```
5   A%=7%
10  PRINTLARGER@(5%,3%)
15  PRINTLARGER@(-3%,A%+1%)
20  SUB PRINTLARGER@(X%,Y%)
30     IF X%>Y% THEN PRINT X% ELSE PRINT Y%
40  SUBEND
RUN
5
8
```

In the above example, the PRINTLARGER@ subroutine is called first at line 10. The subroutine starting at line 20 is executed with X% equal to 5% and Y% equal to 3%; a 5 is printed. The subroutine is again called at line 15, with X% now equal to -3% and Y% equal to A%+1%, that is, 8%. An 8 is printed.

The variables X% and Y% are called dummy variables. Their values are set by the expressions in the parentheses following the use of the subroutine name in the program. These values hold, however, only in the subroutine. If X% and Y% are used elsewhere in the program, their values remain unchanged by what happened to X% and Y% in the subroutine. In effect, the dummy variables X% and Y% in the subroutine would be different from the X% and Y% used outside of the subroutine.

Special note: In most situations the system will automatically convert between real and integer numbers. In user defined functions and named subroutines, however, the argument must be the same in the definition and when the function or subroutine is used. For instance, the following program gives an argument error when compiled. This is because the definition at line 20 shows an integer argument, but a real argument is used at line 10.

```
10 PRINT FNT(A)
20 DEF FNT(B%)=2
```

The system will go around a function or subroutine definition automatically, just as it goes around DATA statements. So you may place functions and subroutines anywhere in the program. Note that the definition of a function or a subroutine does not need to precede its use.