The programs in this book use the following statements and functions in Microsoft Basic:

Statements		
DATA	Holds numeric or string data for a READ statement	20 DATA 4,6,"AHL"
DEF FNA(X)	Defines any function of X	20 DEF FNA(X)=3*X-2 20 DEF FNA(X)=SIN(X/57.3)
DIM	Declares maximum size of string or numeric array. Array subscripting begins at 0 although many programs do not use the zero subscript.	20 DIM A(50) 20 DIM A$(25),B1$(50)
END	Last statement in program	9999 END
FOR..TO..(STEP)	Executes a loop. The test for ending the loop is made after the loop has been executed. Upon exiting, the counter value equals the upper limit plus the step. For example, 10 FOR J=1 TO 3 20 PRINT "HI" 30 NEXT J will print "HI" three times, and J will equal 4 when the loop is finished.	20 FOR I=1 TO 30 20 FOR J=2 TO N STEP 3
GOSUB n	Branch to subroutine n	20 GOSUB 200
GOTO n	Branch to statement n	20 GOTO 50
IF..THEN n	Branch to statement n if condition is true	20 IF A>1 THEN 50
IF..THEN stmts	Executes statements if condition is true. Drops to next numbered line if false.	20 IF Z<5 THEN A=1:PRINT B
IF..THEN n ELSE m	Branch to n if true or to m if not true	20 IF X=Y THEN 50 ELSE 90

IF..THEN stmts
ELSE stmts
	Does statements after THEN if true, stmts after ELSE if false	20 IF Z>R THEN X=1 ELSE X=2

INPUT	Requests data from keyboard. The prompt string is optional	20 INPUT N 20 INPUT "YES OR NO";Z$
LET	Assigns value of expression to variable. The word LET is optional	20 LET A=1 20 Z$="DRY"
NEXT	Marks end of FOR loop	20 NEXT J
ON m GOSUB...	Branch to mth subroutine	20 ON X GOSUB 100,200
ON m GOTO...	Branch to mth line no. In these statements, m must be an integer starting at 1 and increasing by 1	20 ON Y GOTO 50,80,120
PRINT	Displays strings, constants and variables. Calculations can be done within a PRINT statement	20 PRINT "A=";A 20 PRINT Z$,10*A+B
READ	Moves values of DATA into variables	20 READ N,X1,A$
REM	Remark. Does not execute	20 REMARKABLE PROGRAM
RESTORE	Resets DATA pointer to first item in list	20 RESTORE
RETURN	Go to statement following last GOSUB	20 RETURN
STOP	Terminate program	20 STOP

Functions

ABS (X)	Absolute value.
ASC(X$)	Returns the ASCII value of the first character in the string argument. ASC("A") is 65,ASC("B") is 66, etc.
ATN(X)	Arctangent
CHR$(X)	Converts ASCII value to a character string. CHR$(65) is "A", etc. CHR$(7) is a bell ring.
COS(X)	Cosine
EXP(X)	Value of e raised to the X power
INT(X)	Integer function
LEFT$(X$,Y)	The leftmost Y characters in X$
LEN(X$)	Number of characters in X$
LOG(X)	Log of x to the base e
MID$(X$,Y,Z)	Takes Z characters from X$ starting at position Y
RND(1)	Returns a random number between 0 and 1.
RIGHT$(X$,Y)	The rightmost Y characters in X$
SGN(X)	Sign function. Returns -1 if X is negative, 0 if x is zero, 1 if x is positive.
SIN(X)	Sine
SQR(X)	Square root
STR$(X)	Converts X to a string of decimal digits, for example if X was 8.45 it would be converted to the string "8.45".
TAB(X)	Spaces over to position X on the terminal.
TAN(X)	Tangent
VAL(X$)	Returns the numerical value of the string of digits in X$. Opposite of STR$(X).

In Microsoft (Altair) BASIC, any expression may be evaluated as either true or false. A true condition will return a value of -1, and a false condition 0. Thus, if we say LET Q= -(X=Y), Q=1 if X=Y and Q=0 if X=Y. This logical evaluation of expressions is only used in the Hexapawn game in the user-defined function and with a little ingenuity could be replaced with a look-up table. A few other games use the logical AND and OR operators, which work in a straightforward manner.

The programs in this book were printed on a printer that uses a caret (∧) to indicate exponentiation. This is equivalent to an up arrow. Incidentally, exponentiation and taking roots are among the least accurate functions on small computers. For example, try this program.

```
10 INPUT N
20 I=SQR(SQR(N))
30 J=(I↑2)↑2
40 PRINT N,J
50 GOTO 10
```

Chances are good that N and J will not be the same 25% or more of the time. You can improve the accuracy by substituting J*J for J squared or J*J*J for J cubed.

Microsoft Basic permits more than one statement on a line when the statements are separated by a colon (:). As noted above, in an IF..THEN statement, if the condition is false, control drops to the next numbered line, not to the next statement on the same line.

This means that for TI and other computers that do not permit multiple statements on one line, you will have to insert additional lines. This may be difficult when line numbers are close together. One easy solution is to add a zero to all line numbers, but you must remember to do this in IF..THEN, GOTO and GOSUB statements as well as changing all the numbered lines.

The random function can be especially irksome as it is one that differs widely on different versions of Basic. In Microsoft Basic, RND(1) returns a value between 0 and 0.999999. This is the convention used in all programs in this book. On some computers, you may have to use RND(0), and on others just RND.