Knights T.V. & COMPUTERS

A MEMBER OF RADIO & TELEVISION RETAILERS' ASSOCIATION
Bankers: Clydesdale Bank Ltd., Rosemount Aberdeen.
Consumer Credit Licence 012828 V.A.T. Reg. No. 265 3119 66

108 Rosemount Place,
Aberdeen AB2 4YW
Telephone: 0224 630526
Telex: 739169 "KNIGHTS TV"

KNIGHTS WEE PASCAL

A great deal of coverage has been given in the microcomputing press about PASCAL - the " wonder language." . This KNIGHTS WEE PASCAL interpreter tape provides an excellent introduction to the concepts of PASCAL programming. Experienced programmers will notice many omissions from PASCAL as defined by Jensen and Wirth but many complex programs can be run with a little ingenuity. The subset of commands in KNIGHTS WEE PASCAL has deliberately been kept sufficiently simple for beginners to use and understand quickly.

LOADING KNIGHTS WEE PASCAL

The language tape loads from monitor - simply switch on your MZ-80K and enter LOAD. Press the CR key and the screen will then prompt you to press the play button on the tape unit. Once loaded the tape will stop, a tone will sound and the screen will display KNIGHTS WEE PASCAL. READY and the prompt > will indicate that you are in Command Level.

COMMANDS

A Append a program from tape to end of current text,or to load a program if there is no text in memory.

B Inserts program lines from the end of text until the CR key is pressed on an empty line or an error is detected. A check is made for matching %'s or "'s when a line is entered.

D n Deletes n lines of text from and including the current line.

E $nnnn Limits upper address of memory to $nnnn.

F string Starting from current position,find the line starting with specified string. Leading spaces are ignored but e.g. FL will not find L::L+3; FL:= L+3; would have been needed."F PROC" however will find the line PROC PLOT(X,Y);. If the find command is used with a single % or " an error message will be displayed.

G/ Runs the program

Hn program is copied to the printer for n lines from CP.

I string Inserts a string in a line directly before current position (CP)

I CRkey Enter insert mode and continue inserting lines until the CR key is pressed on an empty line.

K/ Clears current program from memory.

L n Moves n lines upwards from CP

N n Moves n lines downwards from CP

P n Displays n lines on the screen and makes last displayed CP.

Q/ Quit Knights Wee Pascal and return to monitor. GOTO$1200 restores Wee Pascal. GOTO$122C for a warm start.

R string Replace current line with a string.

S Save program on tape.

T n Move to line n

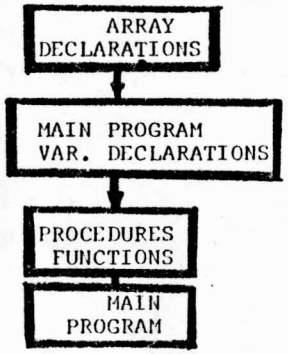
T Move to the top of text (=TO)

Notes L,N by themselves displays current line.
P999 will display the whole program. i.e. no error for printing non existent lines.
Stop listings to the screen by pressing the space bar. Pressing the space bar again resumes listing.

USING KNIGHTS WEE PASCAL

Particular attention must be paid to the layout of all Pascal programs. Pascal does not use line numbers; instead the program has to be layed out according to a general overall structure.Remember a semi colon is used to denote the end of a statement line.

SCHEMATIC VIEW OF KNIGHTS WEE PASCAL



Generally,the main program consists of calls to procedures and functions previously declared. Programs should be elegant and informative as well as being easy to read.

LUXURIES OF PASCAL

Variable names of large length make it possible and desirable to call a variable maxsize rather than use X1 as would be usual in Basic. There is no GOTO statement in KNIGHTS WEE PASCAL - it is versatile enough as it is. There are three types of loop.
These are :

```
FOR X :=1 TO 50 DO BEGIN ..... END;
  WHILE X = 50 DO BEGIN ..... END;
  REPEAT ..... UNTIL X = 50;   (..... is a block of statement
```

VARIABLE NAMES There is no limit to the length of VARIABLE NAMES and all the characters are distinguished.
e.g. AAAAAAAB is treated differently from AAAAAAAC.
N.B. PASCAL programs are written in UPPER CASE LETTERS.

KEY WORDS

VAR declares variables. KNIGHTS WEE PASCAL is an integer PASCAL. Variables are integer and must be declared before use.
e.g. VAR ALPHA;
VAR ALPHA,BETA;

BEGIN represents the start of a block of statements. Blocks are used in procedures and functions and in the main program. Blocks may also be used after conditional statements.

```
e.g. IF GAMMA > DELTA THEN
     BEGIN
       ⋮
     BLOCK
       :
       :
     END;
```

It is generally conventional to place nested blocks with indentation

```
e.g. BEGIN
     IF GAMMA > DELTA THEN
        BEGIN
        IF  SHARP=0 THEN
           BEGIN
             ⋮
           END;
        END;
     END.
```

END; represents end of a block of statements.

END. used only at the very end of a program.

PROC used to declare a procedure. A procedure is a set of statements identified by a name which may take values from the main program. Variables may be declared in the procedure entirely local to the procedure (local variables). The procedure may alter the variables declared at the start of the program (global variables). Programmers must take great care to ensure that there is no mix up between local and global variables.

FUNC similar to procedure except that a value is returned to the main program.

ARRAY Similar to Basic arrays. e.g. ARRAY SHARP[12]
Note single dimension arrays and the use of the square brackets.

READ(X); reads integer X from keyboard with a ? prompt.

IF...THEN ... ELSE Logical IF..THEN..ELSE
e.g. IF SHARP > 4 THEN SUM ELSE DSUM; Note SUM and DSUM are procedure names

PUT(X); places a character with ASCII code X at current position.

PUT("TEXT"); places text at current screen position.

WRITE( ); format for writing information e.g. WRITE("NUMBER IS",X,/);
Note. / is the line feed character.

INP(n); inputs from port n (where n is a number between 0 and 255)

OUT(n); outputs from port n

POKE(X):=Z; inserts byte Z at memory location X.

Z:=PEEK(X) Z becomes content of location X (as in Basic)

Z:=GET; Similar to GET in Basic.

LOGICAL OPERATORS

:=	assignment statement	e.g. X:=3
=	comparison test (as in Basic)	e.g. IF(X=3) THEN
OR	inclusive or	e.g. IF(X=3) OR (Y=2) THEN
XOR	exclusive or	e.g. IF(X=3) XOR (Y=2) THEN
# <>	not equal to	e.g. IF(X#Y) THEN
	not equal to	e.g. IF(X<>Y) THEN
>	greater than	
<	less than	
>=	greater than or equal to	
<=	les than or equal to	
NOT	IF NOT (X<Y) THEN	
AND	IF (Y=3) AND (Y>X) THEN	

ARITHMETICAL OPERATORS

+	ADD	e.g. Y:=X+3;	
*	MULTIPLY	e.g. Y:=X*3;	
-	Subtract	e.g. Y:=X-3;	
DIV	INTEGER DIVIDE	e.g. Y:=X DIV 3	Note. 12 DIV 8 = 1
MOD	REMAINDER	e.g. Y:=X MOD 3;	Note. 12 MOD 8 = 4
RND	RANDOM NUMBER ENERATOR	Y:= RND(13);	
ABS	ABSOLUTE VALUE	Y:= ABS(X);	
INC	INCREMENTS VARIABLES BY 1	e.g. IF X=5,Y:= INC(X) = 6	
DEC	DECREMENTS VARIABLE BY 1	e.g. IF X=5,Y:= DEC(X) = 4	

RUNNING THE SAMPLE PROGRAMS

Load the language as detailed on page one. Enter A and press the CR key, FILE NAME ? will appear on screen, enter the name of the next program on the tape or simply press CR key again. Once loaded G/ runs the program SHIFT and BREAK operate like BASIC. K/ clears memory before loading next program by entering A and pressing the CR key twice.

POINTS TO NOTE WHEN PROGRAMMING

COMMENTS These should be placed on separate lines.

FORMATTING Output from a write statement may be formatted using :X after the variable is written out. X being a positive integer.
e.g. WRITE ('MAXIMUM VOLTAGE IS',VMAX:3);
writes out VMAX in a field of 3. A field of 1 writes out the number without any padding spaces.
Remember there is no immediate mode as in Sharp Basic. e.g. WRITE(3*4); entered at editor level has the effect of creating a syntax error - not printing 12 as you would expect in the equivalent Basic statements.

NUMBERS Numbers are stored as signed 15 bit integers. The acceptable range of numbers is -32768 to 32767 ($0000 to $FFFF)
On entering a program (using I or B) the interpreter checks that values are within this range - if the range is exceeded the message ERROR 00 RUN appears

DEBUGGING

Most errors can be spotted by entering T followed by the line number indicated in the error message. The most frequent errors are: Undeclared variables/arrays . Function does not return a value to main program . Unmatched begin/ends . Missing ; remember the ;;;;;;;;;;; Numbers out of range . A comment on a statement line . Loops not properly closed . Non existent procedure/function calls.

If the programmer uses some sort of indentation to show the nesting of loops, etc - programs become much easier to debug. Mnemonic variable names are also a distinct advantage.

ERROR MESSAGES

THE MOST IMPORTANT ERROR MESSAGES ARE:

00 value out of range
01 stack error
02 can not perform operation
03 syntax error
06 illegal symbol
09 illegal output field
11 illegal operation on array
13 BEGIN missing
14 undeclared variables
16 no function value
19 statement following END.
20 program does not finish with END.
21 invalid array size
25 - 27 printer errors

Knights T.U. & COMPUTERS

The following five programs will assist newcomers to Pascal in learning the procedures and facilities available on Knights Wee Pascal.

This program is one of the simplest possible in Wee Pascal. There are no variable declarations because there are no variables in the program.

```
BEGIN
WRITE ("Knight's for Sharp!");
END.
```

Program one

This program is similar to example 1, but uses variables and a loop to write out the message ten times.

```
VAR X;
BEGIN
FOR X:=1 TO 10 DO
   BEGIN
   WRITE ("Knight's TU!",/);
   END;
END.
```

Program two

If only one statement is to be executed inside a loop, then the BEGIN & END may be omitted.

```
VAR X;
BEGIN
FOR X:=1 TO 10 DO WRITE ("!!",/);
END.
```

Program three

A procedure is declared before the main body of the program, and may take optional parameters from the main program for use within the program. Local variables may be declared within procedures, these exist only for the time that the procedure is being executed, and are completely distinct from any global variables that may have the same name. Procedures are called from the main program or from other procedures using a procedure name declared at the start of the procedure.

```
VAR X;
PROC WRITEOUT (X);
VAR COUNT;
BEGIN
FOR COUNT:=1 TO X DO
   WRITE ("Knight's TU!",/);
END;
BEGIN
   REPEAT
   WRITE ("How many times ");
   READ(X);
   UNTIL (X>0);
WRITEOUT (X);
END.
```

Program four

Program five

This program is an example of a recursive function ie one that calls itself. Note that if the number entered is too large, strange answers will be produced due to internal overflow. Also note the format for output of the result, using a number to specify the output field. A field of 1 indicates that no padding should be supplied.
Also note the use of PUT(#16), which has the effect of ?"[C]" in BASIC.
Printing (/) causes a line feed

```
VAR X;
   FUNC FACT (N);
   BEGIN
   IF N=0 THEN FACT:=1
   ELSE
   FACT:=N*FACT (N-1);
   END;
BEGIN
PUT (#16);
   REPEAT
   WRITE ("What factorial do you require (0-7)")
   READ(X);
   UNTIL (X<8) AND (X>=0);
WRITE (X:1," factorial is ",FACT (X):1);
WRITE (/);
END.
```

NOTES ON KNIGHTS WEE PASCAL by I.S.U.G member Simon Finn

Known Bugs

When the command E (set memory limit) is used,it resets the top of memory pointer. Unfortunately it does not rest any of the other system variables and should therefore be followed by SHIFT-BREAK to force a warm start,when all will be well. When the command K/ is used to delete a program from memory the system stack overwrites the top of memory, ignoring any top of memory limit that may have been set by using E command. This may corrupt any machine code subroutine.

Extra commands and statements

The command M gives the amount of free memory. On a 48K Sharp this is usually more than 32K and so appears negative due to the limitations of integer arithmetic. STOP is a statement causing an immediate halt in program execution Note: unlike BASIC there is no CONT command. ADRES(variable) this function gives the actual address where the variable is stored. This is most useful for passing values to machine code subroutines. CALL passes control to a machine code routine. There are three forms:

1. CALL(address) is like USR in BASIC, but may be used either like a function or like a procedure.
2. CALL (address): = expression.
3. CALL (address): = expression 1, expression 2.

In 2 and 3, the first expression is passed in the HL register and the second expression if used is passed in the DE register pair.

PUT and WRITE are like the PUT and WRITE but use the printer.

Unexpected features

The subscript of an array is only checked to be 0 , so care must be exercised in case the use of one array overwrites another. If a program has only one statement, it does not need BEGIN.....END.

e.g. VAR X;
FOR X:=1 to 10 WRITE(X). is in fact a valid program.

/ may be used instead of DIV.

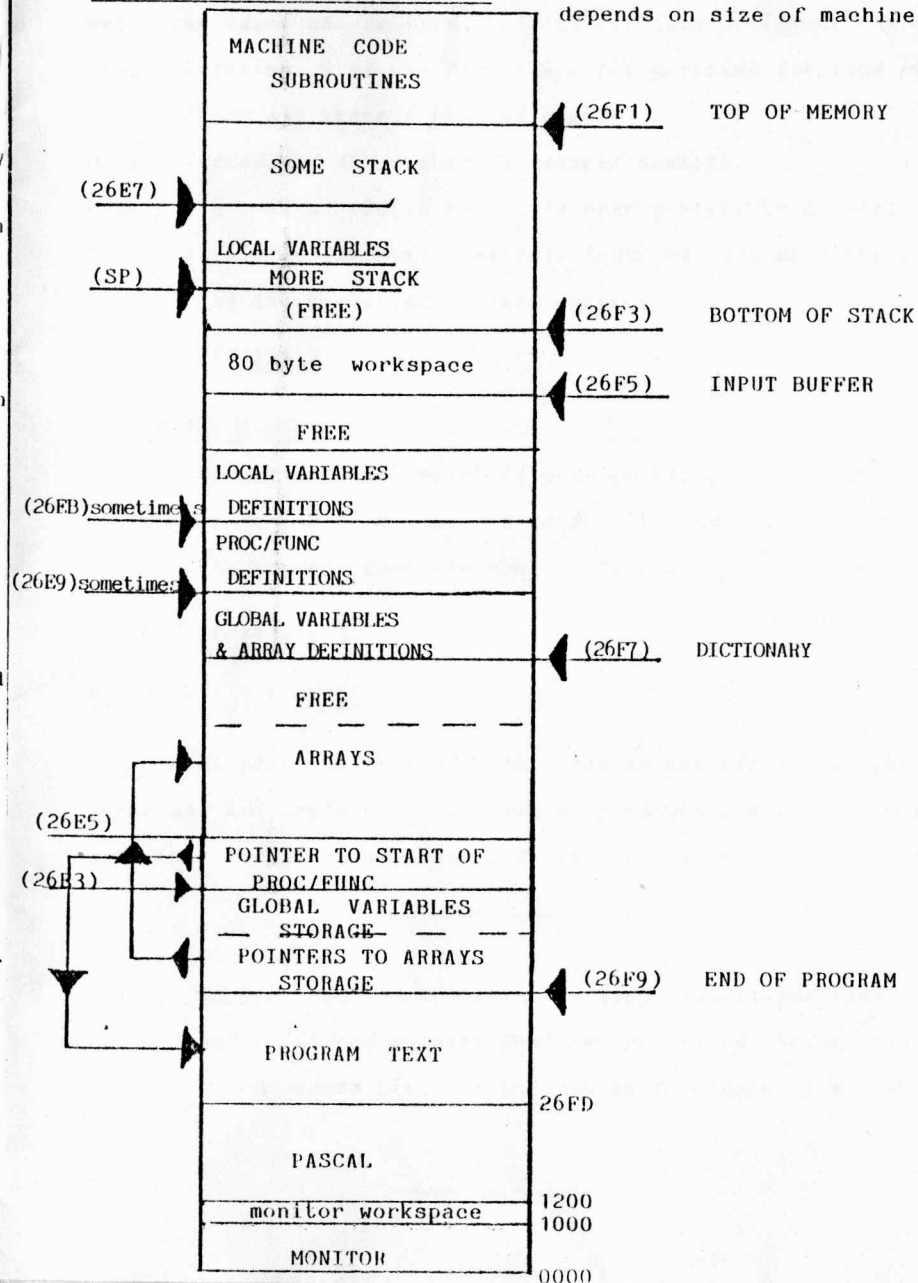
ERROR MESSAGES

0 value out of range
1 stack error or insufficient space to load program.Or array name used twice.
2 cannot perform operation - especially divide by 0
3 syntax error 4 BEGIN without matching END
5 REPEAT without matching UNTIL.
6 IF without THEN or unknown statement type.
7 DO missing 8 FOR without either DO or DOWN
9 missing. 10 := missing. 11 missing. 12 missing.
13 Main program does not end with . 14 Variable not known.
16 Wrong number of parameters for function or procedure.
17 Too many parameters on CALL statements.

ERROR MESSAGES continued....

18 Expression in quotes does not consist of one character.
19 Text following . 20 Program does not finish with a .
21 Array size is not positive decimal integer.
22 Invalid name 23 Procedure used in function call.
24 Tape READ/WRITE error. 25-27 are all printer errors.

MAP OF THE SPACE UTILISATION.



Knights T.V. & COMPUTERS

A MEMBER OF RADIO & TELEVISION RETAILERS' ASSOCIATION
Bankers: Clydesdale Bank Ltd., Rosemount Aberdeen.
Consumer Credit Licence 012828 V.A.T. Reg. No. 265 3119 66

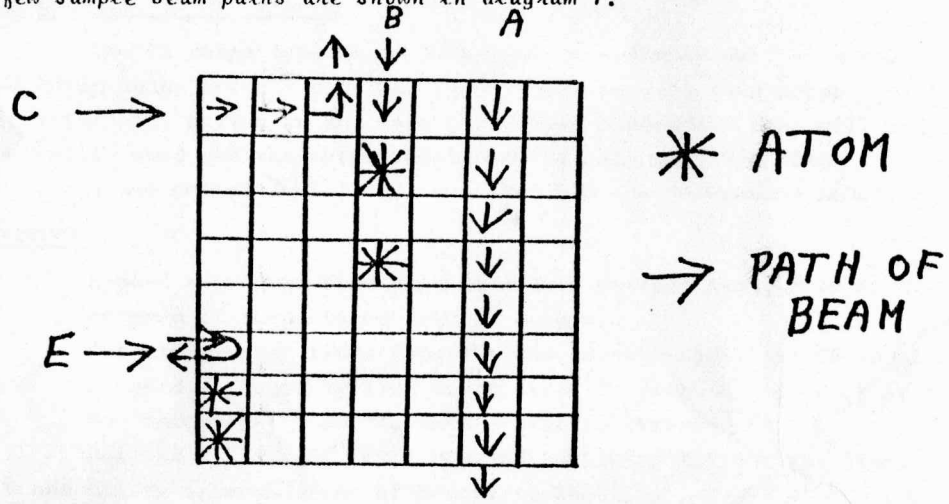
108 Rosemount Place,
Aberdeen AB2 4YW
Telephone: 0224 630526
Telex: 739169 "KNIGHTS TV"

KNIGHTS WEE PASCAL "BLACK BOX"

Black Box is an intriguing game for up to four players involving skill, deductive reasoning and a certain amount of luck. Four atoms are hidden in a 7 x 7 black box - the players have to find the exact location of each atom. The game consists of rounds, each with two sections per player.

SECTION 1 - firing a beam

To try to locate the atoms each player fires a beam into the box from one edge. The beam rebounds off the atoms until it finally emerges or is absorbed. A few sample beam paths are shown in diagram 1.



A: beam passes through the box and emerges on the other side undeviated by any atoms.

B: beam enters the box and is absorbed by an atom as it strikes it head on.

C: beam passes near an atom and is deflected through 90 .

D: beam cannot deviate to the right or left due to the presence of two atoms - the beam therefore deviates through 180 .

E: atom is located on the edge square of the screen so that any beam entering near is deflected off the screen.

After the player has keyed in the place for the beam to enter - the exit position (if it is not absorbed) is displayed. Different sounds are made as the beam travels through the box and depending on the position of the [illegible] possible for the beam to ricochet before exiting.

SECTION 2

At any stage in the game a player may guess up to four of the positions where the atoms are located. Getting all four positions correct wins the game. Entering 0 at any time stops the guessing for that player's turn.

There are three levels of play:

Mode 1 : displays the number of correct guesses.

Mode 2 : a is displayed if one or more guesses is correct.

Mode 3 : This is the most difficult level of play as there is no indication of the number of correct guesses.

SECTION 3

ENDING THE GAME

No matter which level of play is being used - all games are ended by entering E in response to the SIDE prompt.

The hidden atoms are now displayed.

SECTION 4

REPLAY ALL THE MOVES

As play has proceeded the program has stored all the moves and these are now replayed. This time each player sees the beam enter from positions selected and their path through the box is revealed.

Note. We have not incorporated any copy protections into this program and we indeed suggest that inexperienced PASCAL programs use the P command to list the program on the screen and study its structure.
