Introduction to Scientific and Engineering Computation

(BIL 102E)

LECTURE 3

Main Parts of a Program

1. Beginning of the program
2. Declarations (of variables, constants etc)
3. Execution
   1. Initialisation
   2. Reading data from input devices
   3. Calculations
   4. Writing results to output devices
4. End of the program

SYNTAX

When writing programs a certain set of grammar rules has to be obeyed. These rules are called the *syntax* of the programming language.

For example, as a syntax rule in C language a variable name cannot start with a number.

HELLO WORLD PROGRAM

```
#include <stdio.h>
main()
{
  printf("Hello World!\n");
}
```

This simple program just prints the words `Hello World!’ on the screen.

A C program can be seen as a list of statements.

Don’t ever forget that

- a program starts from one point (the curly parenthesis after the main() statement), after each statement the following statement is executed unless otherwise is forced by the program.
- The program ends at one point (the curly parenthesis that closes main() part).
- Only one statement is executed at a time.

The preprocessor statement **#include** causes the library called *stdio* to be used.

Here *stdio.h* is called a *header* file. Header files are used to be able to use predefined functions. For example **printf** function is defined in *stdio.h*.

The execution of the program always starts with the **main** function.

Every C program must have a main() function. More on functions later on.

The **printf()** (standard library) function is used to print out whatever is inside the quotes.

Note that \n inside a print statement means the new line character.

In UNIX environment, in order to go to the beginning of a line, the carriage return character, \r, can be used.

The **return** statement in the main function causes the program to stop the execution. Here 0 indicates that the program has terminated normally.

Note that C is a **case sensitive** language. Generally all predefined statements and functions are in lower-case letters.

There is a semi-colon (;) after all statements.

Compiling and Linking

In order to convert a C program to machine code we need to compile it. The program that does this job is called a C **compiler**.

The compiler takes the C program (source code) and all the necessary header files required by the program and produces and object file (.obj).

This object file, if necessary together with other object files (libraries), is converted to an executable file by another program called **linker**. The resulting executable file can be *run* on the machine.

In some software development environments such as MS Visual C++ the compiler and linker can be together.

ERRORS

There are two kinds of errors:

1. Syntax (Grammar) Errors
2. Semantic (Logical) Errors

Syntax Errors

If your program does not obey the grammar (syntax) rules of the programming language you end up with syntax errors. For example,

- **## include** <stdio.h>
- **# include stdio.h**
- **Main()**
- **main**
- **Printf(“Hello World”);**
- **printf**(“Hello World”)

Compiler usually detects these type of errors. (**compiler errors**)

Semantic Errors

If there is a problem with the logic of your program you end up with these kind of errors. These can be much more difficult to detect and remove.

Example:

In the above program if we had written

z=x-y

Instead of z=x+y statement the program would be grammatically correct but produce wrong results.

These type of errors are sometimes called **run time errors**.

Software Development

Four steps in writing a program:

- Specify the problem clearly.
- Analyse the problem and break it down into smaller parts.
- Code the program.
- Test the program exhaustively.