C16 Sprite

Does your C16 lack a little something? Then add sprites with this useful utility.

By Frank Bingley

When I first acquired a Commodore 16, I was impressed by the improvements made to Commodore basic. But it wasn't too long before it dawned on me that Commodore had forgot one important feature which 64 owner's had come to take for granted. It didn't support sprites!

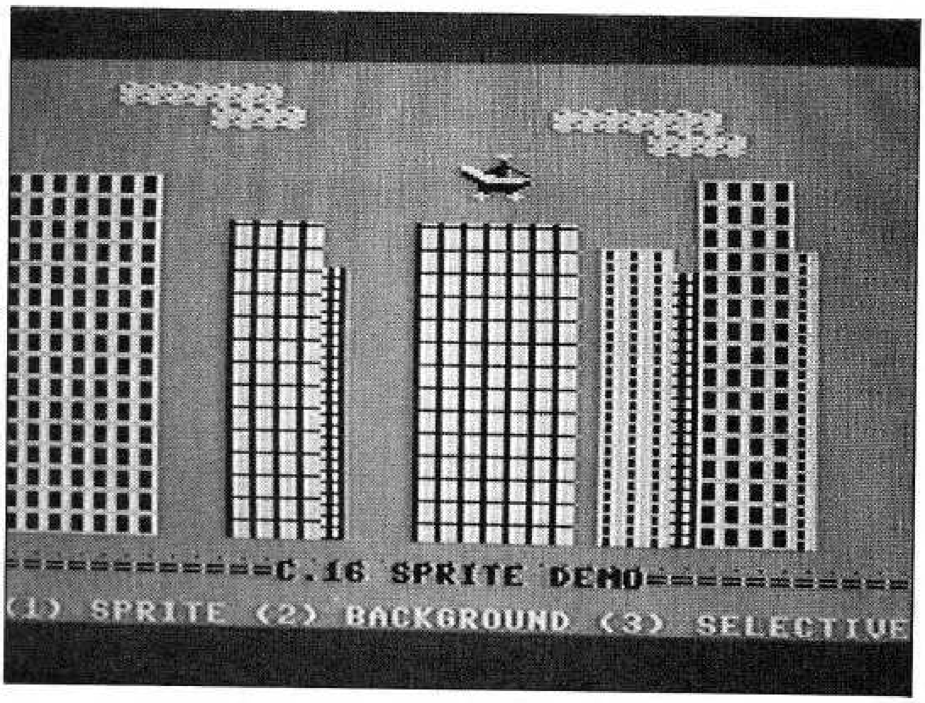
For most serious applications and general computer use, the computer performed quite adequately. It was only when I came to games programing that this missing feature became something of a problem. The only way possible to move object blocks of any size around the screen was to incorporate the necessary characters into a string, along with complicated control characters. These strings were then guided around using the handy CHAR command. This worked reasonably well, but suffered from screen scrolling problems and wiped out any other characters that got in the way!

These problems prompted me to write a machine code routine which would, to some extent, mimmick the sprite facility on the 64. **C16 SPRITE** is an interupt-driven routine with which it is possible to display a six character sprite anywhere on the screen. Certain pokes will instantly move the sprite around whilst others control which sprite data block is displayed as well as sprite to background priority. Other features are collision detection and selective priority (not found on the 64).

Along with notes on how to use this utility, I've included a short basic program which demonstrates how easy the system is. Equally good results can be obtained by using this routine as part of a basic or machine code program. To effectively use **C16 SPRITE**, it will be necessary to know briefly how the routine works. A block of six characters (3 wide by 2 deep) will be displayed on the screen. Their position will be determined by two addresses which hold their X and Y co-ordinates. The actual characters (and colours) displayed will be held in a sprite block buffer, just above the utility. Another address will hold the sprite block number. The sprite will be displayed either behind or in front of any other characters on the screen according to the contents of two other addresses. The system automatically remembers what characters and colours lie beneath the sprite, and replaces them when the sprite is moved or turned off.

Table 1 shows which addresses do what.

HEX	DEC	DESCRIPTION
$3E44	1594ø	X Position
$3E45	15941	Y Position
$3E6D	15981	Sprite to background priority
$3E6E	15982	Priority end character
$3E6F	15983	Collision detection
$3E72	15986	Sprite block pointer
$3Cøø-$3EE9	1536ø-163ø5	C16 SPRITE utility resides here
$3Føø-$3FFF	16128-16383	Sprite data buffer
SYS DEC ("3E95")		Turn on sprite
SYS DEC ("3E75")		Turn off sprite



```
============C.16 SPRITE DEMO===========
(1) SPRITE (2) BACKGROUND (3) SELECTIVE
```

Typing in C16 Sprite

Type in listing 1 as it appears and save to tape or disk after correcting any errors. Now run the program. Next, with the data in place, enter the monitor and save a boot-load version by typing:
S 'C16. SPRITE',08,3C00,3EE9
Remember to use 01 instead of 08 if using tape.

How To Use C16 Sprite

Let's test the system. First protect the utility from being overwritten by Basic. To do this type:
POKE 52,47:POKE 56,47:CLR
Now load 'C16 SPRITE' by entering monitor and typing:
L 'C16. SPRITE',08
Tape users will need to position the tape at the beginning of the utility before this command.

Next, we need to form a sprite. Either the standard character set or user defined graphics may be used, but for this exercise let's keep it simple! Our test sprite will be a red reverse block.

At this point, I will explain how the sprite data is stored in the buffer. The buffer is divided into 21 blocks, each containing 12 bytes. The first 6 bytes of each block are for character data, leaving the other 6 for character colour. So, to fill the first block with reverse spaces, type:
FOR A=0TO5 :POKE 16128+A,160:
NEXTA
16128 is the first address for the first block, whilst 160 is the screen code for a reverse space. Now we need to add the colour. To do this, type:
FOR A=0TO5 :POKE 16134+A,82:
NEXTA
This will colour all of the reverse spaces a medium red. Having placed the sprite data into position, it will be necessary to let the system know where it is. So, to set the sprite pointer to point to our sprite, type:
POKE 15986,0

Finally, before we turn on the sprite, choose a position for it. For the X position, choose any value between 0 and 37. The Y position can be any value between 0 and 23. Addresses 15940 and 15941 are for the X and Y positions respectively, so choose values for X and Y and type:
POKE 15940, X: POKE 15941, Y

Now for the moment of truth! Clear the screen and turn on the sprite. Type:
SYS DEC('3E95')

If the above procedure has been followed correctly, then a red block will be displayed on the screen at your chosen position. If not, then turn off the sprite (SYS DEC '3E75') and try again. Please note that the system will not respond to position values outside maximum and minimum parameters.

Don't be afraid to experiment a little more with our sprite. Try altering some of the addresses in the above chart to alter sprite positions and colours or characters. Remember that **C16 SPRITE** is interupt driven, so that it will not be necessary to keep turning on the sprite, just alter the appropriate address for instant results. Also remember to turn off the system when not required (SYS DEC('3E75')), or strange things may start to happen!

If you wish to change sprite to background priority, then a 0 in address 15981 will cause the sprite to lie in front of any characters on the screen. A 255 in this address will change it to lie behind. It is possible to make the sprite lie in front of some characters and behind others. This can be very useful, and used to good effect. Here is an example of how to use this facility:
POKE 15982,26:POKE 15981,1
Now, any characters whose screen code is within the range 1 to 26, (i.e. alpha numerics), will now always appear in front of the sprite. Any others will be behind. Two rules apply here: all characters with screen codes within selected parameters will be foreground; and 127 is the maximum amount of foreground characters.

Another handy feature of this utility is collision detection. As long as our sprite rests in a clear part of the screen, address 15983 will contain a zero. If, when the sprite is moved, it happens to be in front of, or behind a character, then address 15983 will contain a 1. This makes collision detection a piece of cake!

Limitations

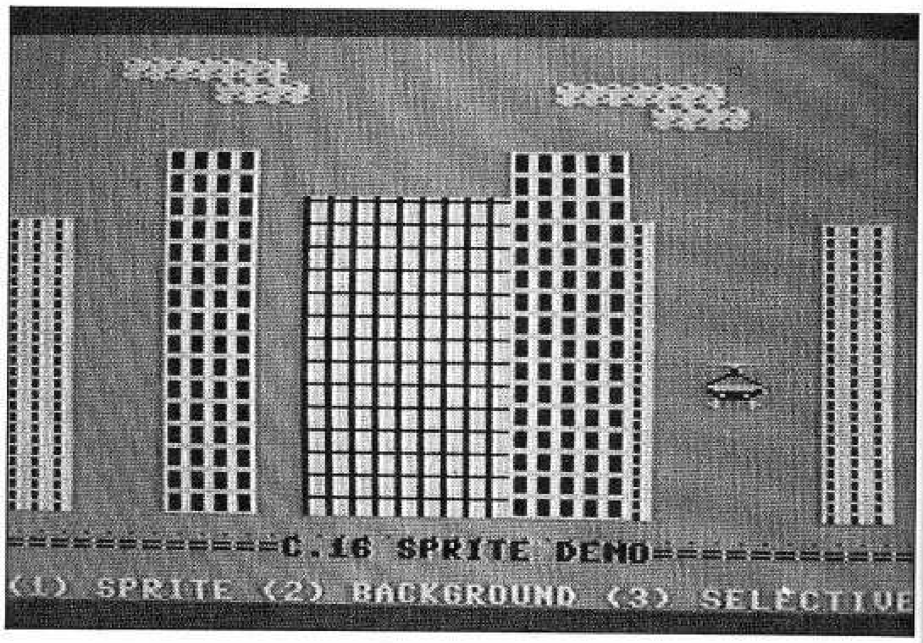
Using **C16 SPRITE** will not cause the screen to scroll, but your program may. If this happens then the sprite characters will also scroll. This is, unfortunately, a limitation of the system. Another is that the system updates the sprite only if any changes in the control addresses are detected; this is to avoid flickering. However, flickering may occur if, during animation techniques, the sprite pointer is changed rapidly when the sprite is near the bottom of the screen.

The Demonstration Program

Program 2 is the demonstration program which show **C16 SPRITE** in action. Guide the helicopter through the skyscrapers by the following keys:
Z Move left

X Move right
L Move up.
. Move down.
1 Move in front of buildings.
2 Move behind buildings
3 Move between buildings.
4 Collision mode.

Key 4 will toggle between collision mode and normal. If the sky turns grey then colliding with anything will cause a crash. Remember to have **C16 SPRITE** in memory before running the demo. Line 10 will automatically load it for you. If you are using tape, then **C16 SPRITE** must be saved immediately after the demo program as a 'boot load' file as described earlier. Also the device number '8' must be changed to a '1'.
Have fun!


============C.16 SPRITE DEMO===========
(1) SPRITE (2) BACKGROUND (3) SELECTIVE

```
PROGRAM: LISTING 1

0 POKE52,47:POKE56,47:CLR:CH=0
1 FORA=10TO940STEP10:READ H$
2 AD$=LEFT$(H$,4):AD=DEC(AD$):PR
INTAD
3 FORB=0TO7:N$(B)=MID$(H$,(3*B+5
),3):N=DEC(N$(B))
4 POKEAD+B,N:CH=CH+N:NEXTB,A
5 IFCH<>74307THENPRINT"DATA ERRO
R":ELSE PRINT"O.K."
6 STOP
10 DATA "3C00 A0 00 A2 28 B9 46
3E 99
20 DATA "3C08 00 0C B9 4C 3E 99
00 08
30 DATA "3C10 C8 C0 03 D0 EF B9
46 3E
40 DATA "3C18 9D 00 0C B9 4C 3E
9D 00
50 DATA "3C20 08 C8 E8 C0 06 D0
EE 60
60 DATA "3C28 AD 44 3E C9 00 10
01 60
70 DATA "3C30 C9 26 30 01 60 AD
45 3E
80 DATA "3C38 C9 00 10 01 60 C9
18 30
90 DATA "3C40 01 60 A9 01 8D 6C
3E 60
100 DATA "3C48 A0 00 8C 52 3E 8C
 53 3E
110 DATA "3C50 98 CC 45 3E F0 0C
 18 69
120 DATA "3C58 28 90 03 EE 53 3E
 C8 4C
130 DATA "3C60 51 3C A0 00 CC 44
 3E F0
140 DATA "3C68 0C 18 69 01 90 03
 EE 53
150 DATA "3C70 3E C8 4C 64 3C 8D
 52 3E
160 DATA "3C78 18 AD 53 3E 69 0C
 8D 53
170 DATA "3C80 3E AD 52 3E 8D 08
 3C 8D
180 DATA "3C88 19 3C 8D 0E 3C 8D
 1F 3C
190 DATA "3C90 8D F4 3C 8D FF 3C
 8D 2E
200 DATA "3C98 3D 8D 39 3D 8D 0F
 3D 8D
210 DATA "3CA0 1A 3D 8D 49 3D 8D
 54 3D
220 DATA "3CA8 AD 53 3E 8D 09 3C
 8D 1A
230 DATA "3CB0 3C 8D F5 3C 8D 00
 3D 8D
240 DATA "3CB8 2F 3D 8D 3A 3D 38
 E9 04
250 DATA "3CC0 8D 0F 3C 8D 20 3C
 8D 10
260 DATA "3CC8 3D 8D 1B 3D 8D 4A
 3D 8D
270 DATA "3CD0 55 3D 60 A0 00 B9
 46 3E
280 DATA "3CD8 99 60 3E C8 C0 0C
 D0 F5
290 DATA "3CE0 60 A0 00 B9 60 3E
 99 46
300 DATA "3CE8 3E C8 C0 0C D0 F5
 60 A0
310 DATA "3CF0 00 A2 28 B9 00 0C
 99 54
320 DATA "3CF8 3E C8 C0 03 D0 F5
 BD 00
330 DATA "3D00 0C 99 54 3E C8 E8
 C0 06
340 DATA "3D08 D0 F4 A0 00 A2 28
 B9 00
350 DATA "3D10 08 99 5A 3E C8 C0
 03 D0
360 DATA "3D18 F5 BD 00 08 99 5A
 3E C8
370 DATA "3D20 E8 C0 06 D0 F4 60
 A0 00
380 DATA "3D28 A2 28 B9 54 3E 99
 00 0C
390 DATA "3D30 C8 C0 03 D0 F5 B9
 54 3E
400 DATA "3D38 9D 00 0C C8 E8 C0
 06 D0
410 DATA "3D40 F4 A0 00 A2 28 B9
 5A 3E
420 DATA "3D48 99 00 08 C8 C0 03
 D0 F5
430 DATA "3D50 B9 5A 3E 9D 00 08
 C8 E8
440 DATA "3D58 C0 06 D0 F4 60 20
 D3 3C
450 DATA "3D60 A2 06 A0 00 B9 54
 3E C9
460 DATA "3D68 20 F0 09 99 46 3E
 8D 54
470 DATA "3D70 3E 9D 46 3E C8 E8
 C0 06
480 DATA "3D78 D0 EA 60 20 D3 3C
 A2 06
490 DATA "3D80 A0 00 B9 54 3E CD
 6D 3E
500 DATA "3D88 30 0E CD 6E 3E 10
 09 99
510 DATA "3D90 46 3E BD 54 3E 9D
 46 3E
520 DATA "3D98 C8 E8 C0 06 D0 E4
 60 A9
530 DATA "3DA0 00 8D 6F 3E A0 00
 B9 54
540 DATA "3DA8 3E C9 20 F0 06 A9
 01 8D
550 DATA "3DB0 6F 3E 60 C8 C0 06
 D0 EE
560 DATA "3DB8 60 A9 00 8D 6C 3E
 20 28
570 DATA "3DC0 3C AD 6C 3E F0 1F
 20 48
580 DATA "3DC8 3C 20 EF 3C 20 16
 3E 20
590 DATA "3DD0 D3 3C AD 6D 3E D0
 0F 20
600 DATA "3DD8 00 3C 20 9F 3D AD
 47 3E
610 DATA "3DE0 F0 03 20 E1 3C 60
 C9 FF
620 DATA "3DE8 D0 06 20 5D 3D 4C
 D7 3D
630 DATA "3DF0 20 78 3D 4C D7 3D
 A9 00
640 DATA "3DF8 8D 6C 3E 20 28 3C
 AD 6C
650 DATA "3E00 3E F0 12 AD 6D 3E
 F0 07
660 DATA "3E08 C9 FF F0 03 20 E1
 3C 20
670 DATA "3E10 26 3D 4C C6 3D 60
 AD 72
680 DATA "3E18 3E CD 73 3E F0 25
 8D 73
690 DATA "3E20 3E 18 A0 00 98 CC
 72 3E
700 DATA "3E28 F0 0A C0 14 F0 15
 69 0C
710 DATA "3E30 C8 4C 25 3E A8 A2
 00 B9
720 DATA "3E38 00 3F 9D 46 3E C8
 E8 E0
```

```
730 DATA "3E40 0C D0 F4 60 00 00
 00 00
740 DATA "3E48 00 00 00 00 00 00
 00 00
750 DATA "3E50 00 00 00 0C 20 20
 20 20
760 DATA "3E58 20 20 10 10 10 10
 10 10
770 DATA "3E60 00 00 00 00 00 00
 00 00
780 DATA "3E68 00 00 00 00 01 00
 5D 00
790 DATA "3E70 00 00 00 FF FF 78
 A9 00
800 DATA "3E78 8D 6C 3E 20 28 3C
 AD 6C
810 DATA "3E80 3E F0 0E 20 26 3D
 A9 0E
820 DATA "3E88 8D 14 03 A9 CE 8D
 15 03
830 DATA "3E90 58 60 00 00 00 78
 A9 A5
840 DATA "3E98 8D 14 03 A9 3E 8D
 15 03
850 DATA "3EA0 20 BE 3D 58 60 AD
 44 3E
860 DATA "3EA8 CD FB 3E D0 24 AD
 45 3E
870 DATA "3EB0 CD FC 3E D0 1C AD
 6D 3E
880 DATA "3EB8 CD FD 3E D0 14 AD
 72 3E
890 DATA "3EC0 CD 73 3E F0 09 20
 16 3E
900 DATA "3EC8 20 26 3D 20 B9 3D
 4C 0E
910 DATA "3ED0 CE 20 F6 3D AD 44
 3E 8D
920 DATA "3ED8 FB 3E AD 45 3E 8D
 FC 3E
930 DATA "3EE0 AD 6D 3E 8D FD 3E
 4C 0E
940 DATA "3EE8 CE 00 00 00 00 00
 00 00
```

PROGRAM: DEMO

```
10 IFC=0THENC=1:LOAD"C16. SPRITE
",8,1
20 POKE52,47:POKE56,47:CLR:VOL8
30 TRAP470:POKE1176,44:FORA=0TO5
11:POKE(12288+A),PEEK(53248+A):N
EXTA
40 FORA=12800TO13055:READD:POKEA
,D:NEXTA
50 FORA=0TO71:READD:POKE16128+A,
D:NEXTA
60 POKE65302,1:POKE65303,33:POKE
65305,65:POKE65301,93
70 POKE65287,PEEK(65287)OR16:POK
E65298,PEEK(65298)AND251:POKE652
99,48
80 A1$="[s +][s +][s +][s +][UP]
[LEFT][LEFT][LEFT][LEFT]":FORA=0
TO15:B$(1)=B$(1)
+A1$:NEXT
90 A1$="[c -][c -][c -][c -][UP]
[LEFT][LEFT][LEFT][LEFT]":FORA=0
TO13:B$(2)=B$(2)
+A1$:NEXT
100 A1$="[s -][s -][s -][UP][LEF
T][LEFT][LEFT]":FORA=0TO12:B$(3)
=B$(3)+A1$:NEXT
110 A1$="[255][255][255][UP][LEF
T][LEFT][LEFT]":FORA=0TO11:B$(4)
=B$(4)+A1$:NEXT
120 B$(5)="[c *][c *][c *][c *][
c *][c *][c *][DOWN][LEFT][LEFT]
[LEFT][c *][c *]
[c *][c *]":POKE1339,121:PRINTCH
R$(147)
130 FORA=1TO4:R=INT(RND(1)*36):C
HAR,R,20,B$(4):NEXTA
140 FORA=1TO4:R=INT(RND(1)*36):C
HAR,R,20,B$(3):NEXTA
150 FORA=1TO3:R=INT(RND(1)*35):C
HAR,R,20,B$(2):NEXTA
160 FORA=1TO3:R=INT(RND(1)*35):C
HAR,R,20,B$(1):NEXTA
170 CHAR,5,1,B$(5):CHAR,24,2,B$(
5)
180 CHAR,0,21,"[GREEN][c B][c B]
[c B][c B][c B][c B][c B][c B][c
 B][c B][c B][c
B][c B][c B][c B][c B][c B][c B]
[c B][c B][c B][c B][c B][c B][c
 B][c B][c B][c
B][c B][c B][c B][c B][c B][c B]
[c B][c B][c B][c B][c B][c B]"
190 CHAR,0,22,"[BLACK]----------
--C.16 SPRITE DEMO-------------"
200 CHAR,0,24,"[WHITE](1) SPRITE
 (2) BACKGROUND (3) SELECTIVE[HO
ME]"
210 X=0:Y=0:GOSUB380:SYSDEC("3E9
5"):S1=0:S2=1:CO=0:CR=0
220 GETA$
230 CO=CO+1:IFCO=3THENCO=0:S1=S1
+1:IFS1>S2THENS1=S1-2
240 IFA$="Z"THENX=X-1:GOSUB340:G
OSUB380
250 IFA$="X"THENX=X+1:GOSUB360:G
OSUB380
260 IFA$="L"THENY=Y-1:GOSUB380
270 IFA$=","THENY=Y+1:GOSUB380
280 IFA$="1"THENPOKE15981,0
290 IFA$="2"THENPOKE15981,255
300 IFA$="3"THENPOKE15981,91
310 IFA$="4"THENCR=CR+1:IFCR>1TH
ENCR=0:POKE65301,93
320 IFCR=1THENPOKE65301,65:GOTO4
30
330 POKE15986,S1:GOTO220
340 IFPEEK(15986)=4ORPEEK(15986)
=5THENX=X+1:S1=2:S2=3:RETURN
350 S1=0:S2=1:RETURN
360 IFPEEK(15986)=0ORPEEK(15986)
=1THENX=X-1:S1=2:S2=3:RETURN
370 S1=4:S2=5:RETURN
380 IFX<0THENX=0
390 IFX>37THENX=37
400 IFY<0THENY=0
410 IFY>19THENY=19
420 POKE15940,X:POKE15941,Y:RETU
RN
430 IFPEEK(15983)=0THEN330
440 FORA=1TO16:COLOR0,A:SOUND3,1
0*A,2:NEXTA
450 A$=CHR$(32):PRINT"[BLACK]":C
OLOR0,7,5
460 SYSDEC("3E75"):FORQ=1TO1000:
NEXTQ:GOTO120
470 POKE65287,PEEK(65287)AND239:
POKE65298,PEEK(65298)OR4:POKE652
99,208:STOP
480 DATA 255,0,0,0,2,8,32,128,25
5,192,128,128,160,168,170,175
490 DATA 192,0,0,0,15,63,254,250
,255,191,106,26,5,13,63,12
500 DATA 255,255,170,170,85,1,3,
0,232,160,128,64,64,192,240,192
510 DATA 63,0,0,0,0,2,8,32,255,4
8,32,32,168,2,0,0
520 DATA 240,0,0,0,0,0,128,32,63
,47,26,7,1,12,60,12
530 DATA 255,255,170,87,85,0,0,0
,240,224,144,64,0,192,240,192
540 DATA 3,0,0,0,240,252,191,175
,255,3,2,2,10,42,170,250
550 DATA 255,0,0,0,128,32,8,2,43
,10,2,1,1,3,15,3
560 DATA 255,255,170,170,85,64,1
92,0,255,254,169,164,80,112,252,
48
570 DATA 3,0,0,0,2,8,32,128,240,
192,128,128,160,168,170,175
580 DATA 0,0,0,0,15,63,254,250,3
,0,0,0,0,2,8,32
590 DATA 252,48,32,32,168,2,0,0,
0,0,0,0,0,0,128,32
600 DATA 0,0,0,0,240,252,191,175
,15,3,2,2,10,42,170,250
610 DATA 192,0,0,0,128,32,8,2,25
5,215,215,215,215,215,215,255
620 DATA 191,170,191,191,191,191
,191,191,255,223,223,223,255,223
,223,223
630 DATA 238,238,238,170,238,238
,238,170,60,207,255,255,63,252,2
55,60
640 DATA 64,65,66,67,68,69,121,1
21,121,121,121,121
650 DATA 82,83,84,67,68,69,121,1
21,121,121,121,121
660 DATA 70,71,72,73,74,75,121,1
21,121,121,121,121
670 DATA 85,86,87,73,74,75,121,1
21,121,121,121,121
680 DATA 76,77,78,79,80,81,121,1
21,121,121,121,121
690 DATA 88,89,90,79,80,81,121,1
21,121,121,121,121
```