Lesson no. 3: C-Program Structure

Before we study basic building blocks of the C programming language, let us look a bare minimum C program structure so that we can take it as a reference in upcoming chapters.

A C program basically consists of the following parts:

- Preprocessor Commands
- Functions
- Variables
- Statements & Expressions
- Comments

Let us look at a simple code that would print the words "Hello World":

```
#include <stdio.h>

int main()

{

/* my first program in C */

printf("Hello, World! \n");

return 0;

}
```

Let us look various parts of the above program:

The first line of the program #include **<stdio.h>** is a preprocessor command, which tells a C compiler to include stdio.h file before going to actual compilation.

The next line **int main()** is the main function where program execution begins.

The next line **/*...*/** will be ignored by the compiler and it has been put to add additional comments in the program. So such lines are called comments in the program.

The next line **printf(...)** is another function available in C which causes the message "Hello, World!" to be displayed on the screen.

The next line return 0; terminates **main()** function and returns the value 0.

Compile & Execute C Program:

Lets look at how to save the source code in a file, and how to compile and run it. Following are the simple steps:

Open a text editor and add the above-mentioned code.

Save the file as hello.c

Open a command prompt and go to the directory where you saved the file.

Type gcc hello.c and press enter to compile your code.

If there are no errors in your code the command prompt will take you to the next line and would generate a.out executable file.

Now, type a.out to execute your program.

You will be able to see "Hello World" printed on the screen

Hello, World!

Make sure that gcc compiler is in your path and that you are running it in the directory containing source file hello.c.

Tokens in C

A C program consists of various tokens and a token is either a keyword, an identifier, a constant, a string literal, or a symbol. For example, the following C statement consists of five tokens:

printf("Hello, World! \n");

The individual tokens are:

printf

(

"Hello, World! \n"

)

;

Semicolons ;

In C program, the semicolon is a statement terminator. That is, each individual statement must be ended with a semicolon. It indicates the end of one logical entity.

For example, following are two different statements:

printf("Hello, World! \n");

return 0;

Comments

Comments are like helping text in your C program and they are ignored by the compiler. They start with /* and terminates with the characters */ as shown below:

/* my first program in C */

You cannot have comments within comments and they do not occur within a string or character literals.

Identifiers

A C identifier is a name used to identify a variable, function, or any other user-defined item. An identifier starts with a letter A to Z or a to z or an underscore _ followed by zero or more letters, underscores, and digits (0 to 9).

C does not allow punctuation characters such as @, $, and % within identifiers. C is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in C. Here are some examples of acceptable identifiers:

mohd zara abc move_name a_123

myname50 _temp j a23b9 retVal

Keywords

The following list shows the reserved words in C. These reserved words may not be used as constant or variable or any other identifier names.

auto	else	Long	switch
break	enum	Register	typedef
case	extern	Return	union
char	float	Short	unsigned
const	for	Signed	void
continue	goto	Sizeof	volatile
default	if	Static	while
do	int	Struct	_Packed
double			