790                       <bl>
780 ifx>1and(peek(av-1)and2)thenx=
x-1:pokeaw-1,1:goto590            <jm>
790 dn=23:gosub2710               <om>
800 printtab(16)zz$ze$ze$ze$ze$lef
t$(ql$,14)c4$rn$"kein durchgang"c2
$:fori=1to1000:next               <gk>
810 printtab(16)c2$spc(14)left$(ql
$,14)c4$spc(14)c2$:goto600        <an>
820 printcl$" ";                  <ap>
830 fori=1toh:printze$ze$ze$;:next
:printc4$c4$                      <om>
```

```
840 forj=1tov:printz9$c2$c1$z9$c2$
c1$z9$c4$c4$;:fori=1toh                <ja>
850 k=peek(v1+i+j*h1)                  <dd>
860 printmid$(ze$+ze$+zp$+b2$+z9$+
ze$+ze$+ze$+b3$,1+3*k,3);              <gi>
870 ifk<2thenprintc2$c1$z9$c2$c1$z
9$c4$c4$;                              <ip>
880 printc2$c1$;                       <dk>
890 ifpeek(w+i+j*h1)=1thenprintc1$
+".";                                  <oh>
900 ifi=sxandj=sythenprintc1$+rn$+
" "+rf$;                               <lb>
910 ifi=cxandj=cythenprintc1$+z5$;     <ae>
920 ifi=xandj=ythenprintc1$;mid$("
>v<",d+1,1);                           <oj>
930 printc3$c4$;                       <bj>
940 next:printc4$c4$:next:return       <in>
950 s=0:printc1$                       <ap>
960 av=v1+x+y*h1                       <ca>
970 ondgoto1120,1260,1400              <al>
980 ifsy=y-sandsx=xthengosub2520       <bb>
990 ifcy=y-sandcx=xthengosub2440       <ec>
1000 k=peek(av-(s+1)*h1):ifk=0ork=
2goto1070                              <np>
1010 ifpeek(av-s*h1)and2thengosub1
550:goto1030                           <gf>
1020 gosub1670                         <mj>
1030 ifpeek(av-1-s*h1)and2thengosu
b2020:goto1050                         <oo>
1040 gosub2140                         <ei>
1050 s=s+1:ifs>4goto1540               <fk>
1060 goto980                           <fc>
1070 ifpeek(av-s*h1)and2thengosub1
750:goto1090                           <eo>
1080 gosub2220                         <dp>
1090 ifpeek(av-1-s*h1)and2thengosu
b1890:goto1110                         <kc>
1100 gosub2320                         <mf>
1110 return                            <ec>
1120 ifcy=yandcx=x+sthengosub2440      <ap>
1130 ifsy=yandsx=x+sthengosub2520      <lh>
1140 k=peek(av+s):ifk=0ork=1goto12
10                                     <he>
1150 ifpeek(av+s)=2thengosub1670:g
oto1170                                <bi>
1160 gosub1550                         <fp>
1170 ifpeek(av+s-h1)and1thengosub2
020:goto1190                           <pn>
1180 gosub2140                         <nl>
1190 s=s+1:ifs>4goto1540               <lb>
1200 goto1120                          <aj>
1210 ifpeek(av+s)=0thengosub2220:g
oto1230                                <fb>
1220 gosub1750                         <no>
1230 ifpeek(av+s-h1)and1thengosub1
890:goto1250                           <ln>
1240 gosub2320                         <fh>
1250 return                            <fl>
1260 ifcy=y+sandcx=xthengosub2440      <ah>
1270 ifsy=y+sandsx=xthengosub2520      <mn>
1280 k=peek(av+s*h1):ifk=0ork=2the
ngoto1350                              <ie>
1290 ifpeek(av-1+s*h1)and2thengosu
b1550:goto1310                         <nl>
1300 gosub1670                         <pm>
1310 ifpeek(av+s*h1)=1thengosub214
0:goto1330                             <oc>
1320 gosub2020                         <fn>
1330 s=s+1:ifs>4goto1540               <eh>
1340 goto1260                          <jp>
1350 ifpeek(av-1+s*h1)and2thengosu
b1750:goto1370                         <hl>
1360 gosub2220                         <ga>
1370 ifpeek(av+s*h1)=0thengosub232
0:goto1390                             <hi>
1380 gosub1890                         <po>
1390 return                            <he>
1400 ifcx=x-sandcy=ythengosub2440      <hb>
1410 ifsx=x-sandsy=ythengosub2520      <ck>
1420 k=peek(av-(s+1)):ifk=0ork=1th
engoto1490                             <jo>
1430 ifpeek(av-s-h1)and1thengosub1
550:goto1450                           <eb>
1440 gosub1670                         <im>
1450 ifpeek(av-s)and1thengosub2020
:goto1470                              <md>
1460 gosub2140                         <oo>
1470 s=s+1:ifs>4then1540               <ba>
1480 goto1400                          <ci>
1490 ifpeek(av-s-h1)and1thengosub1
750:goto1510                           <pa>
1500 gosub2220                         <pd>
1510 ifpeek(av-s)and1thengosub1890
:goto1530                              <af>
1520 gosub2320                         <ha>
1530 return                            <in>
1540 dn=11:gosub2710:printtab(19)"
MN"c4$c1$c1$"NM"he$:return             <mm>
1550 dn=yl(s):gosub2710                <cn>
1560 ifs=0thenprinttab(xr(s))ze$       <kl>
1570 ifs>0ands<4thenfori=1to4-s:pr
inttab(xr(s)+5-s)zj$:next              <ol>
1580 ifs>0thenprinttab(xr(s))mid$(
ze$+ze$+ze$+ze$+zj$,s)                 <kc>
1590 ifs=4thenprinttab(xr(s))zj$+z
j$+c4$+c1$+c1$+zj$+zj$                 <fh>
1600 q$="":ifs>0thenq$=mid$(c3$+c3
$+c3$+zj$,s)                           <mo>
1610 ifs<4thenfori=0ton2(s)+1:prin
ttab(xr(s))zj$q$:next                  <pk>
1620 ifs>0thenprinttab(xr(s))mid$(
zm$+zm$+zm$+zm$+zj$,s)                 <ha>
1630 ifs>0ands<4thenfori=1to4-s:pr
inttab(xr(s)+5-s)zj$:next              <fm>
1640 ifs=0thenprinttab(xr(s))zm$       <cg>
1650 printhe$                          <bl>
```

```
1660 return                                    <jd>
1670 dn=yl(s):gosub2710                        <el>
1680 ifs=0thenprinttab(xr(s))"N":g
oto1700                                        <be>
1690 fori=1to5-s:printtab(xr(s)+5-
s-i)"N":next                                   <ek>
1700 dn=n2(s)+2:gosub2720                      <ho>
1710 ifs=0thenprinttab(xr(s))"M":g
oto1730                                        <kc>
1720 fori=0to4-s:printtab(xr(s)+i)
"M":next                                       <jh>
1730 printhe$                                  <ca>
1740 return                                    <dd>
1750 j=3-s:ifs=0thenj=0                        <de>
1760 printhe$;                                 <di>
1770 ifs>0goto1810                             <en>
1780 printtab(20);:printc4$;:fori=
1to18:printzm$;:next:printzm$                  <lb>
1790 dn=n2(0):gosub2720                        <dd>
1800 printtab(20);:fori=1to19:prin
tze$;:next:goto1870                            <fm>
1810 dn=yl(s):gosub2720                        <on>
1820 fori=1to5-s:printtab(j+xr(s)+
2)zj$:next                                     <hp>
1830 printtab(20);:fori=19toj++xr(
s):printzm$;:next:printzj$                     <gk>
1840 ifs<4thenfori=1ton2(s):printt
ab(j+xr(s)+2)zj$:next                          <bo>
1850 printtab(20);:fori=19toj+xr(s
):printze$;:next:printzj$                      <gf>
1860 fori=1to5-s:printtab(j+xr(s)+
2)zj$:next                                     <fg>
1870 printhe$                                  <oj>
1880 return                                    <em>
1890 printhe$;                                 <ij>
1900 ifs>0goto1940                             <fk>
1910 printc4$;:fori=1to11+xl(0):pr
intzm$;:next:printzm$                          <ml>
1920 dn=n2(0):gosub2720                        <nh>
1930 fori=1to12+xl(0):printze$;:ne
xt:goto1870                                    <of>
1940 dn=yl(s):gosub2720                        <im>
1950 fori=1to5-s:printtab(xl(s))z9
$:next                                         <ba>
1960 printtab(xl(s));:printz9$;:fo
ri=xl(s)to17:printzm$;:next:printz
m$                                             <lf>
1970 ifs<4thenfori=1ton2(s):printt
ab(xl(s))z9$:next                              <ca>
1980 printtab(xl(s));:printz9$;:fo
ri=xl(s)to17:printze$;:next:printz
e$                                             <li>
1990 fori=1to5-s:printtab(xl(s))z9
$:next                                         <nb>
2000 printhe$                                  <bb>
2010 return                                    <fb>
2020 dn=yl(s):gosub2710                        <pp>
2030 ifs=0thenprinttab(xl(s))ze$               <lf>
2040 ifs>0ands<4thenfori=1to4-s:pr
inttab(xl(s))z9$:next                          <kd>
2050 ifs>0thenprinttab(xl(s))left$
(z9$+ze$+ze$+ze$+ze$,6-s)                      <ld>
2060 ifs=4thenprinttab(xl(s))z9$+z
9$+c1$+c1$+c4$+z9$+z9$                         <pp>
2070 q$="":ifs>0thenq$=mid$(c3$+c3
$+c3$+z9$,s)                                   <bf>
2080 ifs<4thenfori=0ton2(s)+1:prin
ttab(xl(s))z9$q$:next                          <fl>
2090 ifs>0thenprinttab(xl(s))left$
(z9$+zm$+zm$+zm$+zm$,6-s)                      <pk>
2100 ifs>0ands<4thenfori=1to4-s:pr
inttab(xl(s))z9$:next                          <mg>
2110 ifs=0thenprinttab(xl(s))zm$               <be>
2120 printhe$                                  <jj>
2130 return                                    <ec>
2140 dn=yl(s):gosub2710                        <oj>
2150 ifs=0thenprinttab(xl(s))"M":g
oto2170                                        <ak>
2160 fori=1to5-s:printtab(xl(s)+i)
"M":next                                       <am>
2170 dn=n2(s)+2:gosub2720                      <lh>
2180 ifs=0thenprinttab(xl(s))"N":g
oto2200                                        <mn>
2190 fori=0to4-s:printtab(xl(s)+5-
s-i)"N":next                                   <je>
2200 printhe$                                  <ke>
2210 return                                    <od>
2220 gosub1670                                 <gg>
2230 dn=yl(s)+5-s:gosub2710                    <dm>
2240 printtab(20);                             <ff>
2250 ifs<4thenfori=1toxr(s)-21:pri
ntzm$;:next                                    <ma>
2260 print"P"c1$c4$;                           <on>
2270 ifs<4thenfori=1ton2(s):printz
9$c4$c1$;:next                                 <hj>
2280 printzp$c1$c1$;                           <fc>
2290 ifs<4thenfori=1toxr(s)-20-1:p
rintze$c1$c1$;:next                            <an>
2300 printhe$                                  <nm>
2310 return                                    <km>
2320 ifs=0thenj=5:goto2340                     <kp>
2330 j=0                                       <nh>
2340 gosub2140                                 <gf>
2350 dn=yl(s)+5-s:gosub2710                    <nl>
2360 printtab(19);                             <jf>
2370 ifs<4thenfori=1to18-xl(s)-5+s
+j:printzm$c1$c1$;:next                        <pm>
2380 print"O"c1$c4$;                           <ji>
2390 ifs<4thenfori=1ton2(s):printz
j$c4$c1$;:next                                 <cp>
2400 print"L";                                 <hg>
2410 ifs<4thenfori=1to18-xl(s)-5+s
+j:printze$;:next                              <ca>
2420 printhe$                                  <gk>
2430 return                                    <jn>
2440 ifs=0ors=4thenreturn                      <pc>
```

```
2450 dn=18:gosub2710                                    <pb>
2460 ifs=3thenprinttab(20)c2$+c2$+
c2$+"."                                                 <ec>
2470 ifs=2thenprinttab(19)rn$+c2$+
z8$+rf$+z8$+c1$+c1$+c4$+z5$+"N"+c2
$+zj$                                                   <ln>
2480 ifs<>1thenreturn                                   <fk>
2490 printtab(18)c4$" "ze$ze$ze$le
ft$(q1$,4)c4$"N"b2$"N";                                 <dm>
2500 printzj$left$(q1$,5)c4$z5$z5$
z5$" "zj$left$(q1$,5)c4$z5$z5$z5$"
"                                                       <gd>
2510 return                                             <dn>
2520 ifs=0ors=4thenreturn                               <dl>
2530 dn=20:gosub2710                                    <pb>
2540 ifs=3thenprinttab(19)left$(qu
$,5)+rn$+z1$+z1$+c2$+c1$+c1$+rf$+z
1$+z1$                                                  <ek>
2550 ifs=2thenprinttab(16)c2$+c2$+
c2$+rn$+z8$b3$b3$yq$                                    <lm>
2560 ifs=1thenprinttab(13)rn$+z8$b
$b2$yq$                                                 <ig>
2570 ifs=1thenprinttab(12)rn$+z8$b
$b4$yq$                                                 <mh>
2580 return                                             <mk>
2590 printhe$b$"deine zeit: "left$
(ti$,2)":"mid$(ti$,3,2)"'"right$(t
i$,2)b5$b4$                                             <ek>
2600 k=ti:fori=1to25:poke2035,23:p
oke2036,15:sys65520:print"gefunden
":forj=1to50:nextj                                      <kb>
2610 poke2035,23:poke2036,15:sys65
520:printb$:forj=1to50:nextj,i                          <el>
2620 gosub820                                           <dm>
2630 printtab(10)c2$"noch ein spie
l";                                                     <ml>
2640 sw=1:tm=0                                          <da>
2650 gett$:ift$<>""then2680                             <oa>
2660 ifti>tmthenprintmid$("? ",sw,
1)c1$;:tm=ti+15:sw=3-sw                                 <oo>
2670 goto2650                                           <jc>
2680 ift$="j"thenprint" ja.":goto1
80                                                      <el>
2690 ift$="n"thenprint" nein"+c4$:
end                                                     <gi>
2700 goto2650                                           <go>
2710 printhe$;                                          <ie>
2720 ifdn>0thenforz=1todn:printc4$
;:next                                                  <pk>
2730 return                                             <ph>
2740 printc1$c4$spc(13)"schatzraeu
ber"                                                    <dl>
2750 printc4$spc(7)"(c) 1987 by ec
khard schulz"                                           <cc>
2760 printc4$c4$c4$" finde den sch
atz der in dem labyrinth                                <nl>
2770 printc4$" verborgen ist.                           <ml>
2780 printc4$" cursor up"b4$"- 1 s
chritt vorwaerts"                                       <em>
2790 print" cursor left"b2$"- dreh
ung nach links"                                         <ob>
2800 print" cursor right = drehung
 nach rechts                                            <hf>
2810 print" cursor down"b2$"= dreh
ung um 180 grad"                                        <hb>
2820 print" help"spc(9)"= hilfe"                        <je>
2830 printc4$" analog dazu steueru
ng mit dem joystick"b2$c4$"in port
 ii."                                                   <nf>
2840 printc4$c4$spc(7)"druecke 'ta
ste' zum start"                                         <ji>
2850 getkeya$:return                                    <kc>
2860 b$=chr$(32):b2$=b$+b$                              <ja>
2870 b3$=b2$+b$:b4$=b3$+b$                              <kf>
2880 b5$=b4$+b$:b$=b5$+b5$                              <io>
2890 rem nachspann ==============                       <kh>
2900 rem * farbcodes/steuercodes *                      <ad>
2910 wh$=chr$(005):c4$=chr$(017)                        <gd>
2920 rn$=chr$(018):he$=chr$(019)                        <fm>
2930 c3$=chr$(029):c2$=chr$(145)                        <ma>
2940 rf$=chr$(146):c1$=chr$(147)                        <he>
2950 c1$=chr$(157)                                      <id>
2960 rem *** zeichensatz/graphik *                      <gb>
2970 z1$=chr$(162):z5$=chr$(166)                        <km>
2980 z8$=chr$(169):z9$=chr$(170)                        <mm>
2990 ze$=chr$(175):zj$=chr$(180)                        <fb>
3000 zm$=chr$(183):zp$=chr$(186)                        <me>
3010 yq$=chr$(223)                                      <do>
3020 rem ********* zeichenfolgen *                      <ca>
3030 forq=1to10:zz$=zz$+ze$:next                        <kn>
3040 for q=1 to 40                                      <pp>
3050 qu$=qu$+c2$:q1$=q1$+c1$                            <ib>
3060 next q                                             <fe>
3070 return                                             <ka>
3080 rem ------------------------                       <op>
3090 rem 12277 bytes memory                             <ae>
3100 rem 07955 bytes program                            <pe>
3110 rem 00399 bytes variables                          <dj>
3120 rem 00128 bytes arrays                             <if>
3130 rem 00505 bytes strings                            <ih>
3140 rem 03290 bytes free (0)                           <eh>
3150 rem ------------------------                       <lh>
```



Motor-Race

AUCH GERINGER PROGRAMMIERAUFWAND SCHAFFT SPANNENDE SPIELE

Ein Parcours und eine kleine Steuerroutine genügen, um ein Objekt über die Piste sausen zu lassen. Eine DO...LOOP-Schleife bringt Tempogewinn. INSTR ersetzt viele IF-Abfragen.

Es geht darum, möglichst viele Runden in möglichst kurzer Zeit zu schaffen. Die Steuerung des Fahrzeugs geschieht über Tastendruck, wie es im Programm angegeben ist. Vier Fahrzeuge stehen Ihnen zur Verfügung, damit ein Unfall Sie nicht gleich aus dem Rennen wirft. Die Fahrgeschwindigkeit ist wählbar: Eine kleinere Zahl bedeutet eine höhere Geschwindigkeit.
Unsere Redaktion hat die ursprüngliche Programmierung etwas abgewandelt, da die Tastensteuerung ungünstig plaziert war. Jetzt können Sie die Finger in Bereitschaftsstellung über den Steuertasten halten.

Steuerung:

z = oben
x = unten
, = links
. = rechts

Um Ihnen eine einfache Umbelegung zu gestatten, haben wir diese Werte in der DATA-Zeile Nummer 110 abgelegt, wo Sie sie Ihren Bedürfnissen gemäß abändern können.
Viele Leerzeichen hintereinander machen das Abtippen zum Ratespiel. Die SPC-Funktion schafft Abhilfe. Denselben Zweck wie acht IF-Abfragen erreichten auch eine einzige IF-Abfrage und eine INSTR-Anweisung, verbunden mit einer indizierten Variablen, in den Zeilen 1270 und 1280. Die SIGN-Funktion verhalft zur richtigen Fahrzeug-Darstellung auch bei noch nicht gestartetem Fahrzeug. DO in Zeile 1240 und LOOP in Zeile 1280 brachten erheblichen Tempogewinn gegenüber einem ursprünglichen GOTO.
Da eine DO-Schleife nicht einfach mit einem GOTO verlassen werden darf, schlossen wir diese in den Zeilen 1300 und 1420 durch einen Trick richtig ab. LOOP UNTIL 1=1 hat keinen Einfluß auf den Programmablauf. Es findet also kein Sprung statt. Nötig ist diese Anweisung allerdings, damit die durch die DO-Anweisung im BASIC-Pseudo-Stack abgelegten Werte wieder abgehoben werden. Es soll ja nicht zu einem Programmabbruch wegen Stack-Überlauf kommen.
Genug der Erläuterungen, das Spiel kann beginnen. □

```
10 rem motor-race==========c16 <pp>
20 rem (p) commodore welt team <ho>
30 rem ========================== <ng>
40 rem (c) by michael poser <bp>
50 rem <pd>
60 rem <ah>
70 rem basic v3.5 <nl>
80 rem c16/116,plus4 <ea>
90 rem ========================== <jg>
100 goto360 <pd>
110 data z,x,.,",",-40,40,1,-1 <ed>
120 fori=1to4:readx$:in$=in$+x$:ne
xt <jp>
130 fori=1to4:readf(i):next:goto14
40 <bk>
140 printcl$spc(9)"PSV michael"spc
(2)"PSV" <on>
150 printspc(9)"QTW poser"spc(4)"Q
TW" <le>
160 printspc(9)"RUX presents RUX" <mp>
170 printspc(5)"Z"yo$yq$spc(18)"Z"
yo$yq$ <gg>
180 printspc(5)ym$zz$s2$spc(2)"**m
otor-race**"spc(2)ym$zz$s2$ <fn>
190 printspc(5)yn$spc(4)"---------
-----"spc(2)yn$spc(2) <da>
200 print <ck>
210 printspc(2)"versuchen sie moeg
lichst viele" <af>
220 printspc(2)"runden zu fahren a
ber machen sie" <ki>
230 printspc(2)"keine unfaelle den
n sie haben" <id>
240 printspc(2)"nur vier wagen zur
 verfuegung" <bc>
250 printspc(2)"sie lenken ihren w
agen mit :" <ak>
260 print <kd>
270 printspc(7)left$(in$,1)spc(5)"
= oben" <bb>
280 printspc(7)mid$(in$,2,1)spc(5)
"= links" <km>
290 printspc(7)mid$(in$,3,1)spc(5)
"= rechts" <pb>
300 printspc(7)right$(in$,1)spc(5)
"= unten " <jo>
310 print <ah>
320 printspc(13)rn$"taste"rf$ <ke>
330 r$="" <nm>
340 get r$:if r$=""then 340 <ca>
350 goto 810 <kn>
360 : <am>
370 vol 4 <lh>
380 color0,2:color4,2:vol8:color1,
6,2 <kl>
390 v=65280 <pf>
400 pokev+18,peek(v+18)and251 <bm>
410 pokev+19,peek(v+19)and3or48 <pb>
420 poke740,56:clr:gosub110 <fk>
430 fort=832to849:reada:poket,a:ne
```

```
xt:sys832                                  <ln>
440 fort=12800to13071:reada:poket,
a:next                                     <db>
450 data162,0,189,0,208,157,0,48,1
89,0,209,157,0,49,202,208,241,96           <gl>
460 data  0,48,96,255,255,96,48,0          <oi>
470 data  90,90,24,102,126,24,90,9
0                                          <om>
480 data  90,90,24,126,102,24,90,9
0                                          <fp>
490 data  0,219,24,247,247,24,219,
0                                          <kh>
500 data  0,219,24,239,239,24,219,
0                                          <hc>
510 data  24,24,24,24,24,24,24,24          <fm>
520 data  0,0,0,255,255,0,0,0              <nl>
530 data  0,0,0,248,248,24,24,24           <go>
540 data  0,0,0,31,31,24,24,24             <ca>
550 data  24,24,24,31,31,0,0,0             <kd>
560 data  24,24,24,248,248,0,0,0           <op>
570 data  129,66,0,24,24,0,66,129          <an>
580 data  129,66,0, 0, 0,0,66,129          <ag>
590 data  129, 0,0, 0, 0,0, 0,129          <jb>
600 data  24,24,24,255,255,0,0,0           <lp>
610 data  24,24,24,248,248,24,24,2
4                                          <jd>
620 data  3,4,8,8,8,40,113,81              <hm>
630 data  209,209,209,208,208,80,8
0,113                                      <pj>
640 data  50,16,16,8,12,4,3,0              <hm>
650 data  255,0,0,0,3,212,24,27            <lp>
660 data  123,127,240,34,34,34,188
,1                                         <dn>
670 data  126,126,124,24,0,0,255,2
55                                         <kc>
680 data  192,48,16,16,16,133,71,6
9                                          <fe>
690 data  133,133,5,5,9,138,74,14          <dj>
700 data  8,8,8,16,48,64,192,0             <fj>
710 data  126,66,100,24,0,0,255,25
5                                          <hg>
720 data  0,0,1,1,0,0,1,1                  <kc>
730 data  3,3,7,7,14,14,31,24              <nk>
740 data  48,48,96,96,192,192,0,0          <mi>
750 data  0,192,32,32,192,192,255,
179                                        <ai>
760 data  204,204,51,51,204,204,24
3,31                                       <fc>
770 data  0,0,0,0,0,0,128,254              <el>
780 data  204,204,56,48,224,208,48
,224                                       <fo>
790 data  0,0,0,0,0,0,0,0                  <ge>
800 goto 140                               <nl>
810 r=0                                    <eg>
820 goto 900                               <db>
830 f$="":printcl$"sie haben "r" r
unden in "val(ti$)-1"sek."                 <pi>
840 print"bei schwierigkeitsgrad "
s"geschafft":clr:gosub110                  <mf>
850 print:printspc(10)"weiter j/n"         <oa>
860 getf$:iff$=""then860                   <dp>
870 if f$="n"thenend                       <dh>
880 if f$="j"then900                       <od>
890 goto860                                <jj>
900 q=1                                    <bf>
910 printcl$"bitte geben sie den s
chwierig-"                                 <dh>
920 print"keitsgrad ein (1-100) (l
angsam-schnell)"                           <kp>
930 input s:ti$="000000"                   <fn>
940 printcl$" runden:"spc(6)"schwi
erigkeitsgrad:":n=0                        <cc>
950 printhe$spc(10)r                       <ao>
960 printhe$spc(33)s                       <ia>
970 print                                  <de>
980 print"HFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFG"                          <pm>
990 print"E"spc(38)"E"                    <bj>
1000 print"E"spc(38)"E"                   <kl>
1010 print"E"spc(38)"E"                   <pe>
1020 print"E"spc(3)"HFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFG"spc(3)"E"              <jh>
1030 print"E"spc(3)"IFFFFFFFFFG"sp
c(20)"E"spc(3)"E"                          <hj>
1040 print"E"spc(13)"E"spc(20)"E"s
pc(3)"E"                                   <jf>
1050 print"E"spc(13)"EHFFFFFFFFFFF
FFFFFFFFJ"spc(3)"E"                        <gn>
1060 print"E"spc(13)"EE"spc(23)"E"         <fh>
1070 print"IFFFFFFFFFG"spc(3)"EE H
FFFFFG"spc(15)"E"                          <of>
1080 print"HFFFFFFFFFJ"spc(3)"EE E
 Z"yo$yq$" E"spc(15)"E"                    <kg>
1090 print"E"spc(13)"EE E "ym$zz$s
2$" E"spc(3)"HFFFFFFFFFFFJ"                <na>
1100 print"E"spc(13)"EE E "yn$spc(
3)"E"spc(3)"IFFFFFFFFFFFG"                 <do>
1110 print"E"spc(13)"EE E Z"yo$yq$
" E"spc(15)"E"                             <in>
1120 print"E"spc(3)"HFFFFFFFFFNO E
 "ym$zz$s2$" E"spc(15)"E"                  <km>
1130 print"E"spc(3)"EPSVwrittenE E
 "yn$spc(3)"E"spc(15)"E"                   <fk>
1140 print"E"spc(3)"EQTW"spc(2)"by
"spc(3)"E IFFFFFJ"spc(3)"HFFFFFFFG
"spc(3)"E"                                 <po>
1150 print"E"spc(3)"ERUXm.poserE"s
pc(11)"E"spc(7)"E"spc(3)"E"                <lc>
1160 print"E"spc(3)"IFFFFFFFFFFNFF
FFFFFFFFFOHFFFFFFJ"spc(3)"E"               <ha>
1170 print"E"spc(26)"IJ"spc(10)"E"         <hg>
1180 print"E"spc(3)"HFFFFFFFFFFFG"
spc(22)"E"                                 <nd>
1190 print"E"spc(3)"E"spc(11)"E"sp
c(22)"E"                                   <ga>
1200 print"IFFFJ"spc(11)"IFFFFFFFF
```

```
FFFFFFFFFFFFFF"he$:poke4071,74      <oi>
1210 printhe$left$(qd$,8)spc(16)"u
nfaelle:"spc(6)q-1                  <cm>
1220 printhe$left$(qd$,7)spc(16)"s
ekunden: ";int(ti/60)               <dj>
1230 i=3954:poke i,65:n=0:poke3918
,64                                 <cn>
1240 do:a$="":poke3917-79,85:char
1,27,7,ti$:geta$                    <ef>
1250 foro=1to s:next:if a$=""theno
=0:poke3917-79,89                   <jo>
1260 ifn>0thensound3,200,1:ifi=391
8then1420:elsepokei,32              <mc>
1270 b=n:n=instr(in$,a$):ifn=0then
n=b                                 <eh>
1280 i=i+f(n):ifpeek(i)<69thenpoke
i,65+n-sgn(n):loop                  <en>
1290 rem unfall                     <bg>
1300 loopuntil1=1:sound 3,600,55    <po>
1310 poke i,75:p=0                  <oh>
1320 forp=1to250:next               <kf>
1330 poke i,76:p=0                  <jp>
1340 forp=1to250:next               <gd>
1350 poke i,77:p=0                  <lh>
1360 forp=1to250:next               <ae>
1370 if q<4 thenq=q+1:poke239,0:go
to 940                              <pb>
1380 ifq>3then830                   <ni>
1390 f$="":char1,0,0,cl$:char 1,8,
12,"betruegen gilt nicht":r=r-1     <bn>
1400 getf$:iff$=""then1400          <bh>
1410 if q<4 thenq=q+1:goto 940      <id>
1420 printhe$spc(10);:r=r+1:printr
:if peek(3918)=67thenloopuntil1=1:
goto1390                            <ib>
1430 poke 3918,64:goto1270          <ak>
1440 rem nachspann ================ <di>
1450 rem * farbcodes/steuercodes *  <jb>
1460 c4$=chr$(017):rn$=chr$(018)    <el>
1470 he$=chr$(019)                  <li>
1480 rf$=chr$(146):cl$=chr$(147)    <nb>
1490 rem *** zeichensatz/graphik *  <ni>
1500 s2$=chr$(160):ym$=chr$(219)    <am>
1510 yn$=chr$(220):yo$=chr$(221)    <al>
1520 yq$=chr$(223):zz$=chr$(255)    <ng>
1530 rem ********* zeichenfolgen *  <nb>
1540 for q=1 to 40                  <fp>
1550 qd$=qd$+c4$                    <nh>
1560 next q                         <mo>
1570 return                         <no>
1580 rem ========================== <jo>
1590 rem 12277 bytes memory         <lc>
1600 rem 04710 bytes program        <fa>
1610 rem 00168 bytes variables      <ig>
1620 rem 00062 bytes arrays         <cn>
1630 rem 00402 bytes strings        <jp>
1640 rem 06935 bytes free (0)       <gi>
1650 rem ========================== <bi>
```


TIC-TAC-TOE

Menschen gegen Computer

Daß ein Computer intelligent sei, ist eine Fehlannahme. Wenn sich Intelligenz mit Intelligenz mißt, dann höchstens die des Programmierers mit der des vor dem System Sitzenden. Zeigen Sie, wer hier der Gewinner ist.

Wer kennt es nicht, das kleine Quadrat mit den neun Feldern. Das Spiel heißt TIC TAC TOE.
Wer drei Symbole in eine Reihe bringt, hat gewonnen. Nach dem Start des Spiels mit RUN erscheint das Spielfeld und das Programm frägt, wer beginnt. 'C' heißt Computer, 'M' steht für Mensch. Vor dem Spiel wird der Spielstand eingegeben. Er kann während des Spiels verändert werden. Dies geschieht mit der Taste 'S'.
Es erscheint auf dem Bildschirm links die Bedenkzeit des Computers und rechts die des Spielers.
Das Symbol für den Computer ist ein Kreis auf dem Spielfeld. Der Mensch spielt mit einem Kreuz.
Die Zugeingabe erfolgt durch Angabe der Nummer des gewünschten Feldes.
Nachdem man dem Computer eine Lektion erteilt hat (oder selber eine erhielt), wird man gefragt, ob man nocheinmal spielen möchte. Ja heißt 'j'. □

```
10 rem tic-tac-toe ==========c16  <in>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <mm>
40 rem (c) by ekremhan tuncer     <ia>
50 rem                            <pd>
60 rem                            <ah>
70 rem basic v3.5                 <nl>
80 rem c16/116/plus4              <ki>
90 rem ========================== <km>
100 vol 8:gosub2120               <mc>
110 scnclr                        <me>
120 color 4,1:color 0,8           <co>
130 print chr$(8)                 <fk>
140 print lb$c2$"* "rn$"tic tac to
e"rf$" von "rn$"ekremhan tuncer"rf
$" 1988 *"bk$                     <nj>
150 dim fp(9,1,1),f(9),sk(24)     <bm>
160 b$=f1$:ct$="000000":mt$="00000
0"                                <kg>
170 mz$="MN"+c1$+c1$+c4$+"NM":cz$=
"UI"+c1$+c1$+c4$+"JK"             <he>
180 print c4$c4$c4$               <mm>
190 printspc(14)"UCC"zh$"CC"zh$"CC
I"                                <hg>
200 printspc(14)"B1 B2 B3 B"      <jp>
210 printspc(14)"B"spc(2)"B"spc(2)
"B"spc(2)"B"                      <ib>
220 printspc(14)za$"CC"ym$"CC"ym$"
CC"zi$                            <lh>
230 printspc(14)"B4 B5 B6 B"      <pf>
240 printspc(14)"B"spc(2)"B"spc(2)
"B"spc(2)"B"                      <bd>
250 printspc(14)za$"CC"ym$"CC"ym$"
CC"zi$                            <fk>
260 printspc(14)"B7 B8 B9 B"      <ka>
270 printspc(14)"B"spc(2)"B"spc(2)
"B"spc(2)"B"                      <dn>
280 printspc(14)"JCC"zg$"CC"zg$"CC
K"                                <jc>
290 gosub1420:gosub1720:gosub1610:
goto 1550                         <nd>
300 char 1,3,20,chr$(27)+"q"      <ai>
310 char 1,12,20,chr$(27)+"q"     <bo>
320 ifzu=>9thenchar1,15,12,c1$+c1$
+c1$+f1$+left$(qd$,5)+"unentschied
en":goto1260                      <cm>
330 print"ihr zug "f1$"?"fo$:ti$=m
t$:at$="c"                        <gm>
340 getmp$:gosub1820              <kd>
350 if mp$="s" then gosub 1720:got
o 310                             <op>
360 mz=val(mp$)                   <cp>
370 if mz>9 or mz<1 then 340      <ad>
380 if f(mz)<>0 then 340          <dk>
390 char 1,22,20,mp$              <jd>
400 f(mz)=10:ti$=ct$:at$="m":gosub
 750                              <fp>
410 ifzu=>9thenchar1,15,12,f1$+c1$
+c1$+c1$+left$(qd$,5)+"unentschied
en"+bk$:goto1260                  <oj>
420 gosub 1070:gosub 1820         <bl>
430 if s=1 then gosub 1200:goto 31
0                                 <po>
440 if s=2 then 460               <ap>
450 ifzu=1orzu=2thengosub1360:goto
310                               <kg>
460 restore                       <ch>
470 i=0                           <hh>
480 gosub 1820                    <ad>
490 do                            <mc>
500 i=i+1                         <hk>
510 read a,b,c                    <ok>
520 if f(a)+f(b)=20 and f(c)=0 the
n cz=c:gosub880:goto 310          <de>
530 loop until i=24               <op>
540 gosub 1820                    <jl>
550 ip=0:op=0                     <gh>
560 for ii=1 to 9                 <oj>
570 om=0:restore 2030             <gi>
580 if f(ii)<>0 then 690          <cf>
590 gosub 1820                    <nb>
600 for ia=1 to 8                 <fm>
610 f(ii)=1                       <jo>
620 read a,b,c                    <hg>
630 if f(a)+f(b)+f(c)=2 then om=om
+1                                <fk>
640 f(ii)=10:if f(a)+f(b)+f(c)=20
then op=op+1                      <ll>
650 if om=2 then cz=ii:f(ii)=1:gos
ub 810:goto 310                   <op>
660 if op=2 then ip=ii            <ld>
670 next ia                       <oc>
680 f(ii)=0                       <pc>
690 next ii                       <ij>
700 if op>=2 then cz=ip:f(cz)=1:go
sub 810:goto 310                  <cj>
710 gosub 1820                    <nm>
720 gosub 1200                    <af>
730 goto 310                      <hb>
740 rem zeichnen mensch           <fn>
750 x=fp(mz,1,0):y=fp(mz,1,1)     <ek>
760 char 1,x+13,y+5,mz$           <jl>
770 zu=zu+1                       <hd>
780 gosub 920                     <ng>
790 return                        <ma>
800 rem zeichnen computer         <oj>
810 x=fp(cz,1,0):y=fp(cz,1,1)     <ga>
820 char 1,x+13,y+5,cz$           <fb>
830 zu=zu+1                       <jf>
840 gosub 990                     <jo>
850 sound 1,800,10:sound 2,812,10 <me>
860 return                        <em>
870 rem verteidigen               <ea>
880 f(c)=1                        <fl>
890 gosub 810                     <kp>
900 return                        <jn>
```

```
910 rem sieger kontrolle mensch <bo>
920 for i=1 to 24 step 3 <ao>
930 a=sk(i):b=sk(i+1) <hh>
940 c=sk(i+2) <pk>
950 iff(a)+f(b)+f(c)<>30thennexti:
goto970 <ig>
960 char1,5,18,c2$+rn$+f1$+c3$+c3$
+"bravo sie haben gewonnen":goto12
60 <ja>
970 gosub 1820 <ib>
980 return <dn>
990 rem sieger kontrolle computer <do>
1000 for i=1 to 24 step 3 <ko>
1010 a=sk(i):b=sk(i+1):c=sk(i+2) <li>
1020 if f(a)+f(b)+f(c)=3 then 1180 <pa>
1030 next i <hg>
1040 gosub 1820 <fi>
1050 return <mk>
1060 rem 3 nebeneinander machen <ae>
1070 restore <pa>
1080 gosub 1820 <ep>
1090 for i=1 to 24 <dd>
1100 read a,b,c <po>
1110 if f(a)+f(b)=2 and f(c)=0 the
n cz=c:goto 1170 <oh>
1120 if f(a)+f(c)=2 and f(b)=0 the
n cz=b:goto 1170 <pf>
1130 if f(b)+f(c)=2 and f(a)=0 the
n cz=a:goto 1170 <ha>
1140 next i <cn>
1150 gosub 1820 <ch>
1160 return <kh>
1170 gosub 810 <ej>
1180 char1,3,18,re$:printc2$rn$f1$
"anfaenger !!!"rf$" ich habe gewon
nen !!!"bk$:goto1260 <cm>
1190 rem zufaellig saetzen <eb>
1200 c=int(9*rnd(1))+1 <oh>
1210 if f(c)<>0 then goto 1200 <gk>
1220 f(c)=1:cz=c <kd>
1230 gosub 1820 <cb>
1240 gosub 810 <kf>
1250 return <fl>
1260 getkey a$ <ka>
1270 char 1,10,20,chr$(27)+"q" <af>
1280 char 1,5,20,rf$+fo$+" wollen
sie nocheinmal (j/n) "+f1$+"?"+fo$ <bi>
1290 getkey a$ <ig>
1300 if a$="j" then run <ik>
1310 if a$="n" then sys 65526 <nd>
1320 goto 1290 <co>
1330 rem fehler <mk>
1340 resume <fi>
1350 rem erster schritt i und ii <bm>
1360 cz=(int(5*rnd(1))+1)*2-1 <ei>
1370 gosub 1820 <ld>
1380 if f(cz)<>0 then 1360 <fb>
1390 f(cz)=1 <fa>
1400 gosub 810 <kp>
1410 return <jn>
1420 rem feld init <fl>
1430 restore 1990 <en>
1440 for i=1 to 9 <po>
1450 read a,b,c <og>
1460 fp(i,0,0)=a:fp(i,1,0)=b:fp(i,
1,1)=c <ci>
1470 next i <fl>
1480 restore 2040 <fn>
1490 for i=1 to 24 <aa>
1500 read a <nf>
1510 sk(i)=a <ab>
1520 next i <cd>
1530 return <in>
1540 ---------- <ck>
1550 char 1,3,20,"wer faengt an " <bi>
1560 printf1$+"?"fo$" [computer="r
n$"c"rf$" mensch="rn$"m"rf$"]" <ga>
1570 getkey a$ <bj>
1580 if a$="m" then at$="c":ti$="0
00000":goto300 <ma>
1590 if a$="c" then at$="m":ti$="0
00000":gosub 1360:goto 300 <el>
1600 goto 1570 <em>
1610 char 1,2,7,"UCCCCCCCCI" <pk>
1620 char 1,2,8,"BcomputerB" <jh>
1630 char 1,2,9,za$+"CCCCCCCC"+zi$ <fk>
1640 char1,2,10,"B"+re$+"00:00:00"
+bk$+"B" <dp>
1650 char1,2,11,"JCCCCCCCCK" <kj>
1660 char 1,26,7,"UCCCCCCCCI" <op>
1670 char 1,26,8,"B mensch B" <mj>
1680 char 1,26,9,za$+"CCCCCCCC"+zi
$ <da>
1690 char1,26,10,"B"+re$+"00:00:00
"+bk$+"B" <nc>
1700 char1,26,11,"JCCCCCCCCK" <fe>
1710 return <ph>
1720 rem schwierigkeitsstufe <fn>
1730 char1,0,20,"stufe " <ci>
1740 printf1$"?"fo$" ["rn$"1"rf$"]
leicht ["rn$"2"rf$"]mittel ["rn$"3
"rf$"]schwierig" <hj>
1750 gets$:s=val(s$) <lo>
1760 if s<1 or s>3 then 1750 <ca>
1770 char 1,0,20,chr$(27)+"q" <co>
1780 char 1,14,2,"UCCCCCCCCI <fk>
1790 char 1,14,3,"B"+gr$+"stufe:"+
str$(s)+bk$+"B" <fh>
1800 char 1,14,4,"JCCCCCCCCK <fh>
1810 return <ma>
1820 if at$="c" then 1860 <mn>
1830 ct$=ti$:ze$=re$+left$(ct$,2)+
":"+right$(left$(ct$,4),2)+":"+rig
ht$(ct$,2)+bk$ <bp>
1840 char 1,3,10,ze$ <ak>
1850 return <ba>
```

```
1860 mt$=ti$:ze$=re$+left$(mt$,2)+
":"+right$(left$(mt$,4),2)+":"+rig
ht$(mt$,2)+bk$                    <ki>
1870 char 1,27,10,ze$             <kk>
1880 return                       <em>
1890 offene muehle                <bd>
1900 data 1,2,3,3,6,9,8,9,7,4,7,1 <fo>
1910 data 2,3,1,6,9,3,7,8,9,1,4,7 <ii>
1920 data 1,3,2,4,6,5,7,9,8       <mb>
1930 data 1,7,4,2,8,5,3,9,6       <af>
1940 data 1,9,5,3,7,5             <ak>
1950 data 1,5,9,5,9,1,7,5,3,5,3,7 <cb>
1960 data 4,5,6,5,6,4,2,5,8,5,8,2 <ek>
1970 ------------------------------ <bg>
1980 ---------  feld  ------------ <gk>
1990 data 1,2,1,2,5,1,3,8,1       <ig>
2000 data 4,2,4,5,5,4,6,8,4       <ea>
2010 data 7,2,7,8,5,7,9,8,7       <ke>
2020 ------------------------------ <no>
2030 *** sieger kontrol ***        <fn>
2040 data 1,2,3,4,5,6,7,8,9       <kp>
2050 data 1,4,7,2,5,8,3,6,9       <al>
2060 data 1,5,9,3,5,7             <jc>
2070 ------------------------------ <nh>
2080 -- schritt 1  --             <ie>
2090 data 1,2,3,3,6,9             <on>
2100 data 7,8,9,1,4,7             <ad>
2110 ------------------------------ <jn>
2120 rem nachspann ==============  <jl>
2130 rem * farbcodes/steuercodes * <no>
2140 c4$=chr$(017):rn$=chr$(018)  <lj>
2150 re$=chr$(028):c3$=chr$(029)  <ek>
2160 gr$=chr$(030):fl$=chr$(130)  <aa>
2170 fo$=chr$(132):bk$=chr$(144)  <em>
2180 c2$=chr$(145):rf$=chr$(146)  <db>
2190 lb$=chr$(154):c1$=chr$(157)  <ab>
2200 rem *** zeichensatz/graphik * <hc>
2210 za$=chr$(171):zg$=chr$(177)  <dj>
2220 zh$=chr$(178):zi$=chr$(179)  <oo>
2230 ym$=chr$(219)                <ig>
2240 rem ********* zeichenfolgen * <dd>
2250 for q=1 to 40                <il>
2260 qd$=qd$+c4$                  <hd>
2270 next q                       <op>
2280 return                       <gp>
2290 rem ======================== <cm>
2300 rem 12277 bytes memory       <op>
2310 rem 05479 bytes program      <bn>
2320 rem 00287 bytes variables    <om>
2330 rem 00400 bytes arrays       <mm>
2340 rem 00469 bytes strings      <bi>
2350 rem 05642 bytes free (0)     <od>
2360 rem ======================== <il>
```

LISTINGS

23 neue Befehle für Sie

Suchen, ersetzen, mergen und vieles mehr können Sie mit den neuen Befehlen dieser BASIC-Erweiterung. Testen Sie, und Sie werden sie nicht mehr missen mögen.

AUTO

AUTO wurde in zwei Punkten geändert. Zum einen wird ein Zeichen zwischen der Zeilennummer und dem Cursor gelöscht und nicht nur übersprungen. Zum anderen werden bereits existierende Zeilennummern revers darsgestellt, um zu warnen.

TRON

TRON ist ebenfalls modifiziert. Die Zeilennummer wird in der HOME-Position revers ausgegeben. Man kann mit C= oder mit SHIFT verlangsamen. C verringert die Geschwindigkeit nur unwesentlich, SHIFT verlangsamt so, daß man mitschreiben kann. Mit SHIFT LOCK kann man den Langsamgang auch arretieren. Wie bei LIST ist es auch mit CTRL S möglich, den Prozeß zu stoppen und auf Tastendruck wieder zu starten. Wird mit CMD die Ausgabe auf einen Drucker gelenkt, so sind die Zeilennummern nicht revers.

OLD

OLD dient dazu, ein durch NEW versehentlich gelöschtes Programm wiederherzustellen.
Es darf zwischen NEW und OLD keine Variable definiert werden, weil dann das Programm zerstört wäre.

MERGE

MERGE dient dazu, zwei BASIC-Programme miteinander zu verknüpfen: Man lädt ein Programm normal mit LOAD oder DLOAD oder schreibt es direkt in den Speicher. Das zweite Programm lädt man mit MERGE nach. Ihm darf auch ein Maschinenprogramm folgen, nicht aber dem ersten, weil es sonst überschrieben würde.
Die Syntax ist wie die von LOAD.

DRUN

DRUN ist eine Kombination aus LOAD und RUN. Die Syntax ist wie die bei DLOAD. Es ist nicht möglich, eine Zeilennummer für RUN anzugeben.

DVERIFY

DVERIFY ist ein VERIFY auf Diskette; sonst wie VERIFY.
Die Syntax ist wie die von DLOAD.

DMERGE

DMERGE ist ein MERGE auf Diskette.
Die Syntax ist wie die von DLOAD.

FETCH(AD)

FETCH(AD) liest, ähnlich wie PEEK(AD), ein Byte aus dem ROM. PEEK liest aus dem RAM. Hierbei ist AD die gesuchte Speicherstelle.

LIST#

Dieser Befehl druckt ein Programmlisting aus. Die Syntax ist wie folgt:
LIST#G,[S][,A#][–[E#]]
Dabei bedeuten: G die Geräteadresse 1<G<8, S die Sekundäradresse, A#die erste Zeilennummer, E# die letzte Zeilennummer.
Wird S nicht angegeben, so gilt S=); für die Zeilennummern gilt die Syntax von LIST.
Beispiele:
LIST#4,7,10–80 listet auf Drucker vier in Groß-Kleinschrift von Zeile 10 bis 80.
LIST#5,,10– listet auf Drucker fünf in Groß-

Grafikschrift ab Zeile 10. LIST#3 listet das komplette Programm auf den Bildschirm.

SPLIT
SPLIT erzeugt zwei BASIC-Speicher und stellt den Grafikspeicher bereit.
Die beiden Bereiche sind unabhängig; man kann in jedem Variablen gleichen Namens bereithalten.
Wird, während Split aktiv ist, GRAPHIC CLR eingegeben, so wird nur GRAPHIC 0 ausgeführt.

CHANGE
CHANGE dient zum Auswählen des gewünschten Speicherteils. CHANGE wechselt immer zum gerade inaktiven Teil. Ist noch nicht SPLIT gegeben worden, so erfolgt die Meldung OUT OF MEMORY ERROR.

PART
Dies ist eine numerische Variable und gibt den gerade aktiven Speicherbereich an. Hierbei gilt:
0 = kein Speichersplit,
1 = Speicherbereich 1 und
2 = Speicherberiech 2.

NOSPLIT
NOSPLIT hebt Speichersplit wieder auf und gibt den Grafikspeicher wieder frei.

SEARCH
SEARCH ist ein Suchbefehl, mit dem Texte oder Befehle im Programm gesucht und ersetzt werden können.
Die Syntax ist folgendermaßen: SEARCH S1$ [,[A][,E]][REPLACE S2$]
Hierin sind:
S1$ = Suchstring,
A = erste Zeilennummer
E = letzte Zeilennummer,
S2$ = Ersatzstring.
Statt REPLACE wird auch ein Semikolon akzeptiert.
Ein gültiger Such- bzw. Ersatzstring ist:
1.) in Anführungszeichen gesetzte Texte.
2.) Stringvariable.
3.) In Apostrophe gesetzte Befehle.
4.) Eine belibige Kombination aus 1.) bis 3.).
Der Suchstring darf nicht leer sein, der Ersatzstring schon – zum Löschen.
Um ein Anführungszeichen zu suchen, kann man CHR$(34) verwenden. Ein Apostroph kann auch durch CHR$(1)+ string ersetzt werden, aber dann muß der nachfolgende String bereits tokenisiert vorliegen.
Um einen Text nach REM zu finden, muß man den gesuchten Text in Anführungszeichen setzen. Um Text, der nicht in Anführungszeichen steht, hinter DATA zu finden, benutzt man CHR$(1)+".
Im Such- oder Ersatzstring darf kein CHR$(0) vorkommen, da ein Nullbyte ein Zeilenende markiert. Kommt CHR$(1) vor, so bleibt dies ohne Wirkung. Es ist zu beachten, daß einige Zeichen in verschiedenen Bedeutungen auftreten können, zum Beispiel kann ein * als Multiplikations-Zeichen tokenisiert oder als Stern übernommen werden. Im ersten Fall ersetzt man durch '*' und im zweiten durch "*".
Ein paar Beispiele:
SEARCH'COLOR1,5' REPLACE'COLOR1,7,3' ersetzt im ganzen Programm die Befehlsfolge COLOR1,5 durch COLOR1,7,3.
SEARCH"HUND",100 ;"" sucht das Wort HUND ab Zeile 100 und löscht es.
SEARCH'?',,400 sucht den Befehl PRINT im Programm bis Zeile 400.
SEARCHCHR$(1)+"*" sucht im Programm unter den Befehlen einen Stern. Dies kann normalerweise nicht vorkommen, da der BASIC-Interpreter bei der Programmeingabe einen Stern in ein Multiplikations-Zeichen tokenisiert.
Eine gefundene Zeile wird in der Form <Zeilen#> auf dem Bildschirm angezeigt. Jede Zeile wird nur einmal ausgewiesen, auch wenn der Suchstring mehrmals in ihr vorkommt.
Beim Ersetzen wird so oft ersetzt, wie der String in der Zeile vorkommt.
Man kann mit RUN/STOP den Prozeß abbrechen, doch immer erst dann, wenn eine Zeile komplett abgearbeitet worden ist (daher länger auf der Taste bleiben).
Zu den Fehlermeldungen:
SYNTAX ERROR tritt auf, wenn die Syntax nicht eingehalten wird, der Suchstring leer ist oder ein CHR$(0) vorkommt.
TYPE MISMATCH ERROR ist im Handbuch erklärt.
ILLEGAL QUANTITY ERROR tritt auf, wenn die Zeilennummer unzulässig ist, also A>E, E>63999 oder negative Zeilennummern.
OUT OF MEMORY ERROR kann auftreten, weil der Programmtext beim Ersetzen länger werden kann.
STRING TOO LONG ERROR kann mehrere Ursachen haben: der Such- oder Ersatzstring ist zu lang, beim Ersetzen wird eine BASIC-Zeile länger als 255 Zeichen oder beim Löschen würde eine Zeile ganz wegfallen.
FORMULA TOO KOMPLEX ERROR kann auftreten, wenn der Such- oder Ersatzstring zu umfangreich ist, zu viele Apostrophe enthält.
Da der Suchbefehl das Programm editiert und somit der Programmtext länger oder kürzer werden kann, werden alle Variablen gelöscht.
Es gibt noch eine interessante Anwendung bei der Erstellung von Programmen:
Will man einen Variablennamen benutzen, der ein BASIC-Wort enthält, etwa WINDOW (enthält DO), so benutzt man zuerst eine Hilfsvariable wie QQ und ersetzt dann mit SEARCH'QQ'; "WINDOW" die Variable im Programm. Man erhält dadurch keinen SYNTAX ERROR mehr. Allerdings werden auch hier nur die ersten beiden Buchstaben, also WI, zur Erkennung verwendet.

FAST
FAST dient zum Ausschalten des Bildschirms, um die Rechengeschwindigkeit um die Hälfte zu beschleunigen.

SLOW
SLOW schaltet den Bildschirm wieder ein.

SET
SET dient zum Setzen des Cursors. Die Syntax ist SET Z,S.
Z bezeichnet die Zeile und S die Spalte, in die der Cursor gesetzt werden soll. Für Z sind die Werte Null bis 24 und für S die Werte Null bis 39 zulässig. Ein nachfolgender PRINT-Befehl bezieht sich auf die angegebene Stelle am Bildschirm. Im Direktmodus allerdings nur dann korrekt, wenn der PRINT-Befehl gleichzeitig mit SET eingegeben wird. Sonst würde nach der Ausgabe von READY der Cursor zwei Zeilen tiefer stehen.

GENDAT
Mit diesem Befehl kann man einen Speicherbereich in DATA-Zeilen ablegen. Die Syntax ist folgendermaßen:
GENDAT A TO B [START C][STEP D] [INC]
Hierin bedeuten:
A = Bereichsanfang.
B = Bereichsende.
C = BASIC-Zeilennummer, bei der die Datas beginnen sollen, Default=10.
D = Schrittweite der Zeilennummern (siehe AUTO), Default=10.
Wird INC angegeben, so wird eine Prüfsumme eingebunden.

Bitte lesen Sie weiter auf Seite 127

```
BASICERWEITERUNG==============C16
(P) COMMODORE WELT TEAM
==================================
(C) by Stephan Bielicke

C16/116/P4
==================================
Das Programm ist mit dem Maschinen-
sprachmonitor TEDMON unter Zuhilfe-
nahme des Pruefsummenprogrammes
CHECKMON einzugeben und auf Disket-
te mit s"Basic.erw",8,1001,26d6 ab-
zuspeichern, auf Kassette mit
s"basic.erw",1,1001,26d6.
Danach kann das Programm wie ein
BASIC-Programm gehandhabt werden.
==================================
>1000 00 60 10 0a 00 9e c2 28 :<99>
>1008 34 33 29 aa 32 35 36 ac :<19>
>1010 c2 28 34 34 29 aa 39 37 :<52>
>1018 20 3a 22 14 14 14 14 14 :<cc>
>1020 14 14 14 14 14 14 14 14 :<2e>
>1028 14 14 14 14 14 14 14 14 :<5c>
>1030 14 14 14 14 14 14 14 14 :<4c>
>1038 14 14 14 14 42 49 45 4c :<52>
>1040 49 43 4b 45 27 53 20 42 :<93>
>1048 41 53 49 43 45 52 57 45 :<b6>
>1050 49 54 45 52 55 4e 47 20 :<3a>
>1058 56 31 35 30 34 38 38 00 :<27>
>1060 00 00 a5 2b 85 5f a5 2c :<30>
>1068 85 60 a5 2d 85 5a a5 2e :<f6>
>1070 85 5b a9 d6 85 58 a9 26 :<d6>
>1078 85 59 a9 10 48 a9 82 48 :<97>
>1080 4c c7 88 a2 00 a0 e8 8e :<4a>
>1088 33 05 8c 34 05 a2 34 86 :<02>
>1090 57 a0 11 84 58 a2 38 86 :<14>
>1098 59 a0 11 84 5a a2 7d a0 :<4c>
>10a0 04 20 11 11 a2 38 86 57 :<7c>
>10a8 a0 11 84 58 a2 c6 86 59 :<98>
>10b0 a0 11 84 5a a2 5e a0 06 :<c9>
>10b8 20 11 11 a2 c6 86 57 a0 :<68>
>10c0 11 84 58 a2 d6 86 59 a0 :<ca>
>10c8 11 84 5a a2 02 a0 03 20 :<11>
>10d0 11 11 a2 d6 86 57 a0 11 :<d1>
>10d8 84 58 a2 d6 86 59 a0 26 :<d0>
>10e0 84 5a a2 00 a0 e8 20 11 :<38>
>10e8 11 a2 00 20 55 80 20 4f :<ab>
>10f0 ff 93 0d 20 45 58 54 45 :<e1>
>10f8 4e 44 45 44 20 42 41 53 :<2e>
>1100 49 43 20 56 30 34 2e 38 :<34>
>1108 38 20 00 20 e5 80 4c 22 :<dc>
>1110 80 86 5b 84 5c a0 00 b1 :<6d>
>1118 57 91 5b e6 57 d0 02 e6 :<26>
>1120 58 e6 5b d0 02 e6 5c a5 :<a7>
>1128 58 c5 5a d0 ea a5 57 c5 :<92>
>1130 59 90 e4 60 4c 93 f6 ea :<25>
>1138 78 8d 3f ff 4c 7f fa 78 :<46>
>1140 8d 3f ff 4c 4c fa 78 8d :<e4>
>1148 3f ff 4c 1a fa 8d 3e ff :<20>
>1150 58 8d 3e ff 4c 00 00 8d :<a1>
>1158 3e ff 58 8d 3e ff 20 00 :<8f>
>1160 00 78 8d 3f ff 60 78 8d :<33>
>1168 3f ff 4c 1c f8 8d 3e ff :<4d>
>1170 b1 14 8d 3f ff 60 78 8d :<f7>
>1178 3f ff 4c c3 f7 20 73 04 :<b7>
>1180 f0 19 2c eb 02 10 11 24 :<b4>
>1188 81 10 0d 48 78 8d 3f ff :<a2>
>1190 20 54 f7 8d 3e ff 58 68 :<41>
>1198 20 3f 8c 4c dc 8b 78 8d :<00>
>11a0 3f ff 20 98 f4 8d 3e ff :<d4>
>11a8 58 60 00 00 00 00 00 00 :<7b>
>11b0 00 00 00 00 00 00 00 00 :<72>
>11b8 00 00 00 00 00 00 00 00 :<82>
>11c0 00 00 00 00 00 ff 9c 06 :<9c>
>11c8 56 89 6e 8b a3 06 8c 06 :<ec>
>11d0 5e 06 65 06 6c 06 00 00 :<68>
>11d8 00 00 00 00 00 00 00 00 :<c2>
>11e0 00 00 00 20 16 e8 f0 01 :<ae>
>11e8 60 4c f6 ff a9 2b 8d 85 :<00>
>11f0 06 a9 cd 8d 86 06 4c 7d :<1c>
>11f8 06 c9 28 f0 08 a2 0b 2c :<47>
>1200 a2 0e 4c 0c f7 20 68 e8 :<df>
>1208 85 61 a9 00 85 25 a8 aa :<48>
>1210 c4 61 f0 26 b1 22 c8 c9 :<47>
>1218 20 f0 f5 e6 25 8a a6 25 :<f4>
>1220 e0 09 f0 dc aa 88 b1 22 :<bb>
>1228 c8 c9 32 b0 d3 c9 30 90 :<1c>
>1230 cf 29 01 c9 01 8a 2a 4c :<6f>
>1238 39 e8 a8 4c 08 f6 a0 48 :<d3>
>1240 8c 85 06 a0 9c d0 07 a0 :<2a>
>1248 5c 8c 85 06 a0 9b 8c 86 :<ed>
>1250 06 4c 7d 06 c9 28 f0 03 :<84>
>1258 4c 47 fa 20 26 f7 20 4c :<e5>
>1260 f6 86 78 20 26 f7 a9 08 :<32>
>1268 20 71 e8 a0 07 a5 78 6a :<60>
>1270 aa a9 30 69 00 91 62 8a :<2b>
>1278 88 10 f4 4c 19 f7 20 f5 :<0f>
>1280 e9 a5 0d f0 16 38 a5 5f :<7c>
>1288 e5 2f a5 60 e5 30 90 0b :<b9>
>1290 a0 04 b1 5f c9 01 f0 08 :<4d>
>1298 a2 12 2c a2 16 4c 0c f7 :<f0>
>12a0 18 a9 04 65 5f 85 5f 90 :<1e>
>12a8 02 e6 60 a0 01 b1 5f d0 :<bd>
>12b0 08 c8 b1 5f c9 01 d0 01 :<eb>
>12b8 60 a0 00 84 22 84 23 a6 :<f7>
>12c0 22 a4 23 e8 d0 01 c8 86 :<1a>
>12c8 24 84 25 20 69 e9 e6 24 :<91>
>12d0 d0 02 e6 25 a0 01 b1 5f :<3e>
>12d8 c5 25 d0 ef c8 b1 5f c5 :<5a>
>12e0 24 d0 e8 e6 22 d0 02 e6 :<f7>
>12e8 23 a0 02 38 b1 5f e5 22 :<51>
>12f0 aa 88 b1 5f e5 23 d0 c7 :<70>
>12f8 e0 01 d0 c3 a2 19 a9 00 :<f2>
>1300 95 57 ca 10 fb 60 a2 00 :<42>
>1308 2c a2 02 b5 22 0a a8 b5 :<68>
>1310 23 2a 48 18 98 75 22 a8 :<08>
```

```
>1318 68 75 23 48 18 98 65 5f :<b0>
>1320 a8 68 65 60 48 18 98 69 :<3c>
>1328 03 95 68 95 57 68 69 00 :<8c>
>1330 95 69 95 58 a1 68 95 6c :<de>
>1338 f6 68 d0 02 f6 69 60 20 :<50>
>1340 30 e9 20 33 e9 a0 00 b1 :<12>
>1348 68 85 76 b1 6a 85 49 c8 :<d7>
>1350 b1 68 85 77 b1 6a 85 4a :<c2>
>1358 a0 ff c8 c4 6c f0 1c c4 :<44>
>1360 6e f0 09 38 b1 49 f1 76 :<3a>
>1368 f0 f0 b0 0f a0 02 b1 57 :<bc>
>1370 aa b1 59 91 57 8a 91 59 :<04>
>1378 88 10 f3 60 20 f5 e9 85 :<27>
>1380 76 84 77 a5 0d 48 a5 0e :<9a>
>1388 48 20 29 f7 c9 2c f0 08 :<1a>
>1390 a2 0b 2c a2 16 4c 0c f7 :<13>
>1398 20 26 f7 20 f5 e9 85 49 :<bf>
>13a0 84 4a 68 c5 0e d0 ec 68 :<dc>
>13a8 c5 0d d0 e7 aa d0 0a a5 :<33>
>13b0 0e f0 03 a0 01 2c a0 04 :<ca>
>13b8 2c a0 02 b1 76 85 78 b1 :<5c>
>13c0 49 91 76 a5 78 91 49 88 :<3d>
>13c8 10 f1 60 a9 a5 8d 85 06 :<7d>
>13d0 a9 96 8d 86 06 4c 7d 06 :<df>
>13d8 a9 8d 85 02 20 25 ea f0 :<c4>
>13e0 05 a2 0c 4c 0c f7 20 2e :<70>
>13e8 ea a0 05 20 37 ea a9 b0 :<cf>
>13f0 8d 85 06 a9 8d 8d 86 06 :<a7>
>13f8 4c 7d 06 a9 71 8d 85 06 :<cd>
>1400 a9 88 d0 f1 a9 69 8d 85 :<52>
>1408 06 a9 a7 d0 e8 a9 72 d0 :<cc>
>1410 f5 f0 03 4c 47 fa a5 14 :<5e>
>1418 48 a5 15 48 a9 d1 85 14 :<7c>
>1420 a9 f3 85 15 a0 3a 20 93 :<30>
>1428 06 99 5e 05 88 d0 f7 8c :<41>
>1430 5d 05 68 85 15 68 85 14 :<0a>
>1438 60 20 4c f6 c9 2c f0 03 :<91>
>1440 4c 47 fa 86 76 20 26 f7 :<e3>
>1448 20 f7 f5 85 4f 84 4e a2 :<d4>
>1450 01 20 29 f7 f0 07 c9 2c :<ba>
>1458 d0 e6 20 49 f6 86 77 e8 :<c4>
>1460 f0 03 ca d0 03 4c 19 f6 :<6b>
>1468 a5 76 20 e7 ea f0 05 a2 :<db>
>1470 03 4c 0c f7 86 02 20 f4 :<21>
>1478 ea a9 00 20 c0 f4 a6 02 :<67>
>1480 bd 13 05 aa a9 00 a0 6f :<d4>
>1488 20 bb f4 20 c5 f4 a2 00 :<be>
>1490 20 d4 f4 a9 50 20 45 f7 :<ab>
>1498 a5 76 09 60 20 45 f7 a5 :<af>
>14a0 4e 20 45 f7 a5 4f 20 45 :<6b>
>14a8 f7 a5 77 20 45 f7 a9 0d :<66>
>14b0 20 45 f7 20 d9 f4 a9 00 :<33>
>14b8 38 20 ca f4 60 a2 ed 8e :<91>
>14c0 85 06 a2 ee 8e 86 06 4c :<4d>
>14c8 7d 06 a9 f8 8d 85 06 a9 :<44>
>14d0 a8 8d 86 06 d0 f1 20 be :<bf>
>14d8 f5 a0 ff c8 b1 3b f0 3d :<41>
>14e0 c9 22 d0 09 c8 b1 3b f0 :<d4>
>14e8 34 c9 22 d0 f7 c9 3a f0 :<c1>
>14f0 2c c9 8f f0 04 c9 83 d0 :<97>
>14f8 e2 c8 b1 3b f0 1f a5 3b :<b8>
>1500 48 a5 3c 48 98 48 18 65 :<14>
>1508 3b 85 3b 90 02 e6 3c 20 :<98>
>1510 a6 ef 68 a8 68 85 3c 68 :<f2>
>1518 85 3b 18 90 be 20 29 f7 :<c9>
>1520 d0 03 4c 47 fa a0 ff c8 :<cd>
>1528 b1 3b c9 22 d0 09 c8 b1 :<39>
>1530 3b f0 13 c9 22 d0 f7 c9 :<61>
>1538 27 d0 03 20 f1 ee c9 3a :<75>
>1540 f0 04 c9 00 d0 e1 a2 19 :<be>
>1548 86 16 20 0d f1 20 49 ef :<1d>
>1550 20 0a f1 a4 19 d0 03 4c :<45>
>1558 47 fa 88 b1 1a f0 f8 88 :<09>
>1560 10 f9 a2 00 8e da 06 86 :<c6>
>1568 03 86 04 ca 86 05 a2 f9 :<32>
>1570 86 06 20 29 f7 c9 2c d0 :<cd>
>1578 27 20 26 f7 c9 2c f0 12 :<cf>
>1580 20 f7 f5 a6 14 86 03 a6 :<4b>
>1588 15 86 04 20 29 f7 c9 2c :<0a>
>1590 d0 0e 20 26 f7 20 f7 f5 :<7f>
>1598 a6 14 86 05 a6 15 86 06 :<f0>
>15a0 20 29 f7 f0 21 c9 3b f0 :<1d>
>15a8 0e c9 fe d0 07 20 26 f7 :<f7>
>15b0 c9 96 f0 03 4c 47 fa ee :<65>
>15b8 da 06 20 26 f7 20 0d f1 :<8d>
>15c0 20 3a ef 20 0a f1 ad da :<fe>
>15c8 06 f0 0c a4 1c f0 08 88 :<df>
>15d0 b1 1d f0 e0 88 10 f9 a0 :<f9>
>15d8 00 b1 1a c8 c9 01 d0 06 :<b2>
>15e0 c4 19 d0 f5 f0 ce a5 04 :<b7>
>15e8 c9 fa 90 03 4c 19 f6 a5 :<b1>
>15f0 06 c9 fa b0 f7 38 a5 05 :<1e>
>15f8 e5 03 a5 06 e5 04 90 ec :<ed>
>1600 a5 03 85 14 a5 04 85 15 :<fe>
>1608 20 36 f7 a4 19 88 b1 1a :<6f>
>1610 c9 01 d0 02 84 19 a0 00 :<83>
>1618 b1 5f c8 11 5f d0 03 4c :<34>
>1620 cd fc a0 02 38 a5 05 f1 :<7f>
>1628 5f c8 a5 06 f1 5f 90 ef :<72>
>1630 a6 5f a4 60 86 14 84 15 :<82>
>1638 a0 00 b1 5f aa c8 b1 5f :<6f>
>1640 86 5f 85 60 18 a5 14 69 :<ba>
>1648 03 85 22 a5 15 69 00 85 :<a1>
>1650 23 a2 00 86 78 20 a1 ec :<ec>
>1658 b0 bc a9 02 65 14 85 14 :<e5>
>1660 90 02 e6 15 a9 3c 20 45 :<27>
>1668 f7 20 74 ef a9 3e 20 45 :<df>
>1670 f7 20 59 ed 4c 40 ec 20 :<52>
>1678 af ef e6 22 d0 02 e6 23 :<93>
>1680 20 f7 ec e8 c8 20 58 ef :<22>
>1688 b1 1a 20 58 ef c9 01 d0 :<77>
>1690 0e e8 e4 19 d0 02 38 60 :<e8>
>1698 45 76 85 76 18 90 e6 b1 :<80>
>16a0 22 f0 f3 20 16 ed 10 06 :<2c>
>16a8 a5 76 f0 06 d0 e8 c5 76 :<dd>
>16b0 f0 c8 20 ae ee b0 c3 38 :<2e>
```

```
>16b8 b1 22 20 58 ef f1 1a 08 :<5f>
>16c0 20 58 ef 28 d0 b4 e8 e4 :<dc>
>16c8 19 d0 b9 18 60 a2 00 86 :<91>
>16d0 76 ca a5 78 85 77 a0 00 :<0e>
>16d8 b1 22 20 16 ed 08 a5 78 :<e1>
>16e0 a4 77 28 30 02 85 77 84 :<dd>
>16e8 78 a0 ff 60 c9 8f d0 23 :<a2>
>16f0 a5 77 d0 1f a5 22 d0 02 :<59>
>16f8 c6 23 c6 22 b1 22 e6 22 :<15>
>1700 d0 02 e6 23 c9 fe f0 0b :<69>
>1708 a5 77 48 49 80 85 77 68 :<bf>
>1710 18 90 19 c9 22 08 a5 77 :<21>
>1718 28 d0 11 c9 00 d0 07 49 :<fb>
>1720 01 85 77 18 90 06 48 49 :<2a>
>1728 01 85 77 68 48 68 60 ad :<28>
>1730 da 06 d0 01 60 18 a5 14 :<90>
>1738 69 01 85 22 a5 15 69 00 :<6d>
>1740 85 23 a2 00 86 78 20 a4 :<9d>
>1748 ec 90 03 4c 06 f5 a6 77 :<47>
>1750 8e 00 01 20 b5 ed a2 ff :<1b>
>1758 a0 ff c8 e8 e4 1c f0 11 :<0a>
>1760 20 58 ef b1 1d 20 58 ef :<4d>
>1768 c9 01 f0 ef 91 22 18 90 :<05>
>1770 e9 ae 00 01 86 77 18 98 :<5d>
>1778 65 22 85 22 90 02 e6 23 :<2b>
>1780 a5 22 d0 02 c6 23 c6 22 :<93>
>1788 4c 70 ed 20 61 ef 86 20 :<6f>
>1790 a2 ff a0 ff e8 c8 c4 1c :<15>
>1798 f0 08 b1 1d c9 01 f0 f5 :<5b>
>17a0 d0 f2 86 21 a6 20 e4 21 :<17>
>17a8 d0 01 60 90 03 4c 34 ee :<f2>
>17b0 20 92 ee 90 05 a2 17 4c :<09>
>17b8 0c f7 38 a5 21 e5 20 85 :<f9>
>17c0 76 a6 2d 86 24 a6 2e 86 :<20>
>17c8 25 18 65 2d aa a9 00 65 :<6f>
>17d0 2e c5 34 90 0b f0 05 a2 :<86>
>17d8 10 4c 0c f7 c4 33 b0 f7 :<a3>
>17e0 86 2d 85 2e a2 00 a4 76 :<36>
>17e8 a1 24 91 24 a5 24 d0 02 :<e9>
>17f0 c6 25 c6 24 38 a5 24 e5 :<d5>
>17f8 22 a5 25 e5 23 b0 e9 a5 :<64>
>1800 76 65 5f 85 5f 90 02 e6 :<69>
>1808 60 60 20 7b ee 90 03 4c :<61>
>1810 df ed 38 a5 20 e5 21 85 :<78>
>1818 76 a6 22 86 24 a6 23 86 :<48>
>1820 25 38 a5 2d e5 76 85 2d :<be>
>1828 b0 02 c6 2e a4 76 b1 24 :<f4>
>1830 a0 00 91 24 e6 24 d0 02 :<57>
>1838 e6 25 38 a5 2d e5 24 a5 :<08>
>1840 2e e5 25 b0 e7 38 a5 5f :<31>
>1848 e5 76 85 5f b0 02 c6 60 :<60>
>1850 60 38 a5 5f e5 14 a8 88 :<0d>
>1858 88 88 38 a5 20 e5 21 8c :<12>
>1860 01 01 ed 01 01 b0 18 60 :<53>
>1868 38 a5 5f e5 14 a8 c8 c8 :<5b>
>1870 38 a5 21 e5 20 8c 01 01 :<5a>
>1878 18 6d 01 01 b0 01 60 20 :<58>
>1880 06 f5 38 60 c0 00 f0 02 :<e5>
>1888 18 60 e0 00 f0 fa a5 76 :<22>
>1890 f0 f6 20 58 ef 38 b1 1a :<c7>
>1898 08 20 58 ef 28 10 e9 f1 :<d1>
>18a0 22 d0 e5 a5 22 d0 02 c6 :<5b>
>18a8 23 c6 22 b1 22 e6 22 d0 :<7d>
>18b0 02 e6 23 c9 fe d0 d1 ca :<0d>
>18b8 20 58 ef b1 1a 20 58 ef :<cc>
>18c0 e8 c9 fe f0 c3 38 60 20 :<a7>
>18c8 13 ef a9 22 91 3b c8 a9 :<85>
>18d0 01 91 3b c8 b1 3b f0 10 :<a5>
>18d8 c9 27 d0 f7 20 13 ef a9 :<4c>
>18e0 01 91 3b c8 a9 22 91 3b :<b7>
>18e8 60 a2 ff e8 e0 57 d0 05 :<29>
>18f0 a2 19 4c 0c f7 bd 00 02 :<67>
>18f8 d0 f1 18 98 65 3b 85 76 :<22>
>1900 a2 58 ca bd 00 02 e8 9d :<85>
>1908 00 02 ca e4 76 d0 f3 60 :<c1>
>1910 a0 02 b1 64 99 1c 00 88 :<04>
>1918 10 f8 a9 1f 85 16 60 a0 :<eb>
>1920 02 b1 64 99 19 00 88 10 :<8e>
>1928 f8 a9 1c 85 16 60 48 98 :<52>
>1930 48 8a a8 68 aa 68 60 a2 :<4b>
>1938 ff a0 ff e8 c8 c4 19 f0 :<48>
>1940 08 b1 1a c9 01 f0 f5 d0 :<25>
>1948 f2 60 a2 5f a0 a4 8e 85 :<7c>
>1950 06 8c 86 06 a5 16 48 a2 :<11>
>1958 09 b5 18 48 ca d0 fa a0 :<cb>
>1960 00 b1 14 aa c8 b1 14 a0 :<48>
>1968 19 84 16 20 7d 06 a2 01 :<e6>
>1970 68 95 18 e8 e0 0a d0 f8 :<5b>
>1978 68 85 16 60 a0 53 8c 85 :<21>
>1980 06 a0 89 d0 15 20 bd ef :<01>
>1988 d0 03 4c fb f4 20 bd ef :<ef>
>1990 f0 f8 60 a0 e1 8c 85 06 :<38>
>1998 a0 ff 8c 86 06 4c 7d 06 :<8e>
>19a0 08 a9 00 28 08 48 8a 48 :<04>
>19a8 98 48 a2 0c 8e 85 06 a2 :<35>
>19b0 ef 8e 86 06 20 7d 06 a2 :<d0>
>19b8 c6 8e 85 06 a2 ff 8e 86 :<9e>
>19c0 06 a2 01 20 7d 06 68 a8 :<a1>
>19c8 68 aa 68 28 4c cb 06 48 :<08>
>19d0 a9 4a 8d 85 06 a9 f0 8d :<4e>
>19d8 86 06 68 20 7d 06 4c ce :<51>
>19e0 ef a2 5d 8e 85 06 a2 ee :<9b>
>19e8 8e 86 06 4c 05 f0 a9 ed :<ef>
>19f0 8d 85 06 a9 ee 8d 86 06 :<d3>
>19f8 a9 01 20 7d 06 85 97 8a :<35>
>1a00 a8 a2 d4 8e 85 06 a2 ee :<73>
>1a08 8e 86 06 4c 05 f0 a9 a4 :<08>
>1a10 8d 85 06 a9 f1 8d 86 06 :<44>
>1a18 d0 c1 20 0d f1 20 0a f1 :<92>
>1a20 a2 fd b5 64 48 e8 30 fa :<72>
>1a28 a2 08 20 29 f7 f0 0f c9 :<7b>
>1a30 2c f0 03 4c 47 fa 20 49 :<d4>
>1a38 f6 20 29 f7 d0 f5 e0 00 :<c0>
>1a40 d0 08 a2 09 2c a2 08 4c :<88>
>1a48 0c f7 e0 03 f0 f4 86 52 :<84>
>1a50 68 a8 68 aa 68 f0 ee 20 :<71>
```

```
>1a58 c0 f4 a9 01 a6 52 a0 6e :<b0>
>1a60 e0 01 d0 02 a0 00 20 bb :<3b>
>1a68 f4 20 c5 f4 90 0a 48 a9 :<42>
>1a70 01 20 ca f4 68 4c 29 f3 :<b6>
>1a78 a9 01 8d 1b 03 8d 21 03 :<4b>
>1a80 8d 2b 03 8d 2f 03 8d 31 :<85>
>1a88 03 a9 2b 8d 1a 03 a9 32 :<0e>
>1a90 8d 20 03 a9 39 8d 2a 03 :<6a>
>1a98 a9 47 8d 2e 03 a9 40 8d :<de>
>1aa0 30 03 a2 22 bd e7 f0 9d :<2b>
>1aa8 2b 01 ca 10 f7 a2 01 20 :<34>
>1ab0 cf f4 b0 3e 20 d9 f4 a2 :<c9>
>1ab8 d8 8a 4c 78 f5 78 8d 3f :<84>
>1ac0 ff 4c 0b f0 78 8d 3f ff :<55>
>1ac8 4c ca ef 78 8d 3f ff 4c :<a1>
>1ad0 18 f0 78 8d 3f ff 4c 38 :<f8>
>1ad8 f0 78 8d 3f ff 4c f9 ef :<0a>
>1ae0 a2 1a 2c a2 2c 8e 85 06 :<28>
>1ae8 a2 93 8e 86 06 4c 7d 06 :<27>
>1af0 d0 49 a9 0c 8d 20 03 a9 :<53>
>1af8 ef 8d 21 03 8d 2b 03 a9 :<83>
>1b00 5d 8d 1a 03 a9 ee 8d 1b :<0c>
>1b08 03 a9 08 8d 2a 03 a9 4a :<98>
>1b10 8d 2e 03 a9 f0 8d 2f 03 :<3b>
>1b18 a9 a4 8d 30 03 a9 f1 8d :<bb>
>1b20 31 03 a9 01 20 ca f4 4c :<5b>
>1b28 de f0 20 be f5 20 f7 f5 :<23>
>1b30 84 76 85 77 20 29 f7 c9 :<df>
>1b38 a4 f0 03 4c 47 fa 20 26 :<4c>
>1b40 f7 20 f7 f5 84 03 85 04 :<52>
>1b48 20 a8 f2 b0 03 4c 19 f6 :<5f>
>1b50 a9 0a a0 00 85 05 85 4e :<3c>
>1b58 84 06 84 4f 84 07 20 29 :<9a>
>1b60 f7 c9 fe d0 15 20 26 f7 :<35>
>1b68 c9 91 d0 31 20 26 f7 20 :<23>
>1b70 f7 f5 c9 fa b0 d7 84 05 :<09>
>1b78 85 06 20 29 f7 c9 a9 d0 :<80>
>1b80 12 20 26 f7 20 f7 f5 c9 :<81>
>1b88 fa b0 c2 84 4e 85 4f 05 :<54>
>1b90 4e f0 ba 20 29 f7 c9 fe :<f4>
>1b98 d0 0c 20 26 f7 c9 92 d0 :<ba>
>1ba0 9a c6 07 20 26 f7 20 29 :<7c>
>1ba8 f7 d0 90 a5 05 85 14 a5 :<82>
>1bb0 06 85 15 a0 00 84 8e 84 :<9a>
>1bb8 8f a9 83 99 00 02 a2 00 :<26>
>1bc0 a1 76 18 65 8e 85 8e 90 :<e9>
>1bc8 02 e6 8f a1 76 20 9f f2 :<cf>
>1bd0 20 cb f2 e6 76 d0 02 e6 :<2e>
>1bd8 77 f0 0d 20 a8 f2 90 04 :<47>
>1be0 c0 42 d0 da a5 07 f0 0b :<de>
>1be8 a5 8f 20 9f f2 a5 8e 20 :<6b>
>1bf0 9f f2 c8 a9 00 99 00 02 :<00>
>1bf8 c8 84 0b 18 a5 2d a4 2e :<c8>
>1c00 65 0b 90 01 c8 18 69 04 :<b9>
>1c08 90 01 c8 20 be f2 a9 01 :<a3>
>1c10 8d 03 03 a9 24 8d 02 03 :<ee>
>1c18 a2 06 bd 62 f2 9d 24 01 :<3c>
>1c20 ca 10 f7 a2 00 86 3b a2 :<7f>
>1c28 02 86 3c a2 36 8e 7b 06 :<3e>
>1c30 a2 87 8e 7c 06 4c 73 06 :<ab>
>1c38 78 8d 3f ff 4c 69 f2 a2 :<b2>
>1c40 9c 8e 02 03 a2 06 8e 03 :<45>
>1c48 03 18 a5 14 65 4e 85 14 :<3a>
>1c50 a5 15 65 4f 85 15 90 05 :<f5>
>1c58 a2 11 4c 0c f7 c9 fa b0 :<21>
>1c60 f7 a5 76 05 77 f0 0b 20 :<57>
>1c68 a8 f2 90 06 20 b2 f2 4c :<41>
>1c70 dd f1 4c fb f4 20 d3 f2 :<b5>
>1c78 20 ce f2 4c cd f2 38 a5 :<d7>
>1c80 03 e5 76 a5 04 e5 77 60 :<db>
>1c88 a0 c0 8c 85 06 a0 8c 8c :<24>
>1c90 86 06 d0 0a a2 23 8e 85 :<03>
>1c98 06 a2 89 8e 86 06 4c 7d :<95>
>1ca0 06 a2 2c 8a c8 99 00 02 :<44>
>1ca8 60 a2 20 8e 85 06 a2 fb :<fc>
>1cb0 8e 86 06 4c 7d 06 a9 e6 :<36>
>1cb8 20 c8 f3 20 d4 f3 a9 00 :<11>
>1cc0 8d 78 02 85 0a a0 05 20 :<33>
>1cc8 e0 f3 4c fe f2 a9 00 85 :<8b>
>1cd0 0a 20 31 f3 a2 fe 86 14 :<e9>
>1cd8 e8 86 15 20 36 f7 a6 5f :<73>
>1ce0 a4 60 a5 0a 20 86 f3 08 :<88>
>1ce8 20 95 f3 28 b0 11 20 a2 :<fb>
>1cf0 f3 29 bf d0 4c 86 2d 84 :<15>
>1cf8 2e 20 06 f5 4c fb f4 aa :<a1>
>1d00 d0 02 a2 1e 4c 0c f7 a2 :<ad>
>1d08 6b 8e 85 06 a2 a8 8e 86 :<ab>
>1d10 06 4c 7d 06 f0 37 a9 e6 :<e4>
>1d18 20 c8 f3 20 d4 f3 a9 00 :<d1>
>1d20 8d 78 02 85 0a a0 05 20 :<f3>
>1d28 e0 f3 a5 0a a6 2b a4 2c :<a8>
>1d30 20 86 f3 08 20 95 f3 28 :<9b>
>1d38 b0 c5 20 a2 f3 29 bf f0 :<98>
>1d40 05 a2 1d 4c 0c f7 86 2d :<52>
>1d48 84 2e 20 06 f5 68 68 a9 :<e9>
>1d50 be 8d 7b 06 a9 8b 8d 7c :<fc>
>1d58 06 4c 73 06 48 a9 d5 8d :<8b>
>1d60 85 06 a9 ff 8d 86 06 68 :<cc>
>1d68 4c 7d 06 48 a9 f8 8d 85 :<8c>
>1d70 06 a9 a8 8d 86 06 d0 ef :<e4>
>1d78 a9 b7 8d 85 06 a9 ff 8d :<81>
>1d80 86 06 4c 7d 06 a9 e6 20 :<e4>
>1d88 c8 f3 20 d4 f3 a9 00 8d :<0a>
>1d90 78 02 a9 01 85 0a a0 05 :<99>
>1d98 20 e0 f3 4c ec f3 a2 21 :<72>
>1da0 8e 85 06 a2 cb 8e 86 06 :<07>
>1da8 d0 22 a2 b5 8e 85 06 a2 :<62>
>1db0 cc 8e 86 06 d0 16 a2 3f :<f5>
>1db8 8e 85 06 a2 ca 8e 86 06 :<47>
>1dc0 d0 0a a2 fa 8e 85 06 a2 :<25>
>1dc8 a7 8e 86 06 4c 7d 06 20 :<72>
>1dd0 4c f6 86 76 e0 02 b0 05 :<ad>
>1dd8 a2 09 4c 0c f7 e0 08 b0 :<59>
>1de0 f7 a2 00 86 77 20 29 f7 :<80>
>1de8 f0 21 c9 2c f0 03 4c 47 :<3b>
>1df0 fa 20 26 f7 c9 2c f0 0e :<1c>
```

```
>1df8 20 4c f6 86 77 20 29 f7 :<0b>
>1e00 f0 09 c9 2c d0 e8 20 26 :<bd>
>1e08 f7 f0 e3 a9 00 20 c0 f4 :<c3>
>1e10 a9 00 a6 76 a4 77 20 bb :<cc>
>1e18 f4 20 c5 f4 90 0a 48 a9 :<e5>
>1e20 00 20 ca f4 68 4c 0b f7 :<97>
>1e28 a2 00 20 d4 f4 b0 ef ae :<17>
>1e30 26 03 8e db 06 ae 27 03 :<f8>
>1e38 8e dc 06 a2 c4 8e 26 03 :<f2>
>1e40 a2 06 8e 27 03 20 29 f7 :<b6>
>1e48 20 89 f4 ae db 06 8e 26 :<42>
>1e50 03 ae dc 06 8e 27 03 a9 :<8f>
>1e58 00 20 ca f4 4c d9 f4 08 :<b6>
>1e60 a2 ff 8e 85 06 a2 8a 8e :<bb>
>1e68 86 06 28 4c 7d 06 a2 65 :<a7>
>1e70 8e 85 06 a2 f2 8e 86 06 :<d0>
>1e78 20 7d 06 08 d0 11 a9 00 :<08>
>1e80 20 ca f4 ae db 06 8e 26 :<f4>
>1e88 03 ae dc 06 8e 27 03 28 :<bf>
>1e90 60 48 a9 ba d0 1c 48 a9 :<87>
>1e98 bd d0 17 48 a9 c0 d0 12 :<d2>
>1ea0 48 a9 c3 d0 0d 48 a9 c6 :<ab>
>1ea8 d0 08 48 a9 c9 d0 03 48 :<a4>
>1eb0 a9 cc 8d 85 06 a9 ff 8d :<ba>
>1eb8 86 06 68 4c 7d 06 08 20 :<d0>
>1ec0 be f5 28 d0 53 a9 01 a0 :<41>
>1ec8 00 91 2b 20 06 f5 20 03 :<07>
>1ed0 f5 a9 9a 8d 7b 06 4c 7b :<e5>
>1ed8 f5 a2 4b 2c a2 18 8e 85 :<99>
>1ee0 06 a2 88 8e 86 06 4c 7d :<36>
>1ee8 06 08 20 be f5 28 d0 28 :<90>
>1ef0 a2 01 a0 10 86 2b 84 2c :<1b>
>1ef8 a0 09 a9 00 99 00 10 88 :<39>
>1f00 10 fa 85 75 8c eb 06 a2 :<c9>
>1f08 06 bd 90 f5 95 32 ca d0 :<82>
>1f10 f8 f0 39 08 20 be f5 28 :<47>
>1f18 d0 58 a2 01 a0 40 86 2b :<5b>
>1f20 84 2c ac 8a f5 ae 89 f5 :<20>
>1f28 d0 01 88 ca 86 37 84 38 :<bd>
>1f30 a0 09 a9 00 99 00 40 91 :<b6>
>1f38 37 88 10 f8 8d eb 06 84 :<cf>
>1f40 75 a2 0d bd 89 f5 9d dd :<7f>
>1f48 06 ca 10 f7 a9 7b 8d 7b :<1c>
>1f50 06 a9 8a 8d 7c 06 a9 8b :<df>
>1f58 48 a9 db 48 4c 73 06 01 :<2d>
>1f60 94 03 94 03 94 03 94 00 :<6e>
>1f68 e8 fe e7 00 e8 08 20 be :<d5>
>1f70 f5 28 d0 56 a2 10 2c eb :<a3>
>1f78 06 30 1f a2 0d b5 2b bc :<8f>
>1f80 dd 06 9d dd 06 98 95 2b :<74>
>1f88 ca 10 f2 ad eb 06 49 01 :<51>
>1f90 8d eb 06 60 24 81 10 fb :<71>
>1f98 a2 22 4c 0c f7 aa a5 15 :<40>
>1fa0 48 a5 14 48 8a c9 28 d0 :<29>
>1fa8 21 20 26 f7 20 f7 f5 20 :<0f>
>1fb0 29 f7 c9 29 d0 14 a0 00 :<ab>
>1fb8 20 93 06 aa 20 26 f7 8a :<0c>
>1fc0 a8 68 85 14 68 85 15 4c :<5c>
>1fc8 08 f6 4c 47 fa a2 e1 8e :<63>
>1fd0 85 06 a2 9d 8e 86 06 4c :<16>
>1fd8 7d 06 ac eb 06 c8 a9 00 :<23>
>1fe0 85 0e a2 71 8e 85 06 a2 :<5b>
>1fe8 94 8e 86 06 4c 7d 06 a2 :<ad>
>1ff0 0e 4c 0c f7 20 4c f6 e0 :<07>
>1ff8 19 b0 f4 86 76 c9 2c d0 :<4c>
>2000 42 20 49 f6 e0 28 b0 e7 :<f9>
>2008 20 29 f7 d0 36 8a a8 a6 :<9e>
>2010 76 18 a9 39 8d 85 06 a9 :<8c>
>2018 d8 8d 86 06 4c 7d 06 a9 :<d3>
>2020 81 2c a9 84 8d 85 06 a9 :<37>
>2028 9d 8d 86 06 4c 7d 06 d0 :<2c>
>2030 12 ad 06 ff 09 10 d0 07 :<46>
>2038 d0 09 ad 06 ff 29 ef 8d :<f1>
>2040 06 ff 60 4c 47 fa 48 2c :<20>
>2048 eb 06 30 0f c9 9c d0 08 :<f9>
>2050 a2 00 20 26 f7 a9 d8 2c :<a5>
>2058 a9 d1 2c a9 c3 8d 85 06 :<33>
>2060 a9 c5 8d 86 06 68 4c 7d :<30>
>2068 06 68 c9 12 d0 08 a8 68 :<17>
>2070 c9 8e f0 0a 48 98 48 a0 :<f0>
>2078 00 b1 3b 4c 81 04 a0 00 :<6f>
>2080 b1 3b c9 3a b0 15 c9 20 :<1b>
>2088 f0 08 38 e9 30 38 e9 d0 :<85>
>2090 b0 09 a9 8e 48 a9 12 48 :<68>
>2098 18 90 dc 20 29 f7 d0 0d :<e4>
>20a0 a2 8b a0 06 8e 7b 06 8c :<5a>
>20a8 7c 06 4c 73 06 2c eb 02 :<67>
>20b0 10 09 24 81 10 05 48 20 :<fb>
>20b8 54 f7 68 a2 3f a0 8c 8e :<c4>
>20c0 7b 06 8c 7c 06 4c 73 06 :<c4>
>20c8 a9 5b 8d 85 06 a9 a4 8d :<ab>
>20d0 86 06 4c 7d 06 a9 a8 8d :<4a>
>20d8 85 06 a9 d8 8d 86 06 d0 :<a7>
>20e0 18 aa a9 83 8d 7b 06 a9 :<03>
>20e8 86 8d 7c 06 4c 73 06 a9 :<b8>
>20f0 b0 8d 85 06 a9 9b 8d 86 :<b5>
>20f8 06 4c 7d 06 a9 73 2c a9 :<aa>
>2100 79 8d 85 06 a9 04 8d 86 :<68>
>2108 06 4c 7d 06 48 a9 3d 8d :<75>
>2110 85 06 a9 8a 8d 86 06 68 :<fc>
>2118 4c 7d 06 48 a9 b2 8d 85 :<51>
>2120 06 a9 90 8d 86 06 68 4c :<4c>
>2128 7d 06 a6 ca 86 76 a6 cd :<ee>
>2130 86 77 ae e7 07 ac e6 07 :<07>
>2138 86 ca 84 cd 20 ff f6 a9 :<59>
>2140 12 20 b9 f7 a9 5b 20 45 :<2f>
>2148 f7 20 f2 f6 a9 5d 20 45 :<1f>
>2150 f7 a9 92 20 b9 f7 a2 04 :<68>
>2158 ca 8a 48 a9 20 20 b9 f7 :<fb>
>2160 68 aa d0 f4 a6 76 86 ca :<2d>
>2168 a6 77 86 cd 20 ff f6 ad :<b9>
>2170 43 05 a2 ff c9 02 f0 07 :<2c>
>2178 29 01 f0 07 a2 f5 2c a2 :<49>
>2180 fe a0 00 8a e8 d0 fd c8 :<69>
>2188 d0 fa aa e8 d0 f5 60 a6 :<6a>
>2190 99 e0 03 d0 03 20 45 f7 :<e6>
```

```
>2198 60 a6 ef c9 00 d0 0d a2 :<17>
>21a0 12 a0 87 8e 7b 06 8c 7c :<5d>
>21a8 06 4c 73 06 18 a5 73 65 :<0e>
>21b0 14 85 14 a5 74 65 15 85 :<10>
>21b8 15 e0 00 f0 e2 ca bd 27 :<02>
>21c0 05 c9 1d d0 05 a9 20 9d :<68>
>21c8 27 05 20 36 f7 90 d0 a4 :<79>
>21d0 ef b9 27 05 99 28 05 88 :<7b>
>21d8 10 f7 a9 12 8d 27 05 a6 :<c9>
>21e0 ef a9 92 9d 27 05 e8 a9 :<b3>
>21e8 20 9d 27 05 e8 86 ef 18 :<09>
>21f0 90 ad 20 26 f7 c9 fe f0 :<1b>
>21f8 15 a5 3b d0 02 c6 3c c6 :<04>
>2200 3b a2 17 a0 94 8e 7b 06 :<b4>
>2208 8c 7c 06 4c 73 06 20 26 :<32>
>2210 f7 a0 3f 88 30 e3 d9 5c :<75>
>2218 f8 d0 f8 a9 f8 48 a9 58 :<f7>
>2220 48 98 0a a8 b9 9c f8 48 :<01>
>2228 b9 9b f8 48 4c 26 f7 4c :<67>
>2230 79 04 8f 89 85 9c 9d 00 :<c5>
>2238 00 00 00 00 00 00 00 00 :<92>
>2240 00 00 00 00 00 00 00 00 :<a2>
>2248 00 00 00 00 00 00 00 00 :<b2>
>2250 00 00 00 00 00 00 00 00 :<c2>
>2258 00 00 00 00 00 00 00 00 :<d2>
>2260 00 00 00 00 00 00 00 00 :<e2>
>2268 00 00 00 00 00 00 00 00 :<f2>
>2270 00 d1 fc 03 f6 c6 f5 7d :<00>
>2278 e8 22 e8 00 00 00 00 00 :<dd>
>2280 00 00 00 00 00 00 00 00 :<23>
>2288 00 00 00 00 00 00 00 00 :<33>
>2290 00 00 00 00 00 00 00 00 :<43>
>2298 00 00 00 00 00 00 00 00 :<53>
>22a0 00 00 00 00 00 00 00 00 :<63>
>22a8 00 00 00 00 00 00 00 00 :<73>
>22b0 00 00 00 00 00 00 00 00 :<83>
>22b8 00 00 00 00 00 00 00 00 :<93>
>22c0 00 00 00 00 00 00 00 00 :<a3>
>22c8 00 00 00 00 00 00 00 00 :<b3>
>22d0 00 00 00 00 00 00 00 00 :<c3>
>22d8 00 00 00 00 00 00 00 00 :<d3>
>22e0 00 00 00 00 00 00 00 00 :<e3>
>22e8 00 00 00 00 00 00 00 00 :<f3>
>22f0 e7 f4 f6 f2 3d f3 ae f3 :<d2>
>22f8 df f2 46 fa f8 f3 3c f5 :<26>
>2300 96 f5 46 fa 12 f5 ff ea :<fc>
>2308 61 f6 58 f6 1d f6 46 fa :<70>
>2310 53 f1 46 fa 46 fa 19 f1 :<83>
>2318 43 f0 6f f6 46 fa 01 ea :<b8>
>2320 3a ea 62 ea a5 e9 a7 e8 :<73>
>2328 46 fa 46 fa 0c e8 00 00 :<54>
>2330 00 00 00 00 00 00 00 00 :<83>
>2338 00 00 00 00 00 00 00 00 :<93>
>2340 00 00 00 00 00 00 00 00 :<a3>
>2348 00 00 00 00 00 00 00 00 :<b3>
>2350 00 00 00 00 00 00 00 00 :<c3>
>2358 00 00 00 00 00 00 00 00 :<d3>
>2360 00 00 00 00 00 00 00 00 :<e3>
>2368 00 00 00 00 00 00 00 00 :<f3>
>2370 00 00 00 00 00 00 00 00 :<03>
>2378 00 00 00 00 00 00 00 00 :<13>
>2380 00 00 00 00 00 00 00 00 :<24>
>2388 00 00 00 00 00 00 00 00 :<34>
>2390 00 00 00 00 00 00 00 00 :<44>
>2398 00 00 00 00 00 00 00 00 :<54>
>23a0 00 00 00 00 00 00 00 00 :<64>
>23a8 00 00 00 00 00 00 00 00 :<74>
>23b0 00 00 00 00 00 00 00 00 :<84>
>23b8 00 00 00 00 00 00 00 00 :<94>
>23c0 00 00 00 00 00 00 00 00 :<a4>
>23c8 00 00 00 00 00 00 00 00 :<b4>
>23d0 00 00 00 00 00 00 00 00 :<c4>
>23d8 00 00 00 00 00 00 00 00 :<d4>
>23e0 00 00 00 00 00 00 00 00 :<e4>
>23e8 00 00 00 00 00 00 00 00 :<f4>
>23f0 c9 80 90 23 a0 f9 29 7f :<44>
>23f8 0a 18 69 1a 90 01 c8 85 :<8e>
>2400 22 84 23 a9 fa 48 a9 3e :<4f>
>2408 48 a0 01 b1 22 48 88 b1 :<f2>
>2410 22 48 4c 26 f7 18 24 38 :<f0>
>2418 b0 03 4c 79 04 a2 0b 4c :<64>
>2420 0c f7 aa a9 00 85 22 a9 :<9f>
>2428 fb 85 23 a0 00 ca 10 0f :<d0>
>2430 b1 22 48 e6 22 d0 02 e6 :<d6>
>2438 23 68 10 f4 30 ef c8 b1 :<07>
>2440 22 30 05 20 45 f7 d0 f6 :<5c>
>2448 a0 4b 8c 7b 06 a0 8b 8c :<34>
>2450 7c 06 4c 73 06 48 20 c6 :<6d>
>2458 fa 90 17 c0 01 f0 17 aa :<f0>
>2460 68 8a 18 08 a2 6a 8e 7b :<18>
>2468 06 a2 89 8e 7c 06 28 4c :<09>
>2470 73 06 68 38 b0 ed 48 a0 :<e9>
>2478 ff c8 b9 00 02 d0 fa c0 :<82>
>2480 58 d0 03 4c 1a ef 98 aa :<5e>
>2488 e8 e8 c8 88 ca b9 00 02 :<05>
>2490 9d 00 02 c4 3b d0 f4 68 :<e7>
>2498 a0 02 d0 c3 a9 00 85 22 :<0f>
>24a0 a9 fb 85 23 a0 00 84 0b :<8f>
>24a8 88 c8 b1 3b 38 f1 22 f0 :<45>
>24b0 f8 c9 80 f0 1b b1 22 30 :<19>
>24b8 03 c8 d0 f9 c8 e6 0b 18 :<47>
>24c0 98 65 22 85 22 90 02 e6 :<3a>
>24c8 23 18 a0 00 b1 22 d0 da :<b7>
>24d0 05 0b 85 0b c8 60 4f 4c :<b1>
>24d8 c4 4d 45 52 47 c5 44 52 :<df>
>24e0 55 ce 44 56 45 52 49 46 :<0d>
>24e8 d9 44 4d 45 52 47 c5 46 :<90>
>24f0 45 54 43 c8 4c 49 53 54 :<de>
>24f8 a3 53 50 4c 49 d4 43 48 :<c2>
>2500 41 4e 47 c5 50 41 52 d4 :<e5>
>2508 4e 4f 53 50 4c 49 d4 53 :<6c>
>2510 45 41 52 43 c8 46 41 53 :<8d>
>2518 d4 53 4c 4f d7 53 45 d4 :<4c>
>2520 56 45 52 53 49 4f 4e a4 :<c9>
>2528 47 45 4e 44 41 d4 53 54 :<9d>
>2530 41 52 d4 49 4e c3 50 52 :<55>
```

```
BASIC-ERWEITERUNG

>2538 4f 43 45 4e c4 50 52 4f :<2c>
>2540 43 45 44 55 52 c5 47 52 :<ff>
>2548 41 50 48 49 c3 52 45 50 :<ee>
>2550 4c 41 43 c5 50 4f d0 49 :<2a>
>2558 4e 49 54 4b 45 d9 52 45 :<c6>
>2560 43 4f 52 44 a3 45 58 43 :<24>
>2568 48 41 4e 47 c5 41 53 53 :<a4>
>2570 4f 52 d4 42 49 54 a4 42 :<7d>
>2578 44 45 c3 52 45 53 45 d4 :<b3>
>2580 00 00 00 00 00 00 00 00 :<26>
>2588 00 00 00 00 00 00 00 00 :<36>
>2590 00 00 00 00 00 00 00 00 :<46>
>2598 00 00 00 00 00 00 00 00 :<56>
>25a0 00 00 00 00 00 00 00 00 :<66>
>25a8 00 00 00 00 00 00 00 00 :<76>
>25b0 00 00 00 00 00 00 00 00 :<86>
>25b8 00 00 00 00 00 00 00 00 :<96>
>25c0 00 00 00 00 00 00 00 00 :<a6>
>25c8 00 00 00 00 00 00 00 00 :<b6>
>25d0 00 00 00 00 00 00 00 00 :<c6>
>25d8 00 00 00 00 00 00 00 00 :<d6>
>25e0 00 00 00 00 00 00 00 00 :<e6>
>25e8 00 00 00 00 00 00 00 00 :<f6>
>25f0 00 00 00 00 00 00 00 00 :<06>
>25f8 00 00 00 00 00 00 00 00 :<16>
>2600 00 00 00 00 00 00 00 00 :<26>
>2608 00 00 00 00 00 00 00 00 :<36>
>2610 00 00 00 00 00 00 00 00 :<46>
>2618 00 00 00 00 00 00 00 00 :<56>
>2620 00 00 00 00 00 00 00 00 :<66>
>2628 00 00 00 00 00 00 00 00 :<76>
>2630 00 00 00 00 00 00 00 00 :<86>
>2638 00 00 00 00 00 00 00 00 :<96>
>2640 00 00 00 00 00 00 00 00 :<a6>
>2648 00 00 00 00 00 00 00 00 :<b6>
>2650 00 00 00 00 00 00 00 00 :<c6>
>2658 00 00 00 00 00 00 00 00 :<d6>
>2660 00 00 00 00 00 00 00 00 :<e6>
>2668 00 00 00 00 00 00 00 00 :<f6>
>2670 00 00 00 00 00 00 00 00 :<06>
>2678 00 00 00 00 00 00 00 00 :<16>
>2680 00 00 00 00 00 00 00 00 :<27>
>2688 00 00 00 00 00 00 00 00 :<37>
>2690 00 00 00 00 00 00 00 00 :<47>
>2698 00 00 00 00 00 00 00 00 :<57>
>26a0 00 00 00 a9 9d 4c fd f4 :<39>
>26a8 a2 1e 86 61 a2 e2 86 62 :<e7>
>26b0 a2 fc 86 63 20 19 f7 60 :<69>
>26b8 43 20 42 59 20 53 54 45 :<d2>
>26c0 50 48 41 4e 20 42 49 45 :<16>
>26c8 4c 49 43 4b 45 20 56 31 :<7a>
>26d0 35 30 34 38 38 2e 20 20 :<30>
==================================
P R O G R A M M E N D E
==================================
```

23 neue Befehle

Fortsetzung von Seite 120

Für A und B gilt der Bereich Null bis 65535, für C und D gilt Null bis 63999.
Es werden in jeder Datazeile 22 zweiziffrige Hex-Zahlen geschrieben. In der letzten Zeile können auch weniger stehen. Wurde INC angegeben, so folgt noch eine vierziffrige Hex-Zahl, die sich als Summe der 22 Werte ergibt.
Beispiele:
GENDAT0TO255
legt die ersten 256 Byte ab Zeile 10 mit Schrittweite 10 als BASIC-Datas ab.
GENDATDEC("2000")TODEC("3FFF")START1000STEP1INC
legt ein Grafikbild (Bereich von hex2000 bis hex3fff) ab Zeile 1000 mit Schrittweite 1 und Prüfsumme im BASIC-Programm ab.
Geben Sie folgende Sequenz zum besseren Verstehen ein:

```
NEW
GENDAT0TO24START12STEP3
LIST
```

Jetzt erscheint:

```
12 DATA00,00,00,00,
   00,00,00,00,00,00,
   00,00,00,00,00,00,
   00,00,00,00,00,00
15 DATA00,00,00
```

Allerdings werden nicht überall nur Nullen stehen. Geben Sie dagegen GENDAT0TO24INC ein, so erhalten Sie:

```
10 DATA00,00,00,00,
   00,00,00,00,00,00,
   00,00,00,00,00,00,
   00,00,00,00,00,00,
   0000
20 DATA00,00,00,0000
```

Lesen Sie den ersten Fall im Programm ein mit:

```
10 RESTORE12:FORI
   =0TO24:READA$
   :POKEI,DEC(A$)
   :NEXT
```

Im zweiten Fall ist es etwas komplizierter:

```
3 RESTORE10:FORI
  =0TO24STEP22:b
  =0:FORJ=0TO21
6 IFI+J>24THEN30
  :ELSEREADA:B
  =B+DEC(A$):POKEI,
  DEC(A$)
9 NEXTJ:READA$:IFB
  <>DEC(A$)THEN
  PRINT"PRUEFSUM
  MENFEHLER":END
  :ELSENEXTI
10 DATA...
20 DATA...
```

30 hier geht's dann weiter.
Zu beachten ist, daß bereits existierende Zeilennummern ohne Warnung überschrieben werden.

PROCEDURE

PROCEDURE lenkt die Eingabe auf eine Datei eines externen Gerätes. Die Syntax ist PROCEDURE"dateiname",G ; wobei G die Geräteadresse ist (Default ist 8).
Vorsicht: Wird von einer Floppy eingelesen, so darf während der Abarbeitung kein CLOSE auf den Fehlerkanal auftreten, da sonst die Eingabedatei mit geschlossen wird (siehe Floppy-Handbuch).

PROCEND

PROCEND beendet eine mit PROCEDURE aufgerufene Datei.

POP

POP dient dazu, eine Rücksprungadresse von GOSUB vom Stack zu nehmen, falls man ein Unterprogramm nicht mit RETURN verlassen möchte.

INITKEY

INITKEY stellt die Standard-Funktionstastenbelegung her.

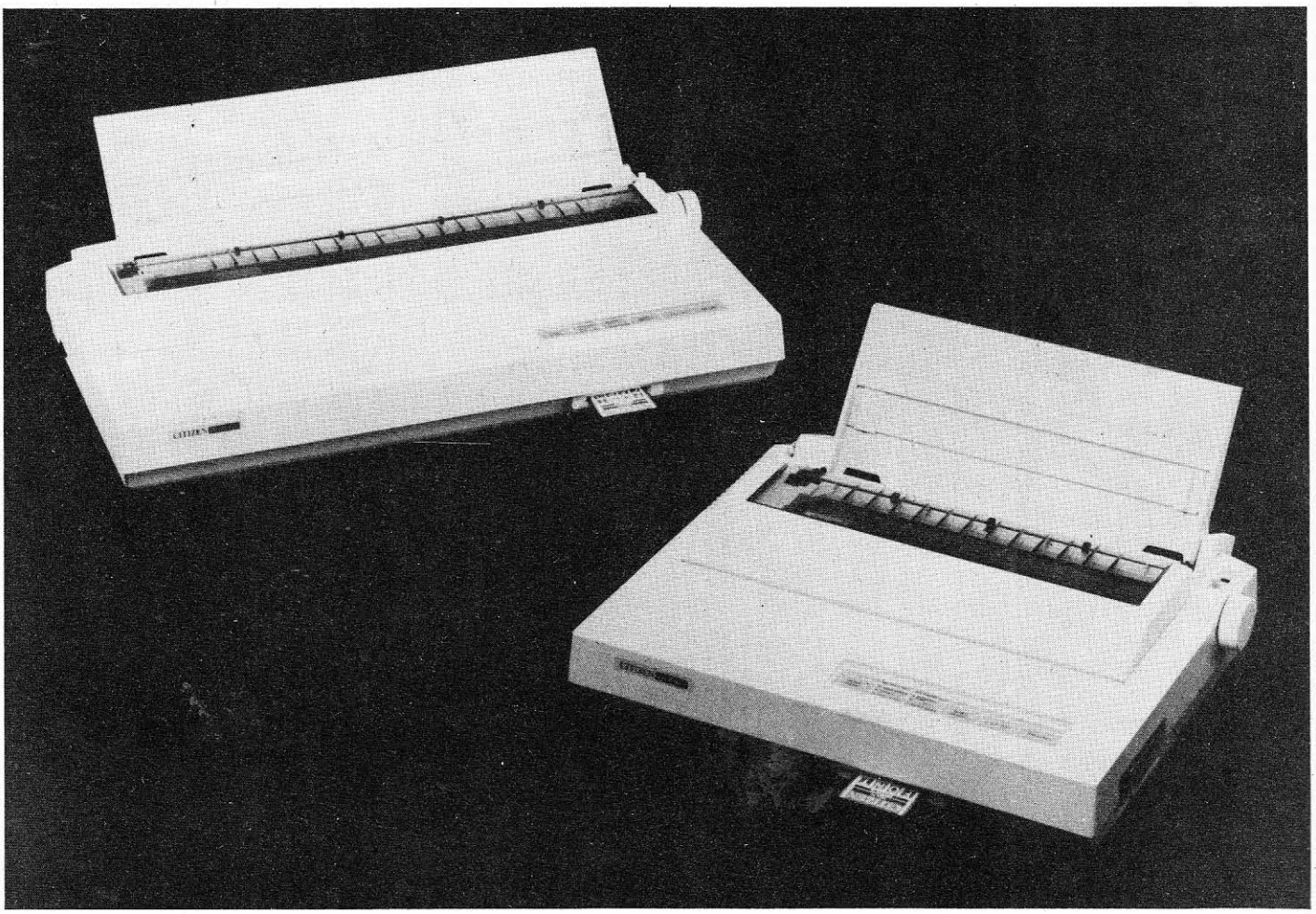
RECORD#

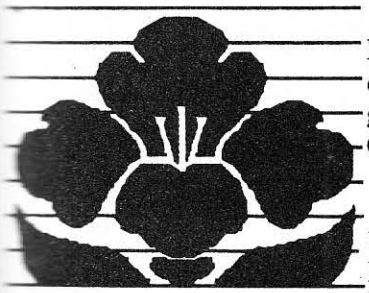
RECORD# A,B,C positioniert den Satzpointer einer relativen Datei mit der Filenummer A auf den Satz mit der Nummer B auf das C-te Zeichen. Dieser Befehl ersetzt somit den Befehl PRINT#15,"p"+CHR$(96+filenummer)+CHR$(lbsatz)+CHR$(hbsatz)+CHR$(byte). □

Schnell und flach

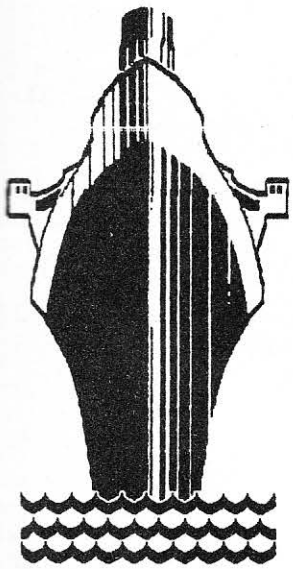
Mit der neuen Druckerserie MSP-50/55 will Citizen den Angebotsbereich der professionellen Bürodrucker verstärken. Beide Modelle unterscheiden sich lediglich durch die Verarbeitungsbreite von 80 beziehungsweise 132 Zeichen pro Zeile und durch den Anschaffungspreis.

Citizen-like ist die Technik in einem superflachen Gehäuse mit kaum zehn Zentimetern Höhe verpackt. Die standardmäßige parallele Schnittstelle wird rechts herausgeführt. Dadurch ist ein Konflikt zwischen Endlospapier und Anschlußkabel ausgeschlossen.

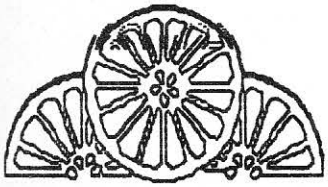




Gleich unterhalb der Schnittstelle finden sich, frei zugänglich, die beiden DIP-Schalter-Blöcke zur Einstellung der Standard-Parameter und des nationalen Zeichensatzes.

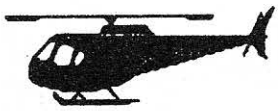


Über die Folientasten des Bedienungspanels können die Schriftarten Draft, NLQ, Highspeed und PS angewählt werden. Ferner kann in den Quiet-Modus umgeschaltet werden. Eine weitere Option gestattet es, zusätzliche Schriftarten von handlichen IC-Cards nachzuladen.



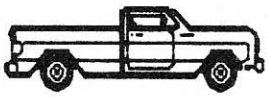
Ein bidirektionaler Formulartraktor und ein Single Sheet Feeder gehören ebenfalls zur Grundausstattung.

Für den Papiereinzug bieten die neuen Citizen-Drucker mehrere Möglichkeiten zur Auswahl. Der Traktor wird durch einen Handgriff vom Schub- zum Zugtraktor umfunktioniert. Hierbei



wird das Endlosformular von der Rückseite zugeführt. Ein Schlitz in der Unterseite des Gehäuses signalisiert eine weitere Einzugsmöglichkeit.

Unter Verwendung des Einzelblatteinzugs können die MSP-50/55 auch Briefbögen verarbeiten.

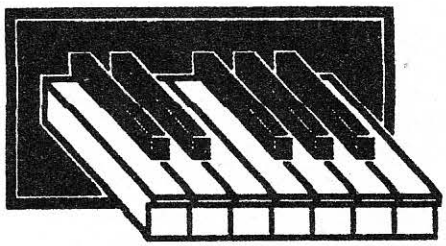


Das Einspannen der Blätter entfällt völlig. Ein Sensor erkennt das neue Blatt und löst einen Mechanismus aus, der es automatisch einzieht. Durch die Papierführungs-

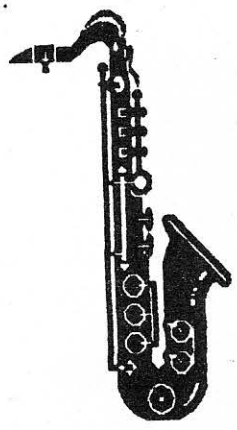
schienen wird das Blatt meistens korrekt eingezogen und ein Zoll unterhalb der Oberkante zentriert. Ein Nachzentrieren zählt zu den Ausnahmefällen.

Leider besitzt der Single Sheet Feeder einen kleinen Nachteil: Er muß auf den Traktor aufgesteckt werden. Dadurch

arbeitet der Traktor auch im Leerlauf weiter, obwohl er für die Einzelblattverarbeitung nicht benötigt wird. Die Citizen-Drucker sehen keine Möglichkeit vor, sie für diese Zeit auf Eis zu legen. Im Leerlauf ist das monotone Tackern des Traktors eine störende Geräuschkulisse.

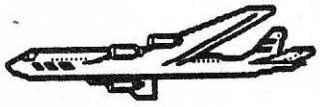


Der Druckkopf ist mit neun Nadeln bestückt. Er bringt Zeichen in Draft Quality mit einer Auflösung von 9x9 (IBM Grafikzeichen 12x12) und in Near Letter Quality mit 17x17 Punkten zu Papier.



Schon alleine die Bezeichnung NLQ verdeutlicht, daß das Schriftbild nicht die optimalen Möglichkeiten eines Matrixdruckers erreicht. Der unpersönliche Computerausdruck ist auf Anhieb zu erkennen und dient nicht gerade als Reklameschild (ein allgemeines Problem der Neun-Nadel Technologie).

Im Gegensatz dazu hat Citizen den empfohlenen Preis von 1898 Mark für den MSP-50 (80 Spalten) jedoch um mindestens hundert Mark höher angesetzt, als qualitativ hohe 24-Nadel-Drucker anderer Hersteller dem Anwender abverlangen. Der MSP-55 wird sogar mit rund 2400 Mark angeboten. Nach unserer Ansicht hat Citizen hier danebengegriffen.

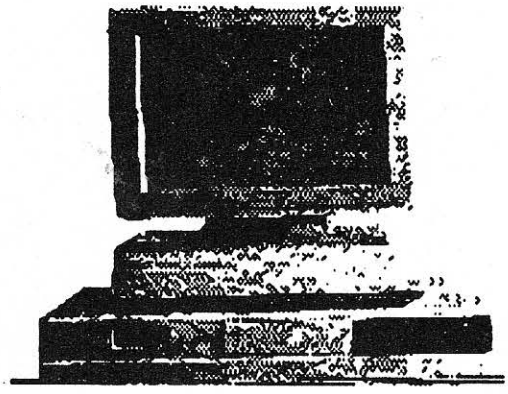


Die mit dem MSP-50/55 erstellten Grafiken im Bit-Image-Mode erreichen eine Auflösung bis zu 240 Punkten pro Inch.

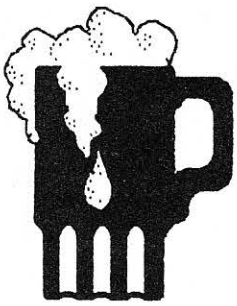
Durch eine optimale Erweiterung können die MSP-50/55 Texte und Grafiken sogar farbig zu Papier bringen. Diese Möglichkeit stand uns im Test leider nicht zur Verfügung.

Als Zwischenspeicher ist ein Acht-KByte-RAM eingebaut. Anpassungsprobleme gibt es in keinem professionellen Software-Paket. Die MSP-50/55 können als Epson EX/FX oder als IBM-Prowriter installiert werden. Einer dieser Treiber stellt Ihnen Ihr Programm garantiert zur Verfügung.

Die Geräuschentwicklung während des Druckvorgangs wurde von Citizen mit weniger als 55 dB (A) angegeben. Dieser Wert weicht ganz erheblich von unseren Testergebnissen ab.



Tatsächlich gehört die Citizen-Baureihe MSP-50/55 zu den geräuschintensivsten Nadel-Druckern, die jemals in unserer Testredaktion standen. Selbst der viel gepriesene Quiet-Modus schafft es bei halbierter Geschwindigkeit nicht, die Lautstärke auf weniger als 55 dB(A) zu reduzieren. Der leerlaufende Traktor bei der Einzelblattverarbeitung trägt noch seinen Teil zur Geräuschkulisse bei.



Bei den Angaben zur Geschwindigkeit müssen die Herstellerangaben korrigiert werden. Sie beziehen sich auf eine einzige Zeile ohne Berücksichtigung der Zeilenvorschubzeit. Erst die Berechnung des Durchschnitts einer DIN A-4-Seite ergibt genaue Werte der Geschwindigkeit.

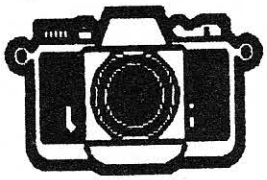
Unsere Testwerte und die Angaben des Herstellers im Vergleich:

Druckart	Tatsächliche Geschwindigkeit	Angaben des Herstellers
Draft/HS	170,3 cps	300 cps
Draft	155,3 cps	250 cps
NLQ/HS	45,9 cps	60 cps
NLQ	39,7 cps	50 cps

Die Angaben HS (Highspeed) beziehen sich auf die Schriftart Elite mit 12 cpi.

Fazit

Die Citizen-Modelle MSP-50/55 haben interessante Features zu bieten. Über die Qualität der Ausdrucke gibt es nichts Negatives zu berichten –



dies gilt aber nur für den Bereich der Neun-Nadel-Technik. Hier kann Citizen allemal standhalten.

Die Möglichkeiten der MSP-50/55 -Drucker, ihr Handling und die Geschwindigkeit halten mit vergleichbaren Modellen durchaus Schritt. Von der Lautstärke her übernimmt Citizen sogar die Spitzenposition dieser Technologie.

Die Preise sind höher als die einiger weitverbreiteter 24-Nadel-Drukker, die mit einer Auflösung bis zu 360 Dots per Inch auftrumpfen können. Hier sollte Citizen seine Preispolitik gründlich überdenken. ■

PLUS4! Ich suche Amateurfunkprogramme f. Plus4' Angebote an Rudolf Halmschlag, Dorfstr. 29E, 2060 Travenbrück

---

Wer will CBS-Telespiel m. 7 Spielen. Tausche auch gg. Drucker f. C16 (funktionsf.) Ruf mich mal an: 06103/ 62341, Stefan.

C16/P4-USER: Orig.-Spielesoftware auf Kass. spottbillig. Restliste gg. 80-Pf.-Freiumschlag. Harald Scheel, Kollaustr. 178, 2000 Hamburg 61

C16/116(64K)/PLUS4! Relat. Dateiverw.: bis zu 1.500 Datensätze! Freie Maskenerstellung in 10 Feldern a 24 Zeichen! Mit Handbuch! Preis 19,50 DM plus NN/Porto. Jürgen Vorwerk, Dörener Weg 29 a, 4790 Padernborn, Tel. 05251/58423

---

*PLUS4 * C16 * C116 * Auf Disk! Suche Tauschpartner zwecks Prog.- u. Erf.-Austausch. Von einem Partner erwarte icn Pünktlichkeit u. in erster Linie Zuverlässigkeit. Listen an: Arne Woll, Am Stübenhaus, 2900 Oldenburg

---

VERKAUFE Drucker Brother HRS m. Farbbändern u. Interface 92008G m. stabilisiertem Netzteil f. 320,– DM. Orig.-Verpack., Zustand hervorragend. Harald Kesper, Uplandstr. 16, 3542 Willingen, Tel. 05632/69747 nach 18 h

---

VERKAUFE: C16/64K, Floppy 1551, Datas., Userport, R6522 VIAPORT, Eprommer, Software, Lit. (140 Zeitschr. u. 2 Bücher). NP 2.100,– DM, VB 900,– DM. Info bei: Mike Wieland, Danziger Str. 7, 4430 Steinfurt, Tel. 02551/80907

Logo f. den Plus4, C16(64K): Modul, Demodisk u. Handbuch, nur DM 50,–. Eprom-Platine gesockelt DM 20,–. Div. Anwend.-Software auf Eprom. Info gg. Freiumschlag (1,10 DM). Harals Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390

SUCHE DRUCKER Citizen IDP 560 u. anderen Drucker f. Plus4, z.B. MPS803. Angebote an: Paul Kasparbauer, Burgweg 1, 8376 Altenberg

---

SUCHE TAUSCHPARTNER f. Plus4-Games – nur auf Kass. L. Vilikus, Umeletzka 11, 170 00 Prag 7, CSSR

VERKAUFE C16(64) – u. Plus4-Spiele. Suche Tauschpartner aus aller Welt f. Plus4-Software. Verk. Spiele billig, nur Tape! Tel. 04752/ 228, Jürgen Oschinger, Cadenberger Weg 25, 2171 Geversdorf. P.S. Suche Kontakt zu Atari-XL800-Besitzern!

---

25 Games f. den C16 stehen zur Auswahl, pro Stück 1,– bis 5,– DM, Paintbox f. 12,– DM. Liste anfordern bei S. Plasa, Schallmauerweg 25, 5030 Huerth! Nur m. entsprechendem Rückporto!

ACHTUNG, C16/PLUS4-USER! Wir suchen noch Mitglieder f. den ICUC im In- u. Ausland! Kostenlose Infos bei: ICUC, Halweg 77, 4320 Hattingen. Wir bieten viel: Software, Clubzeitung, Kontakt zu andern Mitgliedern, und, und, und!

DDR-Bürger sucht Commodore +4 od. C64, Datasette, evtl. Floppy, u. Joystick preisgünstig, kostenlos od. Wertausgleich m. antiquar. Büchern. Gert Gebauer, Zieglertal 43, DDR-8704 Cunewalde

---

HILFE! HILFE! Suche dringend f. Plus4 Drucker u. Floppy! Biete pro Gerät 200,– DM! Bitte melden bei: Markus Fingscheidt, Niederweniger Str. 324 A, 4300 Essen 15, od. Tel. 0201/ 484592, wochentags ab 17 Uhr. Es eilt!

FÜR PLUS4! Verk. 2 doppelseit. bespielte Disketten f. nur DM 20,–. Wählt selber! Spiele m.u.o. Joystick, CAD-, Kopier- od. Anwend.-Prog. Michael Schuster, Norderwungweg 1, 2250 Husum

---

ACHTUNG! Ich suche Computerclub im Raum Hagen zwecks Software- u. Erf.-Austausch. Auch an privaten Kontakten bin ich interessiert. Meldet Euch zahlreich bei Henning Ritter, Berghofstr. 22, 5800 Hagen, Tel. 02331/ 83140

Endlich: EPROMBRENNER für Plus/4

Für den Plus/4 gibt es endlich einen Eprombrenner!
Er brennt Eproms bis ‚27256'. Für die Verwaltung der Eproms gibt es eine Eprombank 192 KB aufrüstbar auf 384 KB; alles schaltbar.
INFO gegen 0,80 Pfennigbriefmarke (Freiumschlag)
Jürgen Braunroth, Moordorferstr. 30, 3057 Neustadt 1

CIG nimmt noch Mitglieder auf. Sofort Superinfo anfordern! Clubzeitung, Infos zu Script, Freesoft. Alles kostenlos. Außerdem Softwarebibliothek m. über 400 Prog. Info gg. 1,60 DM. CIG, c/o K.-D. Schindler, Luciusstr. 10 A, 6230 Frankfurt 80

C16/116/PLUS4-SOFTWARE. Die besten 64KB-Prog. auf einer zweiseit. Disk. Von Top-Kopierprog. bis Strategie- u. Ballerspielen. Also gleich bestellen: 25,– DM Vorkasse an: Robert Waltereit, Donaustr. 51, 2800 Bremen 1

PLUS4/C16/C116: Biete 100 Spiele u. 10. Anwend.-Prog. f. 20,– DM (Disk). Schein od. Scheck. Suche Kopierprog. v. Tape to Disk. Listing gg. 1,– DM, 4 Diskseiten o. Porto. Rainer Pytlik, Lettenallee 69, 1000 Berlin 51

Milchquotengeplagter Provinzbauer sucht intakte Floppy u./od. Drucker. System P4. Max. 100,– DM pro Gerät. Angebote bitte an Georg Bohlen, Untenende Ost 33, 2912 Nordgeorgsfehn

---

*** MPS 803 *** Commodore Drucker m. Traktor, Handbuch, orig.-verpackt, VB 320,– DM. Suche Anwend.-Software u. Bombjack C16. Tel. tagsüber 08141/60290, J. Lachenmeyer, Augsburger Str. 23, 8950 Kaufbeuern

---

P4-BESITZER sucht Tauschpartner(in), Disk, evtl. auch Kass. Verk. Karate King (64KB) u. Strip Poker (64KB) a 10,– DM. Suche Paperboy u. Pin Point. Zahle bis zu 50 % des Neupr. Liste an: M. Schlichting, Lipperkamp 49, 2850 Bremerhaven

---

PLUS4, Datas., Floppy, BASIC-Kass., COM.PRG. Disk, div. Bücher, 2 Jg. CW u. CM, 50 Disks: 4 Textprog., Dateien usw., 100 Spiele, z.B. Ace, Summer & Winter Olymp., Pacman usw. zu verk. Festpreis DM 400,–, Neupreis ca. DM 1.000,–. 1 Jahr alt. Tel. 0591/2086, F. Hackmann

---

VERKAUFE Spiele, KLEINE Kopien, f. C16 und P4: Apollo Rescue, Skramble, Out on a Limb, je 5,– DM. Karate King, Olymp. Winterspiele 15,– DM. Sport Show, Sextett, Pluspaket II, Sommerolympiade je 22,– DM. Immer zuzügl. Porto. Disk-Locher 5,– DM. Tel. 04861/6264

PLUS4 – SUCHE Superbase, Pascal, BASIC-, C-Compiler, 80-Z-Darstellung, Superkopierprog. Anwend.-Prog. gg. Bezahlung od. Tausch. Keine Spiele. Tel. 07273/1595, T. Neumann, Marienstr. 4 A, 6729 Hagenbach

---

ACHTUNG! C16/P4. Wer hat in Mercenary den Schlüssel zur dreieckigen Tür gefunden u. schreibt mir die Lösung? Danke im voraus. A. Bischof, Magdeburgerstr. 58, 4900 Herford

---

SUCHE FÜR C16/PLUS4: d-Base +4, Textword II. Des weiteren suche ich für Bekannte in der DDR noch günstig gebr. Hard- u. Software. Listen u. Angebote an Jörg Hildebrandt, Schlenkenbrink 25, 4973 Vlotho

---

FREESOFTWARE f. C16/116/P4 u. C64. Demodisk doppelseit. nur 10,– DM. Bitte Computertyp angeben! Viktor Jürgens, Prinzenstr. 131, 2330 Eckernförde

---

VERKAUFE C+4, Floppy VC1551, Joystick, 10 Disk m. Anwend.- u. Spielprog. Bücher: Alles über den +4, Bedienungs- u. P4-Softwarehandbuch. Wg. Zeitmangel f. DM 400,– (nur zus.), alles neuwertig. Tel. 089/7591335. H. Pecha, Königswieserstr. 5, 8000 München 71

SUCHE KONTAKT zu Plus4-Usern. Erfahrungs- u. Software-Austausch. Ulrich Lauritzen, Goethestr. 24b/II, 8972 Sonthofen, Tel. 08321/ 88697

VERKAUFE VECTREX Spielkonsole u. 3 Spiele f. nur DM 75,–. Jeses weitere Modul nur DM 5,–. Bei Interesse (alles wirklich gut!) bitte melden bei: Oliver Lass, Rübenhofstr. 32, 2000 Hamburg 63, Tel. 040/599609

---

*BÜCHERPACK C16/116/+4 * Hängt Ihnen die Buchungs-, Konto-, Datei-, Sortierarbeit u. die Rechnungsschreiberei zum Hals raus? Ihr Plus4 kann Ihnen helfen! Im Büropack sind jede Menge Programme, die Ihnen die Arbeit erleichtern! Info: 02378/2766 ab 18 h

---

*** DAS POSTSPIEL: Galaktik-Starwars *** 10–15 Mitspieler pro Game kämpfen um die Vorherrschaft im Weltraum. Info von: ASTRO-DATA-CLUB, Postfach 943, 6400 Fulda

DIE TESTWELT, C16/116/Plus4-Club sucht noch viele Mitglieder! Wir bieten Zeitschrift auf Tape od. Disk. Inhalt: Tests, Programme, Berichte u.v.m.! Info, bitte m. 0,80 DM, bei: TESTWELT, Ch. Schweier, Hermannstr. 98, 4330 Mülheim 1

SUCHE TAUSCHPARTNER f. C16/Plus4 u. 80-Zeichen-Monitor f. Plus4, möglichst billig. Suche auch noch Computerschrott. Hans-Jürgen Kinateder, Salinenstr. 6, 6927 Bad Rappenau, Tel. 07264/6957

Matrixdrucker Seikosha SP-1000VC, Commodore-Interface, Traktor f. Endlospapier, Einzelblatteinzug, NLQ, Grafik, Anschluß an C116, C16, P4, C64, 9 Mon., nur 399,– DM. Tel. Mo.–Frei. 8–16 h 0521/444371, ab 17 h 05257/3906

HOBBY-PROGRAMMIERER Suche div. Software f. C16, 116, P4, C64. Angebote bitte 2 80er-Marken f. Rückporto beilegen od. anrufen bei: E.W. Wolters, Bismarckstr. 6, 2930 Varel 1, Tel. 04451/82412

NEUE MAILBOX ONLINE, rund um die Uhr, jeden Tag. TERRA-COM Tel. 06142/561528. Parameter: 300 Baud 8N1, Public-Modus m. SF-Board, Diskussionsboards, Flohmarkt, Online-Games, Systemboards, Witzecke, Rätselboard u. mehr

---

ACHTUNG! Verkaufe C16 m. 64K, Datasette, Joystick, Programmen, Zeitschr., Gesamtwert ca. 550,– DM, f. nur 200,– DM od. Tausch gg. C64 m. Handbuch od. aufgerüsteten VC20 m. Handbuch u. Netzteil. Tel. 040/6449573

SUCHE Astronomie-, Astrophysik-Programme f. Plus4 m. 1551. Ebenso Geometrie u. Trigonometrie, nur auf Disk. Johannes Rabowski, Rather Kreuzweg 65A, 4000 Düsseldorf 30, Tel. 0211/656418

C16/116/Plus4-Programme ab DM 1,–. Alle aus eigener Entwicklung. Liste gg. 50-Pf.-Briefm. f. Porto bei Uwe Ludschoweit, Hassloch am Wasserwerk, 6090 Rüsselsheim

---

C16/116/PLUS4! 25 Top-Games auf Tape f. ganze 10,– DM! Schein an: Dietmar Neumann, Trierer Str. 398, 5100 Aachen. P.S. Tausche auch Games!

C16/PLUS4-MAGAZIN-DISK m. vielen Programmen u. Anwendungen, News, Tips u. Tricks, Wissen, Pokes u. Super-Testbildgenerator. Mit Picutres. 15,– DM schicken an Uwe Ludschoweit, Hassloch am Wasserwerk, 6090 Rüsselsheim

---

PLUS4 m. 1551 Diskdrive, VIC 1525 Graphics-Printer, Monitor 12" bernstein, Joystick PR5000, Scriptplus, Turboplus (Module), Software aus CW-Sonderheften, PED 3D-Grafik, Sommerolympiade: Preis DM 550,–. Tel. 089/3104451

*** ACHTUNG! *** Wer schickt mir kostenlos seinen Computerschrott (alles am liebsten von Commodore). Ich zahle selbstverständlich das Porto. Danke! Frank Hennig, Fallingbosteler Str. 11, 3036 Bonlitz 1

HALLO, C16-FREAKS, wie wäre es m. ein paar neuen Spielen f. Eueren Commodore C16, 116 od. Plus4. Ich schicke Euch Infos gg. 80 Pf. Rückporto. Konstantin Miloradovic, Weidenweg 21, 4516 Bissendorf 1

---

15 VERSCH. HARDCOPIES in Postkartengröße gg. 5,– DM-Schein. Starke Pictures od. Picture-Disk zweiseitig m. Hardcopy-Routinen f. C16/+4 10,– DM. Bei Uwe Ludschoweit, Hassloch am Wasserwerk, 6090 Rüsselheim

---

C64/128/C16/116/P4-FANS! Verk. 60 Anwend., Utilities u. Spiele auf einer Disk od. Kass. f. nur 20,– DM. Bei Interesse Schein an: M. Greifenhagen, Stöckelstr. 8, CH-8610 Uster. System angeben!

Suche gebr. Farbmon. f. Plus4 sowie Anw.-Software, keine Spiele. Angebote an: Hans-J. Karth, Asternweg 11, 5653 Leichlingen 2

SUCHE f. meinen Plus4: Spielprog. u. ähnliches auf Kass. Wer verkauft od. tauscht welche? R. Seeholzer, Alleestr. 16, CH-8280 Kreuzlingen 1

C16/116/P4-Programme ab 50 Pf. z.B. China Horoscope, bildschirmgroße Digitaluhr, Testbildgenerator, Lottetip usw. Diskmagazine u. Picture-Disks. Liste gg. 80 Pf. Porto. Uwe Ludschoweit, Hassloch am Wasserwerk, 6090 Rüsselsheim

---

SUCHE DRUCKER MPS xxx od. anderen f. Plus4. Außerdem suche ich einen defekten C116 m. gesunder Tastatur. Eure Angebote bitte an Norbert Schmelzer, Haydnstr. 26, 4178 Kevelaer 1, Tel. 02832/2875

VERKAUFE fast neuwertige Computerzeitschr. wie z.B. COMMODORE WELT, Computronice, c't, Compute mit ASM. Auch Sonderhefte!! Till Franzmann, Tel. 06132/8168

SUCHE FÜR VC20 Speichererw. bis zu 32K. Biete als Gegenleistung bis max. 30,– DM. Angebote an Michael Herboth, Brucknerstr. 15a, 6719 Esenberg/Steinborn

---

+ PLUS4/C16/C116 + Verk. Prog.-Sammlungen I, II u. III v. Fa. Markt & Technik, Spiele u. Anwend., 20,– DM. Auch andere Orig.-Spiele je 8,– DM plus Porto. Bar od. Scheck. Tel. 0231/270660

Wer kann an eine Schülerin gebrauchte Floppy od. Drukker f. meinen C16 verschenken? Manuela Bogdan, Lunikstr. 42, DDR-Hoyerswerda

---

*** VERKAUFE PLUS4*** Mit Floppy 1551, Fachbüchern u. vielen Games. Schickt Eure Angebote an Frederick Althoff, Am Großen Holz 27a, 4800 Bielefeld 1. Oder ruft an: 0521/37565

---PLUS4, C16, C116--- Suche Partner zwecks Softwaretausch im Raum München (möglichst Disk). Liste an Stefan Schulze, Riesstr. 76, 8000 München 50, Tel. 089/145223. Gruß an Walter & Co. u. Gaby!

PLUS4! Suche zuverläss. Tauschpartner f. Software. Roland König, Markt 2, 4558 Bersenbrück, Tel. 05439/2395

SUCHE: Grundplatine f. C16 (nicht defekt). Angebote an: Michael Greifenhagen, Stökkelstr. 8, CH-8610 Uster, Tel. CH-01/9415295 zw. 19 u. 20 Uhr

---

SUCHE dt. Bedienungsanl. f. die integrierten Programme des C16/P4. Udo Lammel, Hebbelweg 3, 7146 Tamm, Tel. 07141/603571

SUCHE Kontakte zu Plus4–Usern. Erfahrungs- u. Softwaretausch. Ulrich Lauritzen, Goethestr. 24b/II, Tel. 08321/88697

*** SUPER! Floppy 1551, Mon. gelb u. Netzteil (in einem Gehäuse), dazu C16 (64K), über 100 Disk-Prog. u. Lit. f. nur 499,– DM komplett an Selbstabholer, Auch einzeln per NN. Info, Tel. 02861/2994, tägl. ab 17 Uhr bei Horst

---

SUCHE TAUSCHPARTNER u. 64-KRAM-Software f. C16, nur Tape. Angebote an: Rolf Büchner, Kemmater Str. 22, 8632 Neustadt

---

SUCHE Computerschrott aller Art. Suche Briefkontakt mit P4-Usern. Angebote u. Antworten an: Erwin Witzing, Karlstr. 17, 7910 Neu-Ulm

---

VERKAUFE Software f. C16/116/Plus4 u. VC20 f. nur 20,– DM auf Kass., randvoll m. Spielen. Schick die Scheine an: Walter Koch, Augustastr. 126, 4220 Dinslaken, und ihr bekommt die Software! Bitte Gerätetyp angeben.

ACHTUNG! C16-Besitzer! Verkaufe Bongo, Bigmac, Rockman, Xadium f. 6,– DM per Stück. Operation Hawai f. 25,– DM (nur 64K). Sebastian Kurzawa, Am Kohlenmeiler 107, 5600 Wuppertal 22

KONTAKTE C16/116/P4 Hello You there! I am looking for an English boy in order to exchange software with him! Can you help me? I only want software for the great +4! Please write to: Henning Thomas, Hauptstr. 27, 5758 Froendenberg Dellwig

---

TAUSCHE Commodore Plotter 1520 gg. neuwert. Floppy f. Plus4. Suche Software auf Disk f. C64! Bereich Gesetzgebung StGB, BGB, STPO, STVO: Straf-, bürgerl. Recht, Verkehrsrecht u. Zulassungsordnung. Ronny Peter Antholzner, Postfach 1102, 8265 Neuötting, Tel. 08671/72119

ORIGINAL-Software u. -Zubehör f. den Commodore 16/116/+4 (Kass. od. Disk)! Gratis-Katalog! Postkarte an: P. Schäfers, Riekestr. 5, 4402 Greven 1. Super-Poke-Liste f. nur 0,80 DM-Briefmarke!

EINSTEIGER (PLUS4). Suche im Großraum Passau u. München nette Leute m. fortgeschrittenen Programmierkenntnissen, an die ich mich hin u. wieder m. Fragen usw. wenden kann. Tel. 20 h bis 0 h, freitags, samstags 8 h bis 10 h: 08531/3445

KOMPLETTAUSSTATTUNG Commodore Plus4 m. Farbmon. 1802, Floppy 1551 u. Seikosha 180VC zu verkaufen wg. Modellwechsel. Tel. A-Wien-6218952, samstags. VB 8.500 ÖS.

SUCHE BILLIG ZU KAUFEN: Jede Art von C16/Plus4-Spielen, die mehr als 16KB Angebote an: Christian Wiards, Hirtenweg 10, 2110 Buchholz

---

DIASHOWS C16/C116/+4 Verkaufe Diashows f. diese Computer in 1A-Spitzenqualität! Am besten zum Vorführen! Außerdem habe ich jede Menge DIGI-Musik auf Lager! Von Falco bis Madonna! SUPER-SOUND! Info unter 02378/2766 ab 18 Uhr!

C16/64KB-USER sucht Software aller Art auf Disk! Spiele, Anwender, auch selbsterst. Prog.! Besonders gesucht: 1-2-3-CAD! Evtl. auch Tausch (Raum HN!). H. Fischer, Kirchhäuser Str. 18, 7103 Schwaigern 2. Auch gewerbl. Angebote v. Soft, Hard!

C16/PLUS4-SOFTWARE abzugeben. Sonderhefte Compute mit Nr. 3/86, 2/87, 3/87, 4/87 je Heft m. Disk 15,– DM; RS232-Bausatz f. C16 99,– DM. Jörg Hildebrandt, Schlenkenbrink 25, 4973 Vlotho-Uffeln

---

FAST GESCHENKT! Commodore VC20 m. Netzteil, Datasette u. Speichererw., 10 Orig.-Spiele, Basic-Kurs, 9 Kass. u. Buch, komplett nur DM 95,– gg. Vorkasse bei: G. Köneke, Westendstr. 26, 6072 Dreieich

---

1 Orig.-Commodore-Joystick, C16-Anschluß. Absolut neuwertig! Für nur DM 16,– (inkl. Versand). Preiter, Tel. 089/562377

VERKAUFE defekte 1531 (der Motor läuft noch!) f. nur DM 10,– excl. Porto! Bei Interesse (ist wirklich billig f. Bastler!) bitte melden bei: Oliver Lass, Rübenhofstr. 32, 2000 Hamburg 63, Tel. 040/599609

ACHTUNG! Ich suche dringend jemanden, der mir die Centronics- u. Userport-Platinen aus „CW 10 u. 11/86" ätzt bzw. besorgt!. Suche auch (gg. Portoerstattung) Hardware-Schrott aller Marken. Thomas Wolfram, Schneiderei 2, 8813 Schillingsfürst

---

C16/P4 m. Datasette! Da Ihr, nur m. Datasette, oft vernachlässigt werdet, biete ich nun meine Programme auf Kass. an. Natürlich „Turbo". Utilities, Spiele. Je Kass. (ca. 30 Prog.) DM 15,–. Karte od. Brief an: Uwe Eden, Christbusch 54, 5600 Wuppertal 2

ORIGINAL Epsoninterface f. alle Commodore m. serieller Schnittstelle incl. dt. Handbuch f. DM 70,–. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 Uhr

---

--- HILFE ! ! --- Große Hardwarenot! Wer schenkt armem Schüler aus der DDR alte Floppy od. Drucker f. Plus4? Burkhard Franke, Rottwerndorfer Str. 1/506, 8045 Dresden, DDR

VERKAUFE jede Menge Software f. C16/Plus4. Ca. 200 Freesoftprogramme ebenfalls zu verk. od. zu tauschen (nur gg. andere Free-Soft). Liste gg. 1,20 DM Porto bei: K.-D. Schindler, Luciusstr. 10A, 6230 Frankfurt 80, Tel. 069/304941

SUCHE User-Prog.-Listings f. C16 m. 64KB. Mathe, Musik, Grafik usw. Keine Spiele! Fairer Preis, P. Kruse, Thienbuettler Weg 1, 2353 Nortorf

---

VERKAUFE Spiele (50 Pf.-2,– DM). Wie? Fordert Liste gg. 80 Pf. Rückporto an: P. Wernscheidt, Birkenweg 37, 7824 Herbolzheim. Wir suchen auch Spiele f. C16/116/P4 zu kaufen od. zu tauschen. Der SPEED-CLUB sucht noch Mitglieder. Stichwort: SPEEDY.

*** C16/116/Plus4 *** Verk. 5 Anwend.-Prog. u. 20 Spiele f. nur 10,– DM Vorauskasse. Auf Disk od. Kass. Bei Tobias Bauer, Hessbergstr. 12, 7156 Wüstenrot

---

SUCHE Bücher u. Programme f. C16/64K (Disk/Tape). Auch Disks zu Sonderheften! Z.B. 1-2-3 CAD, andere CAD-Prog., Spiele usw. ROM-Listing C16. Angebote (auch gewerbl.) an: H. Fischer, Kirchhäuser Str. 18, 7103 Schwaigern 2

SUPERBASE P4 zu verkaufen. Dazu 2 Data-Becker-Bücher, Einsteiger u. Tips u. Tricks, zu Superbase. Suche Software-Tauschpartner. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 Uhr

---

VERKAUFE für C16/Plus4 Psychotests, Spiele, Grafikshows auf einer Kass. f. 10,– DM. Bestellung, Info-Anforderung (80 PF.) an: Jörg Drescher, Lütjenburger Str. 69, 2300 Kiel 14

VERKAUFE neuwert. Set: Plus4, Floppy 1551, Super-Software (M&T-Prog.-Sammlung, Kingsoft-Orig.), Anwender u. Games (Ace 64K u.a.). Alles, auch Disks, Zubehör, Bücher f. nur 400,– DM. MPS 801 f. 170.– DM. Kompl. 475,– DM. Tel. 02102/470915, Frank fragen

---

CBM 610 – 720 – CBM 610 Wer kann Speicher aufrüsten? Suche weitere Soft- u. Hardware, nur Anwender zu Hobby-Preisen. Tel. 06201/33860. Klaus Derksen, Am Tannenbuckel 18, 6943 Birkenau

---

C–I–G nimmt noch Mitglieder auf. Wir bieten Clubzeitung, Softwarebibliothek, Freesoftbibliothek (kostenlos) f. C16/Plus4. Info gg. 1,80 DM in Briefm. bei C–I–G, c/o Klaus-Dieter Schindler, Luciusstr. 10A, 623 Frankfurt 80, Tel. 069/304941

EPROM-PLATINE f. Plus4 u. C16, lwird in den Ex-Port gesteckt. Für alle Modulsoftware. DM 25,– bei Vorkasse, DM 5,– bei NN. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 h.

****
* PLUS 4 C – 16 *
* 80 Zeichen Textverarb. *
* deutsche Umlaute *
* Disk DM 39,90 Fa. BLK *
* Kaspar-Spät-Str. 15 *
* 8000 München 90, *
* Telefon 089 / 68 82 26 *
****

C16/116/P4: SUCHE Tauschpartner f. Spiele u. Prog. Kaufe auch billige Software. Schickt Eure Listen an: Markus Tillenburg, Bilker Allee 219, 4000 Düsseldorf, Tel. 0211/332738

*** SPEED COPY 1551 kopiert 1 Disk. m. Floppy 1551 in nur 30 Sek.! Menü m. vielen Funktionen, kopiert READ ERRORS 22 u. 23. Prog. gg. 20,– DM (bar!) bei: Rüdiger Siebert, Tegeler Weg 1, 3549 Volkmarsen

---

SUCHE P4/C16-Besitzer, die mit mir Spiele, Programme u. Informationen austauschen. Wer Interesse an einem P4/C16-Freak hat, soll mir so schnell wie möglich schreiben. Schreibt an: Markus Gehlken, Westerdeich, 62, 2256 Vollerwiek

---

HILFERUF! Suche Listing des Drucker-Files f. Seikosha SP180VC m. Superscript 128! Das File CBM-DOT-MATRIX nutzt nicht alle Möglichkeiten! Edwin Oestreich, Am Rain 11, 6423 Wartenberg 1

P4: Netzteil, Quiwi, Becker-Masch.-Sprache, sonst. Spiele zu verk. wg. Systemwechsel. Tel. 02363/66522

C16/116/Plus4-Anwender: der Verein speziell f. uns! M. Clubzeitschr. u. Software-Bibliothek. Info beim hrc e.V., Bauerland 15, 4800 Bielefeld 1

RUSSISCH auf dem C16/116P4? Glasnost macht's möglich ... Russs. Zeichensatz, Klimadiagramm, Wechselkurse auf Disk. f. DM 10,– (inkl. Porto u. Verpackung). Info: 80 Pf. od. Antwortschein. Olaf Trzebin, Kornweg 7, 3053 Hohnhorst

---

C-I-G nimmt noch Mitglieder auf: C16/Plus4. Wir bieten kostenlose Clubzeitung, Tips u. Tricks auch f. SCRIPT/PLUS. Free-Soft-Prog. u. Tauschbörse. Geringer Beitrag. Info gg. 2,– DM bei C-I-G, c/o K.D. Schindler, Luciusstr. 10A, 6000 Frankfurt 80

VERKAUFE SUPER-PROGRAMME f. C16/116/P4 z.B. BMX-Racers, Vegas, Jackpot, Galaxy, Music-Master, Big-Mac, Rockman, Tom, Paint-Box, Hektik. Kostenlose Info bei: M. Greifenhagen, Stöckelstr. 8, CH-8610 Uster. Jeder Brief wird beantwortet.

TAUSCHE jede Menge Spiele un. Anwend.-Prog. f. C16 u. Plus4 gg. Free-Softprog. z.B. DINA4 Hardcopy, Dt. Zeichensatz f. SCRIPT/PLUS. Zuschriften bitte m. Free-Softliste an: Jürgen Schindler, Luciusstr. 10A, 6000 Frankfurt 80

TAUSCHE meine Software gg. Eure! Disk. od. Tape. Schreibt od. ruft an. Sven Hüppmeier, Ginsterweg 8, 4796 Salzkotten-Niederntudorf, Tel. 02955/1240. Computersystem Plus4

VERKAUFE Software für C16/Plus4! Liste gg. frank. Rückumschlag von: Sven Kuessner, Bredde 6, 5758 Fdbg.-Ardey

---

Suche f. PLUS4 Kopiermodul od. Prog. Tape/Tape, ebenfalls suche ich einen Drucker f. meinen Plus4. Angebote bitte an: Jachowski, Hauptstr. 5, 2244 Süderdeich

---

Hey, Plus4/C16-USER (64K)! Wer will mit mir Top-Prog. tauschen? An Tobias Larsen, zum Wiegele 22, 6951 Neckarzimmern, Tel. 06261/13422. Eure Liste schicken! Habe z. B. ACE I u. II, Mercenary, Summer u. Winter Games auf Disk.

PLUS4, Drucker, Brother HR5C, Floppy 1551, Joystick, Programme, Bücher, Zeitschr., sonst. Zubehör f. zus. DM 500,–. Tel. 089/3202781

Für P4, C16 etc. Drucker-Umschalter, 6-pol., seriell, 1 Eingang, 2 Ausgänge, 60,– DM u. Porto. P. Range, Feldstedter Weg 14, 1000 Berlin 49

SUCHE Floppy u. Drucker f. C16/Plus4. Zahle je bis zu 50,– DM. Meldet Euch bei: M. Poser, Preetzerstr. 76, 2300 Kiel 14, Tel. 0431/77780

HILFE ! ! ! Wer schenkt mir, einem armen, abgebrannten Schüler, den C16 zwecks Schulunterricht? Leider kann ich höchstens 10,– DM bezahlen. BITTE! Melden bei: Carsten Bullert, Lewerentzstr. 57, 4150 Krefeld 1. DANKE!

BIETE Software f. C16/Plus4. Nur Disk. 1328 Blöcke f. nur 20,– DM (Spiele u. Anwender). Schein an: M. Schroeer, Dorfstr. 4, 4670 Lünen. Kein Porto! Lohnt sich!

SUCHE Skat-Prog. f. Commodore Plus4. Bernhard Lettow, Wickedestr. 6, 2400 Lübeck

---

Div. DATA-BECKER-BÜCHER FÜR C16/Plus4 zu verkaufen. Z.B. Tips u. Tricks, Effektiv u. kreativ, Grafik- u. Masch.-Sprache-Buch sowie 64er Sonderhefte. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 Uhr

---

Suche Orig.-Handbü. v. Plus4 sowie weitere Lit. zu Gerät u. integrierter Software zu zivilisierten Preisen u. Leergehäuse v. Steckmoduln od. defekte Spielmoduln. Tel. 0731/42314, Samstag/Sonntag. Rufe zurück!

*** VC 1520 *** Suche Plotter 1520! Bis 125,– DM. Nur funkt.-tüchtig! M. Handbuch! (Ich würde auch meine ganzen Prog. hergeben, 800 St.!) Info bei: Frank Bauer, Seeor 2, 8261 Polling

---

VERKAUFE knapp 1/2 alten C16 m. Datas., fast 40 Spielen, Basic-Kurs, Lernbuch, 6 Comp.-Heften. Preis: 200,– DM. Tel. 06142/59383

SUCHE günst. Floppy f. C16 u. Speichererw. 64 od. 128 KB. Angebote: P. Sadowski, Fontanehof 7, 3180 Wolfsburg 1

ROM-LISTING C16/Plus4 f. 30,– DM; Masch.-Sprache DB 15,– DM; Programmierung des 6502 DM 20,–; alles zus. 50,– DM. Heinrich Lurz, Schützendelle 63, 5012 Bedburg, Tel. 02272/7942

Für C16 u. Plus4: Eprom-Platine m. Sockeln f. nur DM 20,–. Div. Software auf Eprom lieferbar. Liste gg. Freiumschlag. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. ab 18 Uhr

---

VERKAUFE C16, 6 Wo. alt, DM 80,–, Datas. DM 30,–. Lehrkurs auf Kass. gratis; Orig.- Kass. Textmanager (M&T) DM 15,–; SCRIPT/PLUS-Modul DM 25,–. P. Schick, Eschersh. Landstr. 24, 6000 Frankfurt 1, Tel. 069/593525

VERKAUFE C116 m. Datasette u. Joystick, Heften u. Spielen f. 150,– DM. Tel. 07271/41265

SOFTWAREKATALOG 1988! Softw. u. Zubehör f. Ihren Commodore 116/16/P4 bietet der Softwarekatalog 88. Sowohl auf Tape als auch auf Disk. Gratis anfordern! Keine Kopien! P. Schäfers, Riekestr. 5, 4402 Greven 1

C16, Datasette, Spiele, Joystick wg. Systemwechsel billig zu verk. Preis: nur 170.– DM. Robert Meinl, Tel. 0731/70520

*** DFÜ *** DFÜ *** Suche Modem f. C16 + 64K! Auch Userp., nur v. DATA GUSS! Würde meine sämtl. Prog. hergeben (800!). Frank Bauer, Seeor 2, 8261 Polling

---

SUPERGRAFIKEN aus allen Bereichen auf 6 Disk. f. C64/C16/P4. Gg. 20,– DM Vorkasse bei: Dirk Bäcker, Simonstr. 28, 56 Wuppertal 1, Tel. 0202/305418

HALLO, C16-FANS! Verkaufe C16-Spiele auf Kass. f. 5,– DM; Quiwi 20,– DM; Winter u. Summer Dames je 20,– DM; Prog.-Sammlung u. Dateimanager je 15,– DM; Datasette 40,– DM sowie 10 Spiele á 10,– DM. Bitte Vorkasse. Die ersten 10 erhalten 1 Disk. gratis. C.P. Pürner, Landeckerstr. 11. 1000 Berlin 33

TAUSCHE UND VERKAUFE C16/P4-Prog. Habe ca. 500. Listen an: Thorsten Sander, Lilienstr. 45, 4300 Essen 1, Tel. 0201/41455

SUCHE f. 10-jährigen Jungen Spiele f. Plus4 u. VC20 günstig auf Disk., evtl. Listings sowie Joystick u. Lightpen. Angebote an: Susanne Kummerer, Nettingsdorferstr. 33, A-4053 Haid/Österr.

Free-Software C16/116/Plus4 zu verkaufen: Monatl. Kosten, Lotto, Vokabeln. Dazu große Liste v. SYS- u. Spiele-Pokes. Schickt 10,– DM u. 1,90 DM in Briefm. an: Wolfgang Koss, Bergstr. 29, 2215 Hademarschen

---

Free-Software: 20 Disk. f. Plus4/C16/116, 66 Disk. f. C64. Info gg. Rückporto. Bitte Computertyp angeben. Viktor Jürgens, Prinzenstr. 131, 2330 Eckernförde

---

SUPER Spiel-, Grafik- u. Anwend.-Prog. f. Plus4/C16/116 abzugeben z.B. Airwolf, Blaze, Ace (64K) usw., nur auf Disk. (dop.-seitig). 1 Disk. 20,– DM-Schein an: R. Steinhauer, Damaschkestr. 42, 5630 Remscheid

---

C16/116/Plus4. VERKAUFE ca. 300 Spiele od. Anwend.-Prog. wg. Systemwechsel auf Kass. (Mit: Formula 1, ACE 16, POD etc.) Nur einmal zu bekommen! Angebote unter: 07136/21328

C16 – GELEGENHEIT – VERKAUFE wg. Systemwechsel: C16, Datasette, Basic-Lehrbuch (m. Kass.), Floppy 1551, Bausatz 64K f. 250,– DM (neuwertig 500,– DM). Helmut H. Ehlers, Gottschedstr. 22, 2000 Hamburg 60, Tel. 040/275979

POP GOES C16/116/PLUS4! Verk. 7 Musik-Masterfiles f. 10,– DM 8 + Rückporto!) nur auf Disk.! Außerdem: 3 Basic-Musik-Demos zusätzl.! Bestellungen an: Jens Helbing, Am Sportplatz 20a, 2220 St. Michaelisdonn

---

ACHTUNG, C16/P4-Anfänger: Wg. Wechsel auf Disk zu verkaufen: 13 Prog. auf Kass. (z.B. Phantom, Ghost'n Goblins, Music Master). NP 160,– DM, VB 60,– DM. Thorsten Sander, Lilienstr. 45, 4300 Essen 1

C16/116 PLUS4 — **Autodata**

Automatischer DATA-Zeilen-Generator

Der eingebaute Monitor des C16 ermöglicht zwar das Abspeichern von Speicherinhalten, aber oft will man die Werte für Maschinenprogramme oder SHAPES in DATA-Zeilen ablegen. Das kleine Programm auf der Rückseite erledigt dabei die ganze Arbeit. Es arbeitet mit der Tastaturpuffer-Methode.

C16/116 PLUS4 — **Bildschirmcode <> ASCII**

Bildschirmcode –> ASCII

```
10 input"bildschirmcode";bc
20 ac=bc-64*(bc<32 or bc>95)-32*(bc>63 and bc<96)
30 print ac
```

Will man die Umrechung oft machen, so ist es sinnvoll, die Werte in einem Feld abzulegen:

```
20 dimac(255):forbc=0to255:ac(bc)=bc-64*(bc<32or
   bc>95)-32*(bc>63andbc<96):nextbc
```

'print ac(1)' ergibt dann 65 (chrS-Code von A)

C16/116 PLUS4 — **Multicolor im HIRES-Modus**

Multicolormodus ein: POKE65286 or 64

Multicolormodus aus: POKE65286 and 191

Im Multicolormodus kann jedes Zeichen seine eigene Hintergrundfarbe haben. Durch POKEn in 65301–65304 einstellen (B: POKE65301,16*L+F).

Dies geht jedoch nicht im HIRES-Modus. Hier kann jedoch vor dem CHAR-Befehl der Hintergrund eingestellt werden und bleibt dann für dieses Zeichen erhalten (siehe Rückseite).

C16/116 PLUS4 — **Zeichen unsichtbar**

Zeichenfarbe = Hintergrundfarbe

COLOR1,RCLR(0),RLUM(0)

Mit RCLR kann man die momentanen Farbeinstellungen erfahren. RLUM liefert die Helligkeit.

Dieser Trick ist nützlich, wenn Programme nachgeladen oder der MONITOR im Programm verwendet wird, ohne daß dies sichtbar sein soll.

C16/116 PLUS4 — **aktuelle Zeilennummer**

Aktuelle DATA-Zeilennummer

Diese ist in den Speicherstellen 63/64 in LO/HI-Darstellung gespeichert. Beispiel für eine Abfrage:

```
10 read a$:zn=peek(63)+256*peek(64)
```

In zn steht nun die Zeilennummer der aktuellen DATA-Zeile.

C16/116 PLUS4 — **HIRES abspeichern**

HIRES-Bild abspeichern

Eine hochauflösende Grafik kann man wie ein Maschinenprogramm abspeichern und laden. Dies geht am besten mit dem MONITOR:

```
s"name",8,1800,4000
```

Bei Kassette statt ',8' ',1' verwenden.
Wird die Farbe nicht gebraucht, dann statt '1800' '2000' einsetzen.

C16/116 PLUS4 — **Adreßvergleich C16 <> C64 II**

Wichtige Adressen beim Umschreiben von C64- in C16-Programme:

C64:	C16	
160–162	163–165	interne Uhr
199	194	REVERSE-Flag
203	198	letzte gedrückte Taste
209–210	200–201	Anfangsadresse der aktuellen Bildschirmzeile
211/214	202/205	Cursor-Spalte/-Zeile
212	203	Quotenmode

C16/116 PLUS4 — **Adreßvergleich C16 <> C64 I**

Wichtige Adressen beim Umschreiben von C64- in C16-Programme:

Bis zur Adresse 75 sind alle für BASIC interessanten Speicherstellen in ihrer Funktion identisch.

C64:	
POKE 53280,F	Hintergrundfarbe setzen
POKE 53281,F	Rahmenfarbe setzen
POKE 646,F	Zeichenfarbe setzen
C16:	alle Farbeinstellungen mit COLOR...

ASCII –> Bildschirmcode

```
10 input"ASCII-Code";ac
20 bc=ac+33*(ac=255)+64*(ac>63)+32*(ac<96)-32*
   (ac<160)+64*(ac>191)
30 poke 3072,bc
```

```
190 rem ****autodata*******************
200 input"start-,endadresse";a,e
210 zn=1000
220 printchr$(147)zn"data";
230 for a=a to a+15
240 if a>e then printchr$(157)" ":print"end:":goto
    270
250 printmid$(str$(peek(a)),2)",";:next
260 printchr$(157)" ":print"a="a":e="e":zn="zn+10"
    :goto220"
270 t=1319:poket,19:poket+1,13:poket+2,13:poket+3,
    13:poke239,4:end
```

Laden einer HIRES;Grafik:

Dies kann im MONITOR geschehen:

1"name",8
oder direkt mit dem LOAD;Befehl:

load"name",8,1 (bzw. ',1,1' bei Kassette)

Demo:

```
10 rem *** im hiresmodus ****
20 color0,2,5:graphic1,1:fors=1to14
30 for1=1to7:f=1:z=1
40 color0,1,f:char,s,z,mid$(t$,s,1)
50 next1:nexts
60 color0,4,0:char,1,1,"SAZXQW"
70 getkeya$:graphic0
```

Rahmenfarbe = Hintergrundfarbe

CLOR4,RCLR(0),,RLUM(0)

Will man später wieder die ursprüngliche Einstellung zurück, so muß man die entsprechenden Einstellungen speichern:

```
10 for i=0 to 4:f(i)=rclr(i):1(i)=rlum(i):next
```

alte Einstellung:

```
20 for i=0 to 4:colori,f(i),1(i):next
```

Nummer der aktuellen BASIC-Zeile

Diese steht in 57/58.

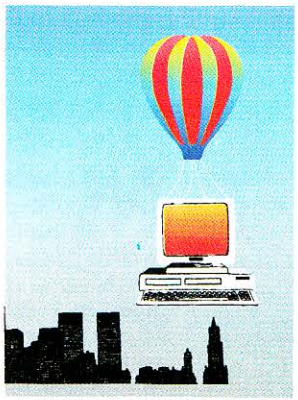
```
100 de=peek(57)+256*peek(58)
200 print de
```

ergibt 100

C64	C16	
204	213	Länge der aktuellen Bildschirmzeile
215	206	letztes Zeichen (I/O)
198	239	Anzahl der Zeichen im Tastaturpuffer
828–1019	819–1010	Kassettenpuffer
631–640	1319–1328	Tastaturpuffer
649	1343	Tastaturpuffergröße
650	1344	Tastenwiederholung
653	1347	Flag für SHIFT/CTRL und C=

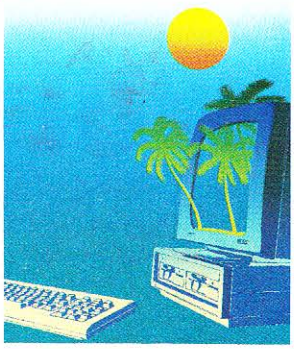
C64	C16	
55296	2048	Farb-RAM für Textmodus
1024	3072	Bildspeicher (Textmodus) (je 1000 Byte)
8192–16383		Farb-RAM für HIRES (beide gleich)

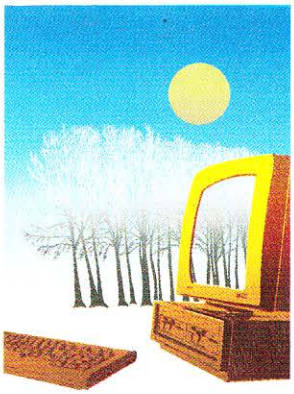
POKE198,0:WAIT198,1 wartet auf Tastendruck (ersetzen durch GETKEY A$)



AT 488 31

AT 488 32

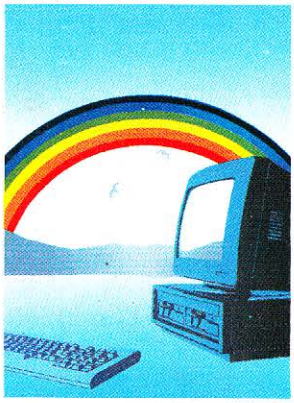




AT 488 33

AT 488 34

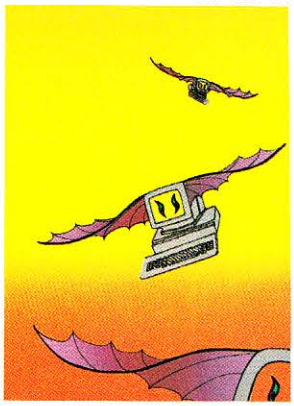
AT 488 35



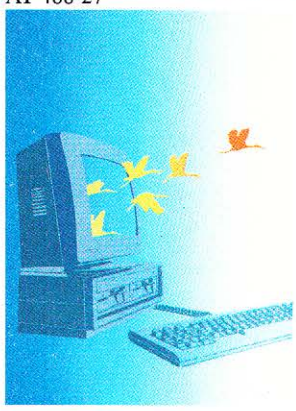
Edition Computer-Kunst

AT 488 29

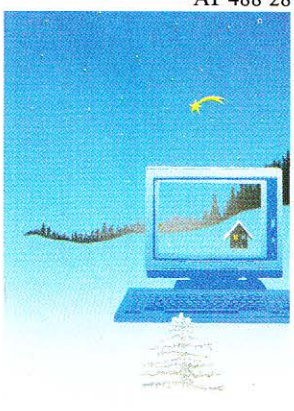
AT 488 30

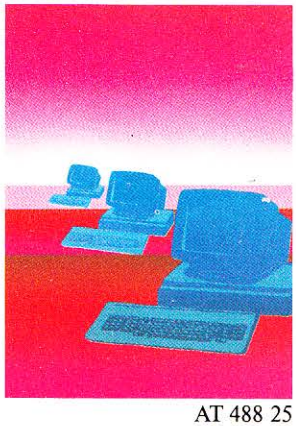


AT 488 27



AT 488 28





AT 488 25

AT 488 26

Computer sind nicht nur Gebrauchsgegenstände. Sie werden zunehmend auch von Künstlern als Motiv entdeckt. Einige besonders gut gelungene Arbeiten haben wir für Sie, unsere Leser, reservieren können. Alle Motive sind strikt auf eine Auflage von 99 Exemplaren limitiert und von der Künstlerin, Sybille Areco, handsigniert und numeriert. Die Exponate werden nach Bestelleingang im 24-Farbendruck von Hand gefertigt, die Vorlage nach dem 99. Druck vernichtet. Unser Angebot: Jedes Motiv nur DM 85,-, zwei Motive DM 150,-, drei DM 210,- und vier Motive nur DM 250,-. Jedes Bild ist 30 x 40 Zentimeter groß und kommt im Passpartout in stabiler Verpackung (im Preis enthalten).
Lieferzeit nach Bestelleingang: ca. drei Wochen.

Bestellcoupon

Hiermit bestelle ich in Kenntnis ihrer Verkaufsbedingungen folgende Exponate:

Nr.: _____

_____

Ich zahle: (Zutreffendes bitte ankreuzen!)
per beigefügtem Scheck ☐ Schein ☐ Gegen Bankabbuchung am Versandtag ☐

Meine Bank (mit Ortsname) _____

Meine Kontonummer _____

Meine Bankleitzahl _____ **(steht auf jedem Bankauszug)**

Nachname _____ **Vorname** _____

PLZ/Ort _____ **Str./Nr.** _____

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Wichtig: Scheckeinreichung und Bankabbuchung erfolgen erst nach dem Versand. Keine Nachnahme möglich. Auf Wunsch Rechnung mit ausgewiesener Mehrwertsteuer.

Unterschrift _____
Bitte ausschneiden und einsenden an

AKTUELL-VERLAG
Heßstraße 90
8000 München 40

Trau keinem unter 24

(Nadeln!)

Die Typenraddrucker sind tot, darüber braucht man nicht mehr lange zu diskutieren. Diese Zwischenstufe einer Schreibmaschine und eines Druckers konnte sich auf dem Markt nicht lange halten. Als nächstes geht es den Neun-Nadel-Matrix-Druckern an den Kragen. Schneider und NEC haben bereits das Messer gegen die Konkurrenz gewetzt. Beide Unternehmen brachten getrennt voneinander einen 24-Nadel-Drucker zum Preis unter 1000 Mark (ohne MwSt.) auf den Markt. Wurde damit schon das Ende der Neun-Nadel-Technik eingeläutet?

NEC hatte den Sprung in die Verkaufshitliste mit der Vorstellung des NEC P5 gestartet. Daraufhin schlossen sich in rascher Folge der P5XL, der P6, P7 und der P9 an – alles hochqualifizierte 24-Nadel-Drukker der mittleren Preisklasse für den professionellen Anwender. Dieses Angebot wurde nun mit dem P2200 nach unten hin abgerundet. NEC greift mit seinem neuesten Modell den Marktbereich der Homecomputer-Besitzer und der semiprofessionellen Anwender an. Rechnen wir beim P2200 mal die Mehrwertsteuer dazu, so beläuft sich der empfohlene Verkaufspreis auf 1136,– Mark – ein Sensationspreis für die Features, die in dem kleinen handlichen Kraftpaket stecken.

Im Gegensatz zu seinem größeren Bruder, dem P6, ist

Kompakt und fast komplett aus Plastik – so zeigt sich der neue Preisbrecher unter den 24-Nadel-Druckern von NEC. Die Verarbeitung ist nicht optimal, doch das Schriftbild kann überzeugen.

beim P2200 bereits ein Formulartraktor fest eingebaut.

Allerdings vermißten wir einen Feststellhebel, um die Papierbreite zu fixieren. Die Erschütterung der Druckkopfbewegung treibt beide Stachelräder zusammen und stört den einwandfreien Durchfluß des Endlospapiers. Dies kann nach längerem Gebrauch, wenn sich das Plastik der Führungen bereits etwas abgeschliffen hat, zum vorzeitigen Austausch des Traktors führen.

Es wurde viel Plastik verwendet

Das Lochrandpapier kann wahlweise von hinten oder durch einen Frontschacht eingeführt werden, wobei der Traktor einmal im Zug- und einmal im Schubbetrieb benutzt wird. Dem vorderen Schacht ist in der Hauptsache jedoch eine viel nützlichere Funktion zugedacht. Er besitzt während des Druckvorgangs absolute Priorität. Um zwischendurch schnell mal einen Brief auf DIN A4-Papier, einen Briefumschlag oder Etiketten zu bedrucken, werden diese Formate einfach von vorne in den Schacht geschoben. Das EDV-Papier bleibt dabei eingespannt, lediglich der Traktor wird desaktiviert. Ein Rändelrad und ein Hebel steuern die Papiereinzugsart. Zur genauen Einstellung des gewünschten Formats dienen im vorderen Schacht eine Skala und eine verschiebbare Anlegekante für den linken Rand.

Direkt über dem Schacht befindet sich das Bedienungspaneel. Der P2200 verzichtet selbstverständlich auf das Mäuseklavier und läßt sich entweder softwaremäßig oder über das Bedienungsfeld dauerhaft einstellen. Die gewünschten Parameter werden in einem EEPROM abgelegt. Durch einen kurzen Tastendruck wechselt der P2200 beispielsweise von Draft in den Briefqualitätsmodus. Genauso einfach ist die Schriftdichte von 10 cpi, 12 cpi oder Proportionalschrift angewählt. Der Quiet-Schalter für Leisetreter reduziert die normale Lautstärke von 57 dB(A) bei verminderter Geschwindigkeit um drei Dezibel. Der Unterschied ist so stark, daß sogar nachts in einer Mietskaserne ruhigen Gewissens noch die persönliche Post erledigt werden kann. Auf den Grafikmodus hat diese Funktion allerdings keinen Einfluß.

Der Druckkopf benötigt viel Platz

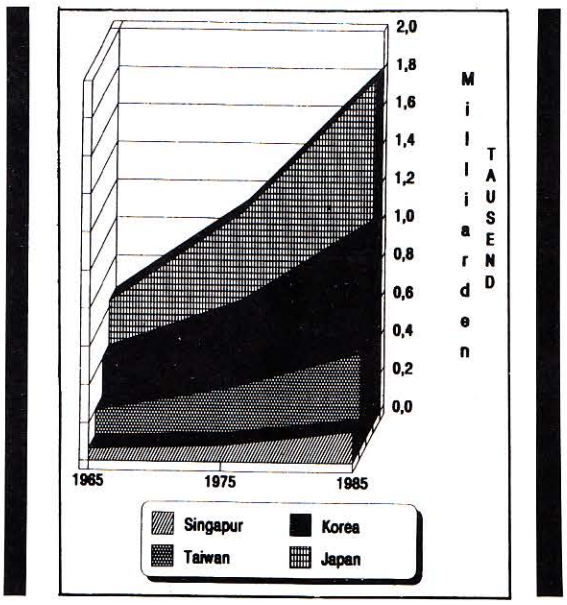
Den meisten Platz im Druckkopfbereich nimmt unzweifelhaft der eingebaute Traktor für sich in Anspruch. Die Walze ist so weit nach unten gerutscht, daß sie aufgrund der doppelten Papierzuführungstechnik nicht mehr sichtbar ist. Wehe dem Anwender, dem einmal ein Etikett unter der Walze kleben bleibt. Um diesen Störenfried zu entfernen, muß der Drucker fast komplett auseinandergebaut werden.

Der Druckkopf mit seinen 24 versetzt angeordneten Nadeln (0,2 mm) ist durch einen Hitzesensor gesichert. Beim Erreichen von 75 Grad Betriebstemperatur schaltet der P2200 automatisch in den unidirektionalen Druckmodus um und gönnt dem Druckkopf nach jeder Zeile eine Sekunde Pause. Bei 90 Grad wird der Druckvorgang für sechs Sekunden unterbrochen und eine optische Warnmeldung ausgegeben.

Die Farbe bringt der P2200 mittels einer Mini-Farbband-Kassette aufs Papier, die direkt auf dem Schlitten des Druckkopfes eingerastet wird.

Der Zeichensatz kann von IBM auf Epson umgestellt werden. Hierin finden sich Sonderzeichen aus 13 Nationen, die natürlich auch softwaremäßig ausgewählt und gewechselt werden können.

Sechs Schriftarten sind im P2200 bereits fest implementiert. Draft Gothic, LQ Courier, LQ Super Focus, LQ OCR-B, LQ ITC Souvenir und LQ Bold PS. Weitere Schriftarten können entweder über den Download-Modus oder über zusätzliche Schriftkassetten eingesetzt werden.



Für die Kassetten besitzt der P2200 auf der Rückseite einen Schacht, dessen Klappe zum Herausbrechen bereits vorperforiert ist. Insgesamt ergeben sich durch die Kombination der verschiedenen Druckarten mehr als 130 Schriftvariationen.

Die Druckgeschwindigkeit wird von NEC bei Draft/Elite mit 168 cps, bei Draft/Pica mit 140 cps und in Briefqualität (Pica) mit 47 Zeichen in der Sekunde angegeben. Diese Werte ergeben sich bei der Berechnung einer Zeile ohne Vorschub und Wagenrücklauf.

Im Test mußten diese Angaben von uns jedoch stark nach unten korrigiert werden. Bei 66 Zeilen mit jeweils 78 Spalten ergab sich für Draft/Pica eine Druchschnittsgeschwindigkeit von 101 Zeichen in der Sekunde, und LQ kam mit 40 Zeichen in der Sekunde aufs Papier.

Im Grafikmodus steht der P2200 seinen erwachsenen Brüdern in nichts nach. Er beherrscht in fünf Stufen Dichten von 60 Punkten pro Zoll bis hin zu einer Auflösung von 360 Punkten. Zum Vergleich: Laserdrucker begnügen sich zum größten Teil mit 300 dots/inch.

Der Puffer ist mit 8 KByte sehr großzügig ausgestattet. Drei bis vier Schreibmaschinenseiten können darin zwischengespeichert werden. In der Standardausstattung ist der P2200 mit einer parallelen Schnittstelle ausgerüstet, kann auf Wunsch jedoch auch mit einem seriellen Port erworben werden.

Das vorbildliche deutsche Handbuch verhilft mit seinen zahlreichen Tabellen, Diagrammen und Beispielen sowohl dem Einsteiger wie auch dem Programmierer zur optimalen Ausnutzung seines Gerätes. Als Druckertreiber kann jeder NEC-Treiber ab P5 aufwärts gewählt werden.

Fazit

Angesichts des niedrigen Preises kann sich theoretisch jetzt jeder private Anwender einen leistungsstarken 24-Nadel-Drucker gönnen.

Vom Schriftbild, der breiten Anwendungspalette und der komfortablen Bedienung her präsentierte sich der P2200 von seiner besten Seite. Weniger gefallen haben uns allerdings die vielen „Weichplastikteile", die relativ schnell verschlissen sind. Auch der Gedanke, bei einem Papierstau unterhalb der Walze das komplette Gerät in seine Einzelteile zerlegen zu müssen, brachte uns nicht gerade zum Jubeln. ■

Viel Platz für Kreativität

Abgerundet sind nicht nur die Kanten des Brother M-1724L, sondern auch die komplette Ausstattung dieses 24-Nadel-Druckers. Eine parallele und eine serielle Schnittstelle, ein 24-KByte-Puffer, eingebauter Traktor und die einfache Bedienung gehören längst nicht zum Standard in dieser Preisklasse.

Brother's neuer 24-Nadel-Drucker M-1724L ist ein Prachtstück in bezug auf moderne Technik und anwenderfreundlichen Komfort.

Der Werbespruch einer bekannten Zigarettenmarke „... und alles geht wie von selbst" ist in dem kaum zehn Zentimeter hohen Gerät zur Realität geworden.

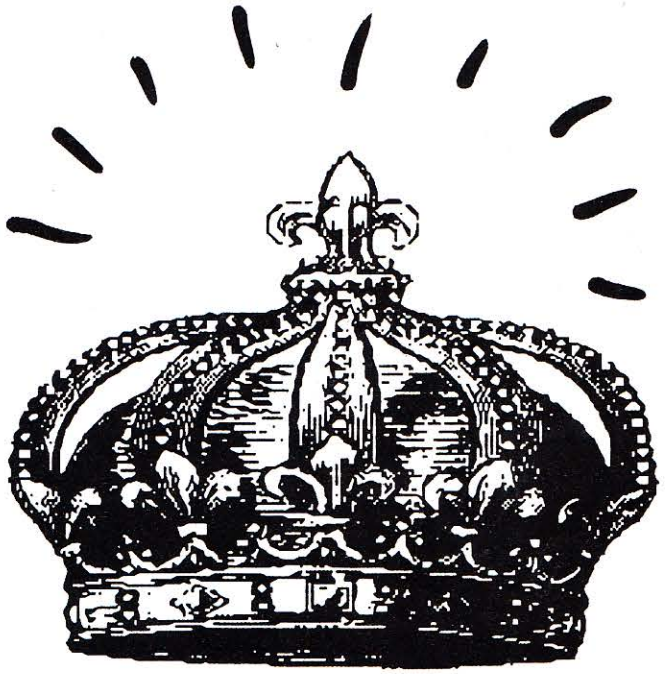
Der Komfort beginnt schon bei der Bestückung des Standardgerätes: ein deutsches Handbuch mit ausführlicher Intallations- und Bedienungsanleitung, eine parallele und serielle Schnittstelle, 24-KByte-Datenspeicher, integrierter Präzisionstraktor, manueller Einzelblatteinzug und eine Farbbandkassette sind im Grundpreis des M-12724L enthalten.

Die Druckbreite erreicht in Pica bis zu 136 Zeichen/Spalte und in komprimierter Schrift (17cpi) bis zu 272 Zeichen.

Die beiden Schnittstellen sowie der Netzstecker wurden seitlich herausgeführt und stören dadurch nicht den einwandfreien Durchfluß des Endlospapiers.

Die Lebensdauer des flachen Druckkopfes wurde von Brother mit 100 Mio. Punkten pro Nadel (ϕ 0,02 mm) angegeben. Das Farbband für 40 Mark verkraftet etwa 2,5 Millionen Zeichen.

Die Geräuschentwicklung ist während des Betriebes mit 58 dB(A) nicht gerade als leise zu bezeichnen.



Brother's M-1724L ist in der Lage, den Epson LQ-1000/1500, den Diabolo 630, Brother's eigene Typenraddrucker-Serie HR und natürlich den IBM-Grafikdrucker zu emulieren. Hierzu stehen 96 ASCII-Zeichen, 43 internationale Zeichen aus 16 Sprachen und der erweiterte IBM-Zeichensatz zur Verfügung.

Die Schriftarten gliedern sich jeweils im Draft- und LQ-Modus in Pica, Elite und Brougham. Doppelte Höhe und/oder doppelte Breite, kursiv, sub- oder superscript, unterstrichen, komprimiert, wie auch Fett- und Schattendruck zählen zu den normalen Eigenschaften des 24-Nadel-Druckers.

Eine Erweiterungskarte, die einfach innerhalb einer Abdeckplatte versenkt wird, erhöht das Schriftenangebot noch um drei weitere Arten (Prestige, Quadro und Anelia PS).

Zusätzlich ist auf dieser Karte nochmals ein 32-KByte-Puffer enthalten, der dem Brother damit einen Gesamt-Speicherbereich von stattlichen 56 KByte verleiht. Etwa 25 Schreibmaschinenseiten Text lassen sich zwischenspeichern und geben den Rechner frühzeitig für andere Aufgaben frei. Mehrfache Kopien desselben Dokuments erledigt der M-1724L ohne Unterstützung durch den Rechner. Eine Copy-Funktion gestattet in Verbindung mit dem großzügig bemessenen Speicher diese monotone Tätigkeit.

Zum Einstellen der Standard-Parameter, wie zum Beispiel nationalem Zeichensatz, Zeilenvorschub oder Parameter der seriellen Schnittstelle, dienen drei unterhalb des Druckkopfbereiches angebrachte DIP-Schalterblöcke. Der weitaus größere Rest der Einstellmöglichkeiten wird entweder direkt über Steuercodes durch die Software übermittelt oder auf dem umfangreichen Frontpanel mittels der Folientasten „programmiert".

Durch eine Kombination der verschiedenen Tasten lassen sich unter anderem Schriftbilder und -größen – sowie acht verschiedene Papierlängen zwischen vier und acht Zoll bequem anwählen. Auch die zusätzlichen Schriftarten der Erweiterungskarte sind über die Folientastatur direkt erreichbar.

Der Traktor für Endlospapier ist, verborgen unter einer dunkel getönten Plexiglas-Abdeckung, in der Rückseite des Gehäuses fest integriert. Feststellhebel und diverse Papierführungshilfen sorgen für einen reibungslosen Einzug. Sowohl Endlos- wie auch DIN A4-Papier werden vollautomatisch eingezogen und

ein Zoll unterhalb der Blattoberseite zentriert. Selbst der Papierandruckhalter bewegt sich wie von Geisterhand zurück, um wenig später das Papier fest an die Walze zu drücken.

Doch wie verhält sich der M-1724L, wenn von EDV- auf Einzelbögen umgeschaltet werden soll?

Er macht's mit einem Tastendruck, ohne auch nur ein einziges Hardwareteil auszutauschen: Knopfdruck – das Endlospapier wird in eine Parkposition auf die Stachelräder zurückgezogen; einen Schalter umstellen – der Traktor wird stillgelegt und der Walzenantrieb aktiv. Zuletzt wird nur noch das Lämpchen für Einzelblatteinzug auf dem Bedienungspanel zum Glühen gebracht und der Brother zieht anstandslos die Schreibmaschinenseite ein. Schneller und anwendungsfreundlicher geht's kaum.

Für die automatische Einzelblattverarbeitung bietet Brother optional einen Schacht für etwa 80 Bögen an. Der empfohlene Verkaufspreis von 683 Mark ist für einen Schacht in DIN A3-Breite sicherlich nicht zu hoch angesetzt. Gewisse Differenzen nach unten ergeben sich dabei zusätzlich noch von Händler zu Händler.

Mit dem automatischen Einzug gab es keinerlei Installationsprobleme.
Der Schacht wird auf die Walze aufgesetzt und durch einen Stecker auf der rechten Gehäuseseite zum Leben erweckt. Die Koordination mit dem Endlospapier war genauso problemlos wie bei der manuellen Version.

Ein Original und zwei Kopien sind von vornherein vorgesehen. Die Andruckstärke regelt ein kleiner Hebel neben dem Walzendrehknopf. Letzterer ist offensichtlich für Linkshänder gedacht.

Selbst Standard-Postkarten der Deutschen Bundespost werden klaglos verarbeitet, wenn der Drucker zuvor durch eine weitere Hebelstellung darauf vorbereitet wurde.

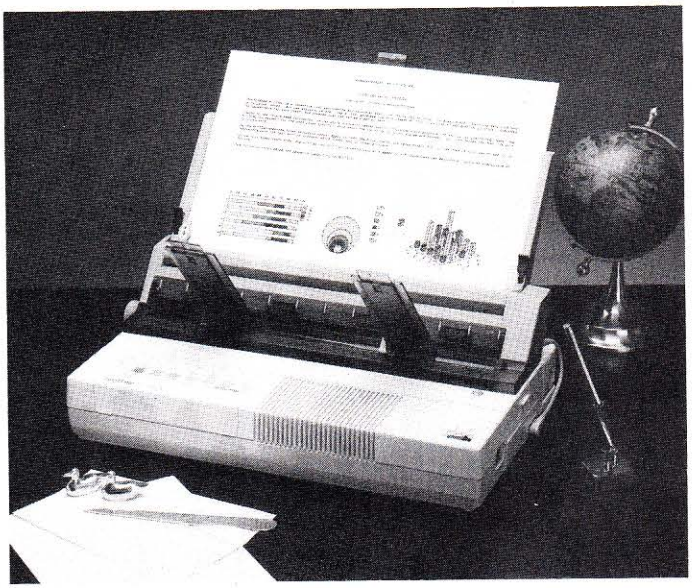
Das Schriftbild in Draft Quality hat im Test durchaus überzeugt. Die Briefqualität der einzelnen Zeichen (LQ-Modus) konnte dahingegen trotz der 24 versetzt angebrachten Nadeln den Matrixdrucker immer noch nicht verleugnen. Anwender, die Schreibmaschinen-Qualität gewohnt sind, müssen beim M-1724L einige Abstriche machen. Daß auch mit Matrixdruckern bessere Qualität zu erzielen ist, macht beispielsweise NEC deutlich.

Im Grafikbereich setzt der Brother M-1724L bis zu 360 Punkte pro Inch (ein Inch = 2,54 cm). Mehr ist im Augenblick aus einem Matrixdrucker nicht herauszuholen. Zur Geschwindigkeit des Matrixdruckers: Brothers gab Werte vor, die unseren widersprechen. Um eine reelle Durchschnittsgeschwindigkeit zu erhalten, muß die Zeilenvorschubzeit mit in Rechnugn gezogen werden. Brother ließ diesen Faktor, wie jeder andere Druckerherstller, gekonnt aus.

Ein Vergleich der angegebenen und der tatsächlich errechneten Durchschnittswerte zeigt:

Druckart	Werksangabe	gemessen
Draft 10 cpi	216 cps	132,0 cps
Draft 12 cpi	180 cps	114,8 cps
LQ 12 cpi	72 cps	51,3 cps

* * *



Neueste Technologie zum vernünftigen Preis: Bereits in der Grundausstattung enthalten sind eine serielle- und eine parallele Schnittstelle, ein 24-KByte-Puffer und diverse Schrifttypen. Ein professioneller Ausbau des Brother M-1724L ist problemlos möglich.

Fazit

Der Brother M-1724L bietet ein Maximum an Komfort und Anwenderfreundlichkeit. Im Test gab es keine Probleme.

Für das Handbuch hätten wir uns ein Stichwortverzeichnis gewünscht, da anhand des Inhaltsverzeichnisses kaum eine bestimmte Funktion gezielt gefunden werden kann.

Die Schrift-Qualität im LQ-Modus könnte besser sein. Ansonsten blieb Brother seiner gewohnten Qualität treu und brachte mit dem DIN A3-Drucker M1724L ein leistungsstarkes Gerät zu einem ausgezeichneten Preis/Leistungs-Verhältnis auf den Markt.

Der Preis für die Standardausstattung beträgt 1995 Mark einschließlich Mehrwertsteuer. ■

IMPRESSUM

C16-P4 Special

erscheint zweimonatlich in der CA-Verlags GmbH (i.G.)

VERANTWORTLICH FÜR DEN INHALT:
Anton Kult
Alfons Mittelmeyer

REDAKTION UND STÄNDIGE MITARBEITER:
Peter Basch, Harald Beiler, Rosemarie Huber, Lothar Miedel, Michael Reppisch, Rudolf Schmid-Fabian, Torsten Seibt, Hermann Wellesen, Bernd Welte

GESCHÄFTSFÜHRER:
Werner E. Seibt

ANSCHRIFT FÜR ALLE VERANTWORTLICHEN:
Postfach 1107,
8044 Unterschleißheim
Tel.: 089/1298011
Telex: 5214428 cav-d

ANZEIGENVERWALTUNG:
ADV-Mediendienste,
Aindlingerstr. 17–19,
8900 Augsburg 1
Tel.: 0821/7904-227
Telex: 533502
Teletex: 821887
Telefax: 0821/7904-243

VERANTWORTLICH FÜR DEN ANZEIGENINHALT:
Brigitte Kostić
Es gilt Preisliste Nr. 8 vom 1.1.1988
Media-Unterlagen bitte anfordern.

VERTRIEB:
Verlagsunion Wiesbaden

Printed in Germany by ADV, 8900 Augsburg 1

COMPUTERN LEICHT GEMACHT

Das PC-Magazin

NEU

Jetzt an ausgewählten Kiosken und im Bahnhofs-Buchhandel