16. Anagram

General description

This program runs on a BBC Model 'B', but without REMs and restricted amount of data it may run on a Model 'A'. However, this has not been tested.

The program generates anagrams from given data, randomly accessing the data and jumbling the letters. It invites a response from the child and checks for certain letter combinations. Those it considers most unlikely it marks with an arrow (^). There are limited sound effects to hold the attention of the player.

It finally invites the player to judge whether the word made is an 'English' word. Deliberately, there are no facilities to rub-out errors and all attempts made by the player are saved for display to the teacher when the program finishes. The teacher is needed to correct and instruct the player when the 'game' has finished and the final display is marked with '?'s against words the computer thinks suspect and with '*'s against those that the player thinks suspect.

In theory, progress on from the display page is controlled by the teacher who will need to type in the '{character. The escape key also follows progress.

You are first presented with an introduction page. This is followed by letters being 'beeped' across the screen for the anagram. The escape key is used to give up on the anagram and progress to the menu page, whence the game can be finished, replayed or results displayed.

It is important to read the educational notes before typing in the program as several decisions must be taken concerning the data. The game seems to hold the attention of players for about 15 minutes.

Lines 1-34 Main structure of the program. The final block of program is jumped to at line 224, otherwise the program is really a simple series of steps through procedures. Certain variables have to be initialised.

35-51 All jumbling of the letters is done on string handling routines. The only array used in the program is to hold the words created by the player. WORD$ is the unmixed word and NEWWORD$ is the anagram.

52-77 This routine marches the letters of the anagram across the screen to a respectable beeping sound. The small delay is to give each letter time to sound its ' beep'Play with the sound if you like.

78-119 This is the heart of the checking section of the program. The educational section of the notes will give you a guide on the data to be included. If you change the data OR renumber the program - beware . . . the program uses a computed RESTORE to point to the correct data line. At the moment the last number in line 86 is a ' 1' This is the line renumber interval. If you renumber with gaps of ten, then change this to ' 10'Similarly, in line 87 the current pointer to the start of the data is ' 94'Any renumber or playing with data may change this. Finally when you change the data remember to finish each line with a ' 9' . This acts as a data terminator.

120-134 This routine checks to see whether the letter chosen is in the anagram. If it is not, the selected letter will generate no screen response. There is also a check that the same letter in the anagram is not used twice. This is done by substituting the character ' 9' into the word once the letter in its position has been used.

135-167 This routine itself calls a number of other procedures. CONWORD$ is a parameter passed between procedures via INWORD$. The last letter pair used is created and checked against letter pairs in the routine above. The player's word is stored for later display with a flag (SIGN) set if the computer does not like the letter combination.

168-181 This routine, called from the one above, determines if the last letter entered (TEST$) was a vowel. A flag (HFLAG) is set if a vowel is found.

182-206 As the introduction is written in MODE 7 the CHR$

refers to the control codes required when working in MODE 7. You are welcome to alter any features at the beginning to suit your needs.

207-222 This simply generates the word to be jumbled (WORD$). The data is accessed randomly so the number in brackets in line 215 must contain a value that is the same as the number of words in the data.

223-248 This displays the end of the game menu. Notice that it is written in MODE 7 with all the above notes on control codes for this MODE. Change the envelope if you do not think that it suits you.

250-265 This routine is called from the menu. Return to the menu should be via the ' {character but I have not trapped the escape key here and this will return to the menu as well . . . not good practice.

266-276 This allow the player to decide if the word is an English word or not and adds a suitable marker ' *' at the end of the word if the player rejects the word made.

Educational Notes

It is very important that these are read carefully before typing in the program and the DATA in particular. The DATA supplied is test data to show that the program works, but will not provide the best use of the program. The teacher using the program must decide (i) the number of words available to be mixed up, (ii) the words themselves and (iii) the letter pairs that will be allowed.

The words themselves must be chosen with care. You may wish to concentrate on the letter pairs ' ST" ,CT" ,LT' It would then be advisable to choose short words for the data which include these letters, and other letters when combined with S, T etc. would not be acceptable. Take e.g. ' BOAST' By limiting the acceptable letter pairs you will test for ' BT" ,BS" ,SB' etc. The longer words provide greater problems for computer analysis, as the chances of three consonants being put together and being accepted by the computer increase. E.g.: ' SST' will currently pass the pairs test as ' SS' and ' ST' both separately pass the test. It is up to the user to decide whether to write a three letter check procedure, but I feel that back checking has to finish arbitrarily anyway, so I have stopped at letter pairs. If the teacher wishes to

avoid this problem, the anagram words must again be chosen with care.

It is partly for this reason, and the educational advantages of making the player decide whether the word he has used is an ' English' word for himself, that such a routine has been included.

I suggest that you keep several copies of the program with different sets of data or spool the data in from tape if you wish to adapt the program in that manner. In this way you can develop a structured series of programs which in themselves will form a learning program.

I will of course be pleased to hear in detail of any program of learning that is developed. If you don' tfollow this idea, the program will stand alone - but will not be as useful as it could be.

Program Listing

```
 1 REM *************************
 2 REM *                       *
 3 REM * THE ANAGRAM PROGRAM !  *
 4 REM * adapted by  I Murray   *
 5 REM *       1983             *
 6 REM *                       *
 7 REM *************************
 8 REM
 9 REM ============================
10 REM =                           =
11 REM =  main structure           =
12 REM =                           =
13 REM ============================
14 REM
16 REMON ERROR GOTO  224
17 MODE 7:SIGN=0
18 PROCstart
19 CLS : MODE 5 :CLEAR:DIM KWORD$(61):SIGN=0
20 FFLAG=0:KW=1:KWORD$(1)=""
21 PROCgetword
22 LL=LEN(WORD$)
23 PROCjumble (WORD$)
24 PROCdisplay
25 PROCgetin
26 IF SIGN THEN KWORD$(KW) = KWORD$(KW) + " ??"
27 PROCchoose
28 SIGN = 0
29 KW = KW + 1
30 IF KW=60 GOTO 224
31 VDU 31,1,10
32 PRINT SPC (400) " "
33 VDU 31,1,12:PRINT SPC(200) " ":GOTO25
34 END
35 REM ============================
36 REM procedure to jumble letters
37 REM ..........................
38 REM
39 DEF PROCjumble (WORD$)
```

```
 40 TEMP$=""
 41 LOCAL Y,X
 42 REPEAT
 43   X=LEN(WORD$)
 44   Y=RND(X)
 45   TEMP$ = TEMP$ + MID$(WORD$,Y,1)
 46   L$=LEFT$(WORD$,Y-1)
 47   R$=RIGHT$(WORD$,X-Y)
 48   WORD$=L$+R$
 49 UNTIL WORD$=""
 50 NEWWORD$=TEMP$
 51 ENDPROC
 52 REM ============================
 53 REM
 54 REM procedure to display letters
 55 REM on screen
 56 REM ..........................
 57 REM
 58 DEF PROCdisplay
 59 LOCAL U,V,W,X,Y,Z
 60 COLOUR 129:COLOUR 2
 61 CLS
 62 Y = LEN (NEWWORD$)
 63 W=2:X=0
 64 REPEAT
 65   X=X+1
 66   LET L$ = MID$(NEWWORD$,X,1)
 67   FOR Z = 1 TO W
 68     PRINT TAB(Z,3) L$
 69     SOUND 1,-10,50+Z,1
 70     FORV=1TO150:NEXT
 71     PRINT TAB(Z,3) " "
 72   NEXT Z
 73   W=W+2
 74   PRINT TAB(Z,4) L$
 75   SOUND 1,-10,30,2
 76 UNTIL X=Y
 77 ENDPROC
 78 REM ============================
 79 REM
 80 REM procedure to check letter
 81 REM combinations are sensible
 82 REM ........................
 83 REM
 84 DEF PROCcheck
 85 LOCAL C$
 86 STOR = (ASC(STOR$)-65)*1
 87 RESTORE (94+STOR)
 88 FFLAG = TRUE
 89 REPEAT
 90   READ C$
 91   IF C$="9" THEN FFLAG = FALSE
 92 UNTIL PAIR$=C$ OR FFLAG = FALSE
 93 ENDPROC
 94 DATA AI,AU,9
 95 DATA BB,BL,BR,BS,BY,9
 96 DATA CH,CK,CL,CR,CT,9
 97 DATA DG,DM,DR,DS,DY,9
 98 DATA EA,EE,EI,EU,9
 99 DATA FF,FL,FR,FS,FT,FY,9
100 DATA GG,GH,GL,GN,GR,GS,GY,9
101 DATA HR,HT,HS,9
102 DATA IA,IE,IO,9
103 DATA 9
104 DATA KK,KN,KS,KY,9
105 DATA LC,LD,LF,LG,LK,LL,LM,LP,LR,LS,LT,LY,9
106 DATA MB,MC,MM,MP,MS,MY,9
107 DATA NC,ND,NG,NK,NN,NP,NS,NT,NY,9
108 DATA OA,OI,OO,OU,9
```

```
  109 DATA PH,PL,PN,PP,PR,PS,PT,PY,9
  110 DATA 9
  111 DATA RB,RC,RD,RF,RG,RH,RK,RL,RM,RN,RR,RS,RT,RV,9
  112 DATA SC,SH,SK,SL,SM,SN,SP,SQ,SS,ST,SW,SY,9
  113 DATA TH,TL,TR,TS,TT,TW,TY,9
  114 DATA UA,UE,UI,UO,9
  115 DATA VV,VS,VY,9
  116 DATA WH,WS,9
  117 DATA XY,XT,9
  118 DATA 9
  119 DATA ZZ,ZY,9
  120 REM ============================
  121 REM procedure to check the
  122 REM validity of character input
  123 REM against characters in the
  124 REM jumbled word
  125 REM ..........................
  126 REM SOUND 1,-15,20,5
  127 REM
  128 DEF  PROCincheck (INWORD$)
  129 LOCAL X
  130 GFLAG=TRUE
  131 X=INSTR(INWORD$,INCHAR$)
  132 IFX=0THEN GFLAG = FALSE ELSE INWORD$ = LEFT$(INWORD$,X
-1)+"9"+RIGHT$(INWORD$,LL-X)
  133 CONWORD$ = INWORD$
  134 ENDPROC
  135 REM ============================
  136 REM procedure to input from the
  137 REM keyboard and display char.
  138 REM ..........................
  139 REM
  140 DEF PROCgetin
  141 LOCAL X,FLAG1,FLAG2
  142 X=1 : CONWORD$ = NEWWORD$
  143 PRINT TAB(1,10) "Make your word"
  144 REPEAT
  145   VDU 31,X,12
  146   INWORD$ = CONWORD$
  147   *FX15,1
  148   INCHAR$ = GET$
  149   IF ASC(INCHAR$)=13 THEN :PRINT:ENDPROC
  150   PROCincheck (INWORD$)
  151   IF NOT GFLAG THEN 148 ELSE PRINT INCHAR$
  152   SOUND 1,-12,53,2
  153   KWORD$(KW)=KWORD$(KW)+INCHAR$
  154   PAIR$ = RIGHT$(PAIR$,1)+INCHAR$
  155   IFX=1THENX=X+1:STOR$=INCHAR$:GOTO145
  156   TEST$=INCHAR$
  157   PROCvowel (TEST$)
  158   IF HFLAG THEN FLAG1=TRUE ELSE FLAG1= FALSE
  159   TEST$=LEFT$(PAIR$,1)
  160   PROCvowel (TEST$)
  161   IF HFLAG THEN FLAG2 =TRUE ELSE FLAG2 = FALSE
  162   FFLAG=TRUE
  163   IF (FLAG1 AND FLAG2 ) OR ( NOT FLAG1 AND NOT FLAG2 )
 THEN PROCcheck
  164   IF NOT FFLAG THEN VDU 31,X,13:PRINT"^":SIGN = TRUE:S
OUND 1,-15,10,4
  165   STOR$ = INCHAR$ :X=X+1
  166 UNTIL LL = X-1
  167 ENDPROC
  168 REM ============================
  169 REM vowel check
  170 REM ............
  171 REM
  172 DEF PROCvowel (TEST$)
  173 RESTORE 180
  174 LOCAL X,C$
```

```
  175 HFLAG=FALSE
  176 FORX=1TO5
  177   READ C$
  178   IFC$=TEST$ THEN HFLAG=TRUE
  179 NEXT
  180 DATA A,E,I,O,U
  181 ENDPROC
  182 REM ============================
  183 REM procedure to introduce the
  184 REM game
  185 REM ........................
  186 REM
  187 DEF PROCstart
  188 PRINT"TO FINISH PRESS " CHR$(131) "ESCAPE" CHR$(135) "
KEY"
  189 PRINT TAB(3,3) CHR$(141);CHR$(131);"THE ANAGRAM TESTER
"
  190 PRINT TAB(3,4) CHR$(141);CHR$(131);"THE ANAGRAM TESTER
"
  191 PRINT TAB(3,6) CHR$(129) "This program is meant to"
  192 PRINT TAB(3,7) CHR$(129) "test the ability to"
  193 PRINT TAB(3,8) CHR$(129) "create letter pairs"
  194 PRINT TAB(3,9) CHR$(129) "from a valid anagram."
  195 PRINT : PRINT
  196 PRINT "RULES ............................."
  197 PRINT
  198 PRINT "You will see lots of letters in a mess"
  199 PRINT "Try to get them into a word."
  200 PRINT "The word can be as long as you like !"
  201 PRINT "The micro will make a noise and"
  202 PRINT "show a little arrow '^' if it"
  203 PRINT "thinks a letter is very silly."
  204 PRINT CHR$(141) ; CHR$(131) "PRESS ANY KEY TO START"
  205 PRINT CHR$(141) ;CHR$(131) "PRESS ANY KEY TO START"
  206 LET R$ = GET$ : ENDPROC
  207 REM ============================
  208 REM choose a word for the poor
  209 REM little mites to do.
  210 REM ........................
  211 REM
  212 DEF PROCgetword
  213 RESTORE 219
  214 LOCAL X,Y
  215 X = RND(10)
  216 FOR Y = 1 TO X
  217   READ WORD$
  218 NEXTY
  219 DATA MASSIVE,GANTRY,TRIBUTE,TRYING
  220 DATA SAUCER,VISION,MIXTURE
  221 DATA METAL,CATTLE,THROUGH
  222 ENDPROC
  223 REM ============================
  224 REM end of game routines
  225 REM ......................
  226 REM
  227 MODE 7:ON ERROR GOTO 224
  228 PRINT CHR$(141);CHR$(131);"CALL YOUR TEACHER"
  229 PRINT CHR$(141);CHR$(131);"CALL YOUR TEACHER"
  230 PRINT
  231 PRINT TAB(10) CHR$(130) "Options"
  232 PRINT TAB(10) CHR$(130) "-------"
  233 PRINT
  234 PRINT TAB(5) "Print out words and results    (1)"
  235 PRINT TAB(5) "Jumble another word            (2)"
  236 PRINT TAB(5) "Restart the program again      (3)"
  237 PRINT TAB(5) "End the program                (4)"
  238 PRINT
  239 PRINT "Enter 1,2,3 or 4 only >> ";
  240 R$=GET$
```

```
  241 IF R$ < "1" OR R$ > "4" THEN 240
  242 PRINT R$
  243 IFR$="1" THEN PROCresult :GOTO 224
  244 IFR$="3" GOTO 16
  245 IFR$="2" GOTO 19
  246 ENVELOPE 1,3,2,2,2,5,5,5,0,0,0,-1,120,120
  247 SOUND 1,1,10,30
  248 PRINT TAB(5,15) CHR$(136);CHR$(131);"BYE BYE ... Ta":E
ND
  249 REM ============================
  250 REM display words of kiddy
  251 REM ......................
  252 REM
  253 DEF PROCresult
  254 CLS :LOCAL X,Y
  255 PRINT CHR$(157) CHR$(141) CHR$(151) CHR$(132) TAB(8) "
RESULTS"
  256 PRINT CHR$(157) CHR$(141) CHR$(151) CHR$(132) TAB(8) "
RESULTS"
  257 PRINT:PRINT CHR$(131) "Jumble was " CHR$(130) NEWWORD$
 "  from  " WORD$
  258 PRINT:PRINT
  259 FOR X=1TO60 STEP 4
  260   PRINT TAB(1)  KWORD$(X); TAB(10) KWORD$(X+1); TAB(20
) KWORD$(X+2); TAB(30) KWORD$(X+3)
  261   IF KWORD$(X+3)="" THEN X= 60
  262 NEXT : PRINT
  263 PRINT"LET YOUR TEACHER LOOK "
  264 CN$=GET$:IF CN$<>"["GOTO264
  265 ENDPROC
  266 REM ========================
  267 REM   child chooses result
  268 REM ......................
  269 REM
  270 DEF PROCchoose
  271 LOCAL X
  272 PRINT TAB(1,15) "An English word ?"
  273 WE$ = GET$
  274 IF WE$ = "N" THEN KWORD$(KW) = KWORD$(KW) + "*":PRINT"
NO":SOUND 1,-15,10,4:ELSE PRINT"YES" :SOUND 1,-12,70,5
  275 FOR X=1TO500 :NEXT
  276 ENDPROC
```