

# AVON NEWS

VOL. 3. No. 6.



PAGE INDEX

- (1).....= FOR YOUR INFORMATION
- (2).....= ONE LINERS
- (3).....= MOUSE TOOLS by M.PUGH
- (4).....= " " continued
- (5).....= MACHINE CODE FOR BEGINNERS by J.NASH
- (6).....= " " " " continued
- (7).....= " " " " continued
- (8).....= " " " " continued
- (9).....= 16 BIT CORNER by B.SMITH
- (10).....= " " " " continued
- (11).....= PROGRAMMING IN BASIC - PART 6
- (12).....= " " " " continued

\*\*\*\*\*

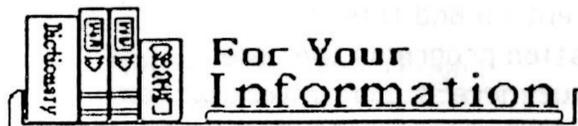
SUPPORT THE EINSTEIN

THE NEXT CLUB MEETING IS ON THE 13th of JULY

starting at 8.00 pm at the

BLACK HORSE, WEST STREET, OLD MARKET, BRISTOL

PLEASE TRY AND COME ALONG



Well it seems that not everybody has forgotten about the Einstein ?, two new books have been released by B & H Computers and they are :-

Albert Revealed by Crystal Research

Source by B & H

"Albert Revealed" was supposed to have come out about 2 years ago !, still it has finally made it and although I have't seen it myself I gather it is well worth getting if you wish to dive deeper into the inner workings of the Einstein. "Source" is also a technical specification manual but is more easy to understand. The price for "Albert Revealed" is £17.00 and the "Source" will cost you £16.00.

B & H have also brought out some new games for you to play and they are :-

Gloop, Escape from Merlin 8, Hobsons Game Pack and Theatre of Europe.

Once again contact B & H for more details.

Emsoft, the U.K.E.U.Groups software label has had a nose sorry a name change, it is now Taurus Computer Systems. Graham Bettany & Mike Smallman have gone into the computer business in a big way and are not only dealing in the Einstein but also in P.C's, they also have brought out two new programs and they are "EINJONG" & "HELPING HAND". At the moment I have no details on Einjong but to quote from their advert "Helping Hand is a machine code program featuring thirteen useful functions for BASIC programmers", the programs cover

- CROSS-REFERENCING by LINE NUMBERS, VARIABLES or FNs.
- Partial or complete RENUMBERING. DUPLICATION of blocks of BASIC statements with auto changing of labels.
- MERGING of other tokenised programs.
- List of DIFFERENCES between two tokenised programs.
- REMOVAL of REMarks, LETs or SPACES.
- LISTing (with referenced line numbers highlighted). SAVING of any amendments and LOADING of other tokenised programs.

The price for these programs are both £10.00 and more details can be got from T.C.S whose address is :- Enterprise House, Unit 15, Riverside Industrial Estate, Rapier Street, Ipswich. Tel.No. 0473/602460.

Also a new label has been brought to my attention and this is "EVANGELSOFT", this label has several christian programs that are available for the Einstein and you can find out more if you contact Dr Ken Dean, 5 Tynamara, 20 Portsmouth Road, Kingston-on-Thames, KT1 2NG. Tel.No. 01/549/2280.

Whilst I am on the information page, don't forget that you people with modems or are thinking of getting one, the Einstein Bulletin Board can be reached on 0752/848806, with your modem set at 300/300 or 1275, 8 bits No parity, 1 stop bit. Chris Conyon is the Sysop and I'm sure he will be pleased to hear from you.

Those members that are following John Nash's "Machine Code for Beginners" will have noticed that he uses an assembler called ZASM, this program will only run under ZDOS or CP/M 2.2 but now a man called Cecil Wallis has converted it to run under XTAL DOS and has called it NEWZASM.COM, if you would like this new version then I will get our Librarian to get a copy of it from Jim Ellacott who is the librarian of the U.K.E.U.G.

Please let me know !!!.

And now just to see if you are still awake!, here are some more one-liners. Don't forget if you want to get all the characters on one line you will have to extend the line length, this is done when you have loaded XBAS by typing :- CLEAR &E00:PTR 12,&E00:PTR 13,249 <ENTER>

These one-liners are in XBAS 4.2 and are best viewed on a colour monitor.

Graham Bettany

```
1 CLS:BCOLO:ORIGIN99,85:A=3:B=6:C=8:FORC=2TO15:FORI=0TO250:XX=Y-
SGN(X)*SQR(ABS(B*X-C)):YY=A-X:X=XX:Y=YY:PLOTX*5,Y*5:GCOLC:NEXT:NEXT
```

Dave Harvey

```
1 CLS:V=.8+RND(1)*3:FORT=0TO200*VSTEPV:R=T/V:X=R*COS(T):Y=R*SIN(T):
DRAW125,98TO125+X,98+Y:NEXT:FORP=1TO999:NEXT:GOTO1
```

Steve Cooper

```
1 BCOL1:R=80:B=125:C=95:FORL=1TO6STEP.5:CLS:GCOLL+7:FORA=0TO2*PI
STEP PI/C:X=R*COS(A)+B:Y=R*SIN(L*A)+C:DRAWB,CTOX,Y:NEXTA,L
```

REPRODUCED WITH THE KIND PERMISSION OF THE U.K.E.U.G.



### MOUSE TOOLS by M.Pugh

The following long (very long!) listing is an early version of a program I have just released for sale through Taurus Computer Systems (formerly EMSOFT).

This version really is my first attempt at this sort of programming but works well. It is basically a Directory Editor The program is mouse driven ONLY and is very easy to use. You just move the pointer around the screen and then click the mouse button on top of any Icon. The Icons represent disk drives A - D, A & B being 3" 188k drives and C & D being 799k 5.25" drives.

The program allows you to perform the GLOBAL directory functions as well as DIR and EXIT to DOS.

ERASE ALL FILES

UN-ERASE ALL FILES

LOCK ALL FILES

UN-LOCK ALL FILES

MAKE ALL FILES SYSTEM FILES (HIDE THEM FROM DIR) RETURN ALL SYSTEM FILES TO NORMAL

The program as well as being very useful is also very easy to use but WILL ONLY WORK WITH THE MOUSE. So those of you who don't have MOUSE ART don't bother typing it in, IT WON'T WORK !!!!.

As I said at the beginning, this was Version.1. of my program. I have since finished Version.3. which is much better and also allows the selective file erase / lock / unlock facility. It has a top line status menu with pull-down Icons. It is also very friendly and can be operated by either MOUSE, JOYSTICK, and the KEYBOARD so you have no excuse to buy it now, HAVE YOU ?.

Version.3. is written in compiled BASIC and runs extremely fast. Also provided on side B. of the disk is a very useful DISK FORMATTER program which will enable you to format your 3" disks to either 40 track 190k, 41 track 194k or 42 track 202k. Modified versions of DOS 1.31 are provided so full use of the extra disk space can be used. I have also provided 2 BACKUP programs which will backup the disk with 41 or 42 tracks from drive A to drive B, as your normal BACKUP program can only copy 40 tracks. The disk formatter also TURBO formats the disk with the best possible interleave factor and increases the disk access time using DOS 1.31 by 50% to 100%.

EXAMPLE. 50k program takes 25 seconds to load using DOS 1.31. and formatted normally using BACKUP ver.1.

50k program takes 12 seconds to load using DOS 1.31. and formatted using FORMAT+.

This sort of disk access speed is normally possible using SYSTEM 5 operating system costing £40.00.

All the above programs are supplied on a single 3" disk for the measly sum of £10.00. Copies are available NOW!, from either Taurus or myself.

M.Pugh G4YPD, 37 FOREST WAY, HOLLYWOOD, BIRMINGHAM, B47 5JS,  
Tel.No.0564/823966 after 7pm.

So don't delay! send today!

Disk contains....

RUNTIME.COM	(Runtime routines for OVL files)
MENU.OVL	(Select mouse, joystick or keyboard)
MTOOLS.OVL	(mouse driver)
JTOOLS.OVL	(joystick driver)
KTOOLS.OVL	(keyboard driver)
FORMAT+.COM	(formats 40,41,42 or 80 tracks)
BACKUP41.COM	(make backups of 41 track disks)
BACKUP42.COM	(make backups of 42 track disks)
41TRACK.DOS	(modified DOS to access 41 track 194k)
42TRACK.DOS	(modified DOS to access 42 track 202k)
80TRACK.DOS	(modified DOS for SS 80t 5.25 drives)
READ.ME	(copyright message and instructions for transfer of DOS files to system tracks)

Now follows the listing of MOUSE TOOLS version.1.

EDITORS NOTE.....

Unfortunately this listing is so long that to include it in the magazine it would have taken most of the pages up. If you wish to have a copy of the listing then please send me a 9" X 4" stamped address envelope and I will send you the listing. My apologies to Mick but I think that for the price of £10 for version.3. and with all the other programs he is offering it is better to buy the disk than include the listing in the magazine.

CROMEMCO Z80 Macro Assembler version 03.04  
Source File: TUTGAM1

```

0001 ;
0002 ;
0003 ;
0004 ; This month we are going to play around in the VRAM. To do this we
0005 ; need to know the addresses of where some of it's occupants live. In this
0006 ; program we are interested in the sprite's. There is in area of memory in
0007 ; the VRAM called the sprite attribute table. This table is 128 bytes long
0008 ; and starts at 3800 hex see appendix D on page 324 of the Einstein Basic
0009 ; Reference Manual.
0010 ; So now we know where it lives what does it look like. Well there
0011 ; are 32 sprites so 128/32 = 4 , so now we know that we have only 4 bytes
0012 ; for each sprite. These 4 bytes are as follows 1st hold the Y co-ordinate,
0013 ; 2nd holds the X co-ordinate, 3rd holds the shape number, and finally the
0014 ; 4th holds the colour number.
0015 ;
0016 ; ADDRESS : SPRITE No. : BYTE 1 : BYTE 2 : BYTE 3 : BYTE 4 :
0017 ; 3800 : 0 : Y co-ord : X co-ord : shape No. : Colour NO. :
0018 ; 3804 : 1 : Y co-ord : X co-ord : shape No. : Colour No. :
0019 ; 3808 : 2 : Y co-ord : X co-ord : shape No. : Colour No. :
0020 ; And so on for all 32 sprites.
0021 ;
0022 ; I will not go into detail on how to make the sprite shape as you have
0023 ; all most likely have done this in BASIC. What we are going to do in this
0024 ; routine is to define the first 4 sprite's and position them on the screen to
0025 ; make one big sprite. We will then do a little routine to move it from left to
0026 ; right using the D and W keys. The space bar key will allow you to escape from
0027 ; the routine. This routine is done this way to use more instruction and MCAL's.
0028 ; There are much faster and sophisticated ways of reading the keys and moving
0029 ; the sprites but lets keep it simple.
0030 ;
0031 ;
0032 ; (00BE) C1rall EQU 0BEh ; MCAL BE this MCAL does the following.
0033 ; ; 1... Clears the screen to 40 col.
0034 ; ; 2... Resets all characters.
0035 ; ; 3... Removes sprites.
0036 ; ; 4... Resets FDC ( floppy disc controller )
0037 ; ; Resets PSG ( programmable sound generator )
0038 ; ; 5... Masks the keyboard, fire and ADC interrupts.
0039 ; (3800) Sprt1 EQU 3800h ; VRAM address of sprite No.1 attributes.
0040 ; (3804) Sprt2 EQU 3804h ; DITTO sprite No.2 attributes.
0041 ; (3808) Sprt3 EQU 3808h ; DITTO sprite No.3 attributes.
0042 ; (380C) Sprt4 EQU 380Ch ; DITTO sprite No.4 attributes.
0043 ; (3801) Xpos1 EQU 3801h ; VRAM address of X co-ord sprite No.1.
0044 ;
0045 ;
0046 ; (0100) DRG 0100h ; ORGanise to run from 0100 hex.
0047 ;
0048 ; 0100 CF 0048 RST 0Bh ; Tell processor to expect a MCAL.
0049 ; 0101 BE 0049 DEFB C1rall ; MCAL BE see above.
0050 ; 0102 010B1D 0050 LD BC,7432 ; Put VRAM address of the character No.145 into
0051 ; ; the BC register.
0052 ; 0105 CF 0052 RST 0Bh ; Tell the processor to expect a MCAL.
0053 ; 0106 C1 0053 DEFB 0C1h ; MCAL C1 send bytes held in the A reg to VRAM address
0054 ; ; held in the BC reg.
0055 ; 0107 119F01 0055 LD DE,Shape ; Load the DE reg with the address of the data for

```

CROMEMCO Z80 Macro Assembler version 03.04  
Source File: TUTGAM1

```

010A 3E20      0056      LD A,20h      ; shape for the sprite.
              0057      ; Load the A reg with 32 the number of bytes to poke
              0058      ; into the VRAM.
010C F5       0059      LoopA: PUSH AF ; PUSH the A reg onto the stack to save its value.
010D 1A       0060      LD A,(DE)     ; Load the A reg with the contents of the DE reg
              0061      ; in BASIC A=PEEK(DE).
010E D308     0062      OUT (B),A    ; Put the value of A through PORT B
0110 13       0063      INC DE       ; Move on to the next byte in the Shape data.
0111 F1       0064      POP AF       ; Get the value of A back off the stack.
0112 D601     0065      SUB 1        ; SUBtract 1 from A ( A=A-1 ).
0114 20F6     0066      JR NZ,LoopA ; Jump Relative the No. of bytes to the address
              0067      ; at LoopA: if A is Not Zero. If A is Zero then
              0068      ; continue with the program.
0116 01003B   0069      LD BC,Sprt1 ; Put VRAM address of the start of sprite attribute
              0070      ; table into the BC register.
0119 CF       0071      RST 08h     ; Tell processor to expect a MCAL.
011A C1       0072      DEFB 0C1h  ; MCAL C1 write to VRAM.
011B 112F01   0073      LD DE,Spdata ; Load the DE reg with the address where the DATA
              0074      ; bytes for the sprites are placed. ( See below ).
011E 3E10     0075      LD A,10h    ; Load the A reg with 16 as a counter to put the 16
              0076      ; bytes at Spdata into the VRAM.
0120 F5       0077      LoopB: PUSH AF ; PUSH the A reg onto the stack to keep it safe.
0121 1A       0078      LD A,(DE)     ; Load the A reg with the contents of the DE reg
              0079      ; in BASIC A=PEEK(DE).
0122 D308     0080      OUT (B),A    ; Put the value of A through PORT B.
0124 13       0081      INC DE       ; Move on to the next byte in Spdata.
0125 F1       0082      POP AF       ; Get the value of A back off the stack.
0126 D601     0083      SUB 1        ; SUBtract 1 from A ( A=A-1 ).
0128 20F6     0084      JR NZ,LoopB ; Jump Relative the No. of bytes to the address
              0085      ; at LoopB: if A is Not Zero. If A is Zero then
              0086      ; continue with the program.
012A CF       0087      Keys: RST 08h ; Tell the processor to expect a MCAL
012B 9B       0088      DEFB 9Bh    ; MCAL 9B this MCAL scans the keyboard and returns
              0089      ; the ASCII value of any key pressed in the A reg.
              0090      ; This MCAL does not wait for a key to be pressed
              0091      ; it works like the KBD command in BASIC.
012C FE51     0092      CP 51h      ; This instruction CP means ComPare. What it does is
              0093      ; subtract the value ( in this case 51h / 81 dec ) from
              0094      ; the A reg WITHOUT altering the A reg. The answer can
              0095      ; be one of three results positive, negative, or zero.
              0096      ; We are going to be looking for ZERO. The ASCII CODE
              0097      ; for the character Q is 51h/81dec. So if the key Q is
              0098      ; pressed the A reg will hold 51h if we then CP 51h the
              0099      ; result will be ZERO.
012E CA3E01   R 0101      JP Z,Left    ; The Jump is like GOTO in BASIC. This time we
              0102      ; have an argument following the JP and that is Z
              0103      ; meaning Zero. So the op we are using is JP ZERO.
              0104      ; Now if key Q had been pressed the A reg would hold
              0105      ; 51h the CP 51h would have resulted in ZERO the JP Z
              0106      ; would have been active and the program would have
              0107      ; jumped to the address the assembler puts Left at.
              0108      ; If the the result had not been ZERO the processor
              0109      ; would have ignored the JP Z op.
0131 FE57     0110      CP 57h      ; This time we are doing a CP on the W key.

```

CROMEMCO Z80 Macro Assembler version 03.04  
Source File: TUTGAM1

```

0133 CA6601   R 0111      JP Z,Right   ; This time if the key W is pressed we are going to
              0112      ; a sub routine at the address Right.
0136 FE20     0113      CP 20h      ; This time we are doing a CP on the space bar.
0138 C8E01    R 0114      JP Z,Fire    ; This time if the space bar is pressed it will go to
              0115      ; sub routine Fire.
013B C32A01   R 0116      JP Keys      ; Go back to the beginning of the key detect routine
              0117      ; and look again.
013E 01013B   0118      Left: LD BC,Xpos1 ; Get VRAM address of sprite No.1 X co-ord into BC.
0141 CF       0119      RST 08h     ; Tell processor to expect a MCAL.
0142 C2       0120      DEFB 0C2h  ; MCAL to return the value in the A reg what is in the
              0121      ; VRAM address contained in the BC reg.
0143 FE00     0122      CP 00h      ; see if its ZERO. If it is the it is at the left
              0123      ; edge of the screen and we wont want it to move any
              0124      ; more.
0145 CA2A01   R 0125      JP Z,Keys    ; Jump to key routine if result is ZERO.
0148 3D       0126      DEC A       ; A=A-1 ( we want to move one pixel to the left ).
0149 CF       0127      RST 08h     ; Tell processor to expect a MCAL.
014A C3       0128      DEFB 0C3h  ; MCAL C3h this MCAL is the opposite to C2h in as much
              0129      ; that it writes the value in A REG to the VRAM address
              0130      ; held in the BC reg.
              0131      ; Now you can see we have took the X co-ord of sprite
              0132      ; No.1 checked to see that it is not zero ( which means
              0133      ; that it would have been at the far left side of the
              0134      ; screen ). If it was we would return without moving
              0135      ; the sprite. If it was greater than zero we moved it
              0136      ; one pixel to the left. Lets now move the other three
              0137      ; sprites one pixel to the left.
014B 01053B   0138      LD BC,Xpos1+4 ; Point the BC reg to the address of the X co-ord of
              0139      ; sprite No.2.
014E CF       0140      RST 08h     ; Tell processor to expect a MCAL.
014F C2       0141      DEFB 0C2h  ; MCAL C2h as above.
0150 3D       0142      DEC A       ; Move sprite one pixel left
0151 CF       0143      RST 08h     ; Tell processor to expect a MCAL.
0152 C3       0144      DEFB 0C3h  ; MCAL C3h as above.
0153 01093B   0145      LD BC,Xpos1+8 ; Point the BC reg to the address of the X co-ord of
              0146      ; sprite No.3.
0155 CF       0147      RST 08h     ; Another MCAL
0157 C2       0148      DEFB 0C2h  ; MCAL C2h as above.
0158 3D       0149      DEC A       ; Move sprite one pixel left.
0159 CF       0150      RST 08h     ; Another MCAL
015A C3       0151      DEFB 0C3h  ; MCAL C3h as above.
015B 010D3B   0152      LD BC,Xpos1+12 ; Point the BC reg to the address of the X co-ord of
              0153      ; sprite No.4
015E CF       0154      RST 08h     ; Another MCAL
015F C2       0155      DEFB 0C2h  ; MCAL C2h as above.
0160 3D       0156      DEC A       ; Move sprite one pixel left.
0161 CF       0157      RST 08h     ; Another MCAL
0162 C3       0158      DEFB 0C3h  ; MCAL C3h as above.
0163 C32A01   R 0159      JP Keys      ; Jump Relative to KEY routine.
0166 01013B   0160      Right: LD BC,Xpos1 ; Move the sprites to the right is identical as the
              0161      ; move to the left but this time we must INCrease the
              0162      ; A reg to move the X co-ord one place to the right.
              0163      ; Another difference is this time we must do a ComPare
              0164      ; 240 dec to see if the sprite is at the far right hand
              0165      ; side of the screen.

```

CROMEMCO Z80 Macro Assembler version 03.04  
Source File: TUTGAM1

```

0169 CF      0166      RST 08h
016A C2      0167      DEFB 0C2h
016B FEFO    0168      CP 0F0h
016D CA2A01  R 0169      JP Z,Keys
0170 3C      0170      INC A
0171 CF      0171      RST 08h
0172 C3      0172      DEFB 0C3h
0173 01053B  0173      LD BC,Xpos1+4
0176 CF      0174      RST 08h
0177 C2      0175      DEFB 0C2h
0178 3C      0176      INC A
0179 CF      0177      RST 08h
017A C3      0178      DEFB 0C3h
017B 01093B  0179      LD BC,Xpos1+8
017E CF      0180      RST 08h
017F C2      0181      DEFB 0C2h
0180 3C      0182      INC A
0181 CF      0183      RST 08h
0182 C3      0184      DEFB 0C3h
0183 010D3B  0185      LD BC,Xpos1+12
0186 CF      0186      RST 08h
0187 C2      0187      DEFB 0C2h
0189 3C      0188      INC A
0189 CF      01C9      RST 08h
018A C3      0190      DEFB 0C3h
018B C32A01  R 0191      JP Keys
018E C9      0192      Fire: RET
                0193
                0194
                0195
018F AF7BA101  0196      Spdata: DEFB      0AFh,78h,0A1h,01h      ; Sprite No.1
0192 AF80A201  0197      DEFB      0AFh,80h,0A2h,01h      ; Sprite No.2
0197 B77BA201  0198      DEFB      0B7h,78h,0A3h,01h      ; Sprite No.3
0198 B780A401  0199      DEFB      0B7h,80h,0A4h,01h      ; Sprite No.4
                0200
019F 01010113 0201      Shape: DEFB      01h,01h,01h,13h,17h,1Fh,1Fh,3Fh      ; Sprite No.1
01A7 808080CB 0202      DEFB      80h,80h,80h,0C8h,0E8h,0F8h,0F8h,0FCh      ; Sprite No.2
01AF 7FFFFFFD5 0203      DEFB      7Fh,0Fh,0Fh,0D5h,0C9h,0C9h,0C1h,0C1h      ; Sprite No.3
01B7 FEFFFFAB 0204      DEFB      0Fh,0Fh,0Fh,0ABh,93h,93h,83h,83h      ; Sprite No.4

Errors      0
Range Count 8

```

## 16 BIT CORNER

One of the problems of upgrading to a different computer system is that you loose all the software you have acquired so that you have to start all over again.

As I do a lot of writing (some people call it waffle), the first item to get was a word processor and to get one without paying hundreds of pounds so I started to look at what there was in the Public Domain. Mike Maddock come up with one for me which is called "PCWRITE Version 2.7", this is a very powerful word processor and comes with its own dictionary, on-screen help file and two manuals that can be printed out but be sure to have plenty of paper ready as one manual is a tutorial and is 17 pages long and the other is a quick guide and is 44 pages long and you can also get a book from the library called "The Shareware Book".

When I first ran the program, what I wrote on the screen did not always match what was printed out on the printer. This was not the programs fault but mine for not setting up the printer.

When I had the Einstein the printer was setup to emulate the Epson FX80 but when I got the IBM, I forgot that I should of setup the printer in IBM mode, this was soon sorted out so that what is printed on the screen is the same as what comes out on the printer.

The next piece of software I required was some form of Desk Top Publisher and the one I use is called FIRST PUBLISHER by Software Publishing and will cost you £85.00 plus VAT. This is a very easy one to start with and you can get an Art Gallery to go with the program for £49.00 plus VAT. The graphics come in what they call MAC format, that means the extension is .MAC ie.LEISURE.MAC and in the Public Domain there are a lot of disks you can get with this extension or you can get a program that will convert other extensions to MAC format ie. PCX to MAC.

Looking for a suitable place to get your Public Domain disks from is the hardest thing you have to do, prices range from £1.50 per disk up to £5.00 per disk and you have to read the small print to make sure that you don't have to add on the dreaded VAT & postage.

I eventually found a firm called SOFTCELL SERVICES which is based in Cardiff. They charge £1.75 per disk including VAT & Firstclass return postage and the disks are sent to you within 24 hours of them receiving

your order infact the service you get from them is absolutely brilliant. The following is a list of the P.D.software I have at the moment and if you want a copy of it then send me £1.50 per disk (5.25") or £2.50 per disk (3.5") (Please make cheques/postal orders to :- M. J. SMITH)

**THESE PRICES INCLUDE THE DISK**

(DON'T FORGET THIS SOFTWARE IS FOR IBM & COMPATIBLES ONLY).

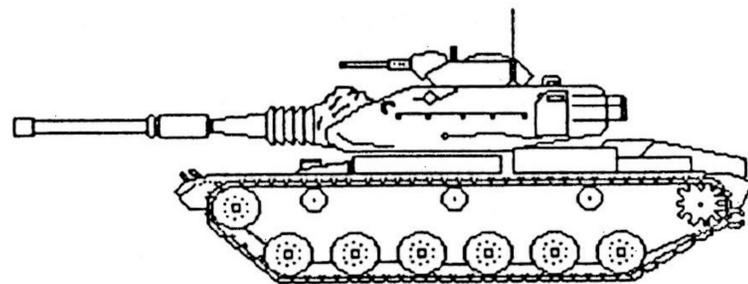
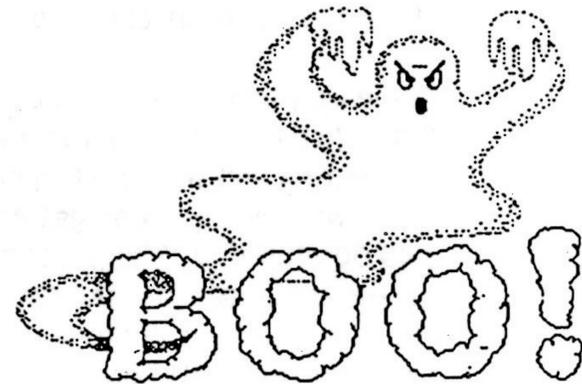
MAC ART DISKS	(2 DISK SET)
PCPAINTBRUSH CLIP ART	(1 DISK)
I CONVERT	(1 DISK)
ZEBRA GIRL	(1 DISK)
PINUPS	(1 DISK)
PCWRITE ver.2.7	(2 DISK)

If you want to get your own P.D.software catalogue then write or phone :-  
SOFTCELL SERVICES, 9 WELLS STREET, RIVERSIDE, CARDIFF, CF1 8DW  
Tel.No. 0222/222895

First Publisher can be bought from :-

CompuAdd, CLIFT HOUSE ROAD, ASHTON GATE, BRISTOL, BS3 1RX.  
Freephone 0800/373535

Below are just some examples of the kind of graphics you can reproduce



Programming in BASIC for Beginners - Part 6.

INPUTS

Displaying information contained within the program is all very well but virtually all computer programs require that the user INPUTs some kind of information. In a game situation this can involve moving a joystick or pressing a fire button. Other applications require that the user selects an option from a menu, or in the case of a database, the user must type in information to be stored and processed. We must now examine some of the ways that the computer can get and store information from the user.

All information obtained from the user when the program is running (RUNTIME), is stored in what is known as VARIABLES. Variables are names used to represent values. These values can be assigned by the programmer or be a result of calculations or operations from within the program or as we have already said, can be INPUT by the user at Run-Time. Prior to being given a value a variable is assumed to be a zero.

There are three types of variables and they are indicated by the character which suffix the variable name.

A NUMERIC FLOATING POINT VARIABLE has no suffix after its name and holds numbers. Example :- ABC

A NUMERIC INTEGER VARIABLE is indicated by the % sign after its name and holds WHOLE numbers. Example :- ABC%

A STRING VARIABLE is indicated by a \$ sign after its name and holds combination or strings of characters. Example :- ABC\$

A variable name is devised by the programmer and can be a combination of letters or letters/numbers but the first character must always be a letter. XTAL BASIC recognises the first five characters only of the variable name although more can be entered. It must be remembered that characters after the fifth will be ignored by BASIC. Care must be taken to ensure that the variable name is not and does not contain those words which are recognised by the computer and make up the BASIC language (RESERVED WORDS). Example :- TABLE, Table cannot be used as a variable name because it contains the RESERVED word TAB. Now let us suppose that we are writing a program that needs to know a particular time. We will use a VARIABLE to hold the information concerning the time and since that value will always be INTEGERS we will use a Numeric

Integer Variable.

Key in and RUN the following program :-

```
10 CLS 20 TIME%=1000 30 PRINT TIME% 40 END
```

It can be seen that this program puts the value of 1000 into the Integer Variable TIME% (line 20), line 30 then prints the value of TIME% on the screen.

Taking this a step further, let us suppose that the programmer requires the user to ENTER the time from the keyboard. One of the commands to allow the user to ENTER information from the keyboard is INPUT. The use of INPUT causes execution of the program to stop, allowing the user to insert information. The program will not continue until the ENTER key is depressed and whatever information has been ENTERed will be stored in the VARIABLE name which follows the INPUT command.

If we now change our line 20 to read :-

```
20 INPUT TIME%
```

and RUN the program we will find that a prompt in the shape of a Question Mark (?) is displayed. Nothing further will happen until the ENTER key is depressed and then line 30 will print the data ENTERed by the user at line 20.

On some applications it is not desirable to have the ? prompt displayed. This can be suppressed by the addition of two inverted commas and a semi colon between the INPUT command and the variable name, e.g. INPUT";TIME%

It is good practice to display a message immediately before an INPUT in order to tell the user what information is expected from him. This could be done with a PRINT statement before the INPUT command but this is not necessary because the INPUT command has a built-in PRINT command which can be used if the programmer so wishes.

Example :-

```
20 INPUT"Please Enter the Time using the 24hr clock ";TIME%
```

When RUN our program will ask the user for the information expected.

AVON EINSTEIN USER GROUP COMMITTEE MEMBERS

CHAIRMAN .....MIKE IVORY, 1 HEATH ROAD, HANHAM, BRISTOL, BS15 3JT.  
TELEPHONE No. 0272-616281

=====

SECRETARY .....JOHN NASH, 38 BURFOOT GARDENS, STOCKWOOD, BRISTOL, BS14 8TE  
TELEPHONE No. 0272-838028

=====

TREASURER .....ROLF GEORGE, 25 FAIR VIEW, CHEPSTOW, GWENT, NP6 5BX.  
TELEPHONE No. 02912-4916

=====

LIBRARIAN .....GRAHAM HIGGINS, 31 LENA STREET, EASTON, BRISTOL, BS5 6BD  
TELEPHONE No. 0272-559874

=====

EDITOR .....BOB SMITH, 2 INGLETON DRIVE, WORLE, WESTON-SUPER-MARE, AVON,  
BS22 0SR. TELEPHONE No. 0934-517465

=====

Any articles, listings, reviews, questions, complaints !! please send them to the Editor

For all Public Domain enquiries !! please contact the Librarian.

=====

PLEASE REMEMBER IF YOU ARE SENDING DISKS, AND YOU WANT THEM RETURNED ?

THEN ENCLOSE AT LEAST 28p RETURN POSTAGE

(Return address label if possible)

Please Note....All articles and programs become the copyright of the A.E.U.G. as well as the original authors !!!.

Front cover was designed by Bob Smith with the help of First Publisher and

printed on a Olivetti DM280 dot-matrix printer

Don't forget to water the plants

if you are going on holiday

