4. Programming in COBOL

The method of writing and testing COBOL programs will vary from system to system - and you will need to look at the reference manual for your own COBOL compiler to know exactly how to do this.

In general - you will need to be able to -
1.   key in and edit the lines of instructions that you want to write;
2.   compile them (translate them to the computer's own language);
3.   set up a file containing the machine code version of the program which the computer can load from disk straight into any vacant area in its memory when required;
and then
4.   try out your instructions (i.e. run the program) to see that the computer does what you want.

Sometimes one program will allow you to do all these things - sometimes you will need a separate program to do each of these tasks - particularly an editor (or word processor), a compiler and a linker.

If you make any mistakes, they will generally show up at two points:
1.   when translating the program from COBOL into machine code (compiling) - because the compiler does not recognize something you have written (e.g. a misspelling, or incorrect punctuation);
and
2.   when running the program you may find that it does not do what you hoped it would do or simply stops.

To put right *compilation errors* - check first for simple mistakes such as leaving out full-stops or spelling words incorrectly. After that, check that what you have written is all correct COBOL grammar (compare with the examples given in the book).

When trying to put right errors which occur when trying to run your program - you have to remember that the computer will try to do exactly what you have told it to do.

Sometimes it will come across something that it finds impossible - such as trying to get information from a file that does not exist, or trying to divide by zero - in which case you will have what is called a *run-time error*. In these cases you have to alter the program as necessary or make sure that the files you told it were available for it to use are in fact on the disk.

At other times the program will work but will not what you wanted it to do because you have given it incorrect instructions (e.g. you have told it to ADD when you meant SUBTRACT). This is called a *logic error* and will mean that you will have to alter the program.