First C Program Example

Creating your first C program is a great way to get started with learning the language. The classic example for beginners is the "Hello, World!" program. This simple program will print "Hello, World!" to the screen.

```c
/*
 * Program: Hello World
 * Author: Mahesh Kumar
 * Date: 01/01/2024
 * Description: This program prints "Hello, World!" to the screen.
 */

#include <stdio.h>  // Includes the standard input/output library

int main() {
    printf("Hello, World!\n");  // Prints Hello, World! to the screen
    return 0;  // Returns 0 to indicate successful execution
}
```

Here's how to write and run this program in Turbo C:

Explanation of the Program

Documentation Section: Comments at the beginning describe the program, the author, the date, and the program's purpose.

Preprocessor Section: #include <stdio.h>: This directive includes the standard input/output library, which is necessary for using the printf function.

Main() Function:

- int main(): The main function is the entry point of any C program.
- printf("Hello, World!\n");: This function call prints "Hello, World!" to the screen.
- The \n is a newline character that moves the cursor to the next line.
- return 0; : This statement indicates that the program has terminated successfully.

Second C Program Example

c Copy code

```c
/*
 * Program: Number Display
 * Author: Mahesh Kumar
 * Date: 01/01/2024
 * Description: This program accepts a number and displays it to the user.
 */

#include <stdio.h>

void main(void) {
    int number;

    printf("Please enter a number: ");
    scanf("%d", &number);

    printf("You entered %d\n", number);

    // No need for return 0 in void main
}
```

Stepwise explanation of the above Program:

#include

- The part of the compiler which actually gets your program from the source file is called the preprocessor.
    - #include <stdio.h>
- #include is a pre-processor directive. It is not really part of our program, but instead it is an instruction to the compiler to make it do something. It tells the C compiler to include the contents of a file (in this case the system file called stdio.h).
- The compiler knows it is a system file, and therefore must be looked for in a special place, by the fact that the filename is enclosed in <> characters.

Introduction to 'C' Language - Module 2

<stdio.h>

- stdio.h is the name of the standard library definition file for all STanDard Input and Output functions.
- Your program will almost certainly want to send information to the screen and read things from the keyboard, and stdio.h is the name of the file in which the functions that we want to use are defined.
- The ".h" portion of the filename is the language extension, which denotes an include file.

void

- This literally means that this means nothing. In this case, it is referring to the function whose name follows.
- Void tells to C compiler that a given entity has no meaning, and produces no error.

main

- In this particular example, the only function in the program is called main.
- A C program is typically made up of large number of functions. Each of these is given a name by the programmer and they refer to each other as the program runs.
- C regards the name main as a special case and will run this function first i.e. the program execution starts from main.

(void)

- This is a pair of brackets enclosing the keyword void.
- It tells the compiler that the function main has no parameters.
- A parameter to a function gives the function something to work on.

{ (Brace)

- This is a brace (or curly bracket). As the name implies, braces come in packs of two - for every open brace there must be a matching close one.
- Braces allow us to group pieces of program together, often called a block.
- A block can contain the declaration of variable used within it, followed by a sequence of program statements.
- In this case the braces enclose the working parts of the function main.

; (semicolon)

- The semicolon marks the end of the list of variable names, and also the end of that declaration statement.
- All statements in C programs are separated by ";" (semicolon) characters.
- The ";" character is actually very important. It tells the compiler where a given statement ends. If the compiler does not find one of these characters where it expects to see one, then it will produce an error.

scanf

- In other programming languages, the printing and reading functions are a part of the language.
- In C this is not the case; instead they are defined as standard functions which are part of the language specification, but are not a part of the language itself.
- The standard input/output library contains a number of functions for formatted data transfer; the two we are going to use are scanf (scan formatted) and printf (print formatted).

printf

- The printf function is the opposite of scanf.
- It takes text and values from within the program and sends it out onto the screen.
- Just like scanf, it is common to all versions of C and just like scanf, it is described in the system file stdio.h.
- The first parameter to a printf is the format string, which contains text, value descriptions and formatting instructions.