VARIATIONS ON A TURTLE

Our examination of the LOGO language has already shown us how procedures can be defined to carry out sequences of commands. Such procedures are more flexible if the user is able to input different values that will alter the effect achieved when the procedure is called.

In LOGO, a word — here we'll take SIZE as an example — may be used in three different ways. To distinguish between these, LOGO uses three different notations: SIZE, :SIZE (pronounced 'dots size'), and "SIZE ('quotes size'). As we have already seen, if LOGO meets the word SIZE, with no preceding punctuation, it takes it to be a procedure name and will carry out the sequence of commands in the SIZE definition. :SIZE is used to indicate the *value* held in the variable name — if LOGO encounters :SIZE it will retrieve the value associated with the name. "SIZE is used to refer to variable names and procedures, but indicates that we are referring to the *name* itself, and not to any value that might be associated with it. Thus, "SIZE might be used to refer to a variable, while :SIZE would refer to the value that has been allocated to that variable. (Note that quotes are required before, but not after, the word. The end of the name is indicated by a space.)

MIT LOGO is not totally consistent in its use of this notation. After the commands EDIT, ERASE and PO, a procedure name should be written without quotes. Thus, the correct syntax is EDIT SQUARE, even though SQUARE is not a call to the procedure SQUARE, but is merely the name of that procedure and should logically be preceded by quotes. LCSI LOGO is more logical and demands that quotes are used with these commands.

To use any of the procedures we have defined so far we simply type the procedure name, in exactly the same way as we would use LOGO commands such as DRAW or HIDETURTLE. However, other commands — FORWARD, for example — need extra information before they can be used. The word FORWARD on its own is meaningless — a value must be assigned to it before LOGO can carry out the command. If variable names are included in our procedures, we can input any required value and thus vary the effect obtained when the procedure is called.

Let's consider the procedure we defined in a previous instalment to draw a square:

```
TO SQUARE
   REPEAT 4 [FD 50 RT 90]
END
```

As it stands, this procedure will draw a square with sides of 50 units in length. However, it would be far more useful if the square could be drawn to any chosen size — to do this, we must input the desired value. To change SQUARE to accept an input, use the editor to replace the fixed value of 50 with the variable "SIDE and add :SIDE to the title line. Our procedure will now look like this:

```
TO SQUARE :SIDE
   REPEAT 4 [FD :SIDE RT 90]
END
```

When the procedure is called, it is now necessary to give the variable "SIDE a value. Try SQUARE 40, SQUARE 10, and so on to see how the size of the square will vary.

Let's see exactly what happens when you type SQUARE 30. LOGO first looks up the definition of SQUARE. The title line tells it that one input is required, and that this is to be named "SIDE. The value on the input line (in this case 30) is allocated to the variable "SIDE and the commands in the procedure definition are then obeyed. The best way of visualising this is to imagine that each variable name refers to a box containing its value. When LOGO reaches the line FORWARD :SIDE it goes to the box labelled "SIDE, retrieves the value found there and uses this as the input to FORWARD. The box labelled "SIDE is used only in the procedure that refers to it. If another procedure also uses "SIDE as the name for an input, it will use a different box. "SIDE is therefore referred to as a *local* variable.

We can also use inputs with subprocedures. The HOUSE procedure we give here is our solution to the problem set in the last instalment (your answer may be different) can be modified so that different values may be input:

```
TO HOUSE :BIG
   SQUARE :BIG
   FD :BIG RT 30
   TRI :BIG
   LT 30 BK :BIG
END

TO SQUARE :SIZE
   REPEAT 4 [FD :SIZE RT 90]
END

TO TRI :SIDE
   REPEAT 3 [FD :SIDE RT 120]
END
```

Here, we have used three different variable names — "BIG, "SIZE and "SIDE. We could have used the same name for all three, as variables are local to the procedures in which they are used, but this could have been confusing.