Basic of C Language

History of C Language

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language. It keeps fluctuating at number one scale of popularity along with Java programming language, which is also equally popular and most widely used among modern software programmers.

Key advantages of learning C Programming

1. Easy to learn
2. Structured language
3. It produces efficient programs
4. It can handle low-level activities
5. It can be compiled on a variety of computer platforms

Facts about C
1. C was invented to write an operating system called UNIX.
2. C is a successor of B language which was introduced around the early 1970s.
3. The language was formalized in 1988 by the American National Standard Institute (ANSI).
4. The UNIX OS was totally written in C.
5. Today C is the most widely used and popular System Programming Language.
6. Most of the state-of-the-art software have been implemented using C.

Applications of C Programming

C was initially used for system development work, particularly the programs that make-up the operating system. C was adopted as a system development language because it produces code that runs nearly as fast as the code written in assembly language. Some examples of the use of C are -
Operating Systems
Language Compilers
Assemblers
Text Editors
Print Spoolers
Network Drivers
Modern Programs
Databases
Language Interpreters
Utilities
Audience

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

Hello World using C Programming.

```
#include <stdio.h>
void main()
{
  /* my first program in C */
  printf("Hello, World! \n");
}
```

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" −

Let us take a look at the 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 the 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 the main() function and returns the value 0.

Compile and Execute C Program

Let us see 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 have 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 see the output *"Hello World"* printed on the screen.

```
$ gcc hello.c
$ ./a.out
Hello, World!
```

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

C Data Types

In C programming, data types are declarations for variables. This determines the type and size of data associated with variables. For example,

```
int myVar;
```

Here, myVar is a variable of int (integer) type. The size of int is 4 bytes.

Basic types

Here's a table containing commonly used types in C programming for quick access.

Type	**Size (bytes)**	**Format Specifier**
int	at least 2, usually 4	%d, %i
char	1	%c
float	4	%f
double	8	%lf
short int	2 usually	%hd
unsigned int	at least 2, usually 4	%u
long int	at least 4, usually 8	%ld, %li
long long int	at least 8	%lld, %lli

Type	Size (bytes)	Format Specifier
unsigned long int	at least 4	%lu
unsigned long long int	at least 8	%llu
signed char	1	%c
unsigned char	1	%c
long double	at least 10, usually 12 or 16	%Lf

int

Integers are whole numbers that can have both zero, positive and negative values but no decimal values. For example, 0, -5, 10

We can use int for declaring an integer variable.

```
int id;
```

Here, id is a variable of type integer.

You can declare multiple variables at once in C programming. For example,

```
int id, age;
```

The size of int is usually 4 bytes (32 bits). And, it can take 232 distinct states from -2147483648 to 2147483647.

float and double

float and double are used to hold real numbers.

```
float salary;
double price;
```

In C, floating-point numbers can also be represented in exponential. For example,

```
float normalizationFactor = 22.442e2;
```

What's the difference between float and double?

The size of float (single precision float data type) is 4 bytes. And the size of double (double precision float data type) is 8 bytes.

char

Keyword char is used for declaring character type variables. For example,

```
char test = 'h';
```

The size of the character variable is 1 byte.

void

void is an incomplete type. It means "nothing" or "no type". You can think of void as **absent**.

For example, if a function is not returning anything, its return type should be void.

Note that, you cannot create variables of void type.

short and long

If you need to use a large number, you can use a type specifier long. Here's how:

```
long a;
long long b;
long double c;
```

Here variables a and b can store integer values. And, c can store a floating-point number.

If you are sure, only a small integer ([−32,767, +32,767] range) will be used, you can use short.

```
short d;
```

You can always check the size of a variable using the sizeof() operator.

```
#include <stdio.h>
int main() {
  short a;
  long b;
  long long c;
  long double d;

  printf("size of short = %d bytes\n", sizeof(a));
  printf("size of long = %d bytes\n", sizeof(b));
  printf("size of long long = %d bytes\n", sizeof(c));
  printf("size of long double= %d bytes\n", sizeof(d));
  return 0;
}
```

signed and unsigned

In C, signed and unsigned are type modifiers. You can alter the data storage of a data type by using them. For example,

```
unsigned int x;
int y;
```

Here, the variable x can hold only zero and positive values because we have used the unsigned modifier.