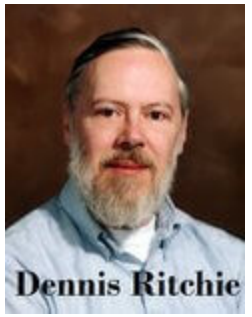
C Programming

Introduction

C programming language was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in the U.S.A.

Dennis Ritchie is known as the **founder of the c language**.

C is the widely used language. It provides many **features** that are given below.

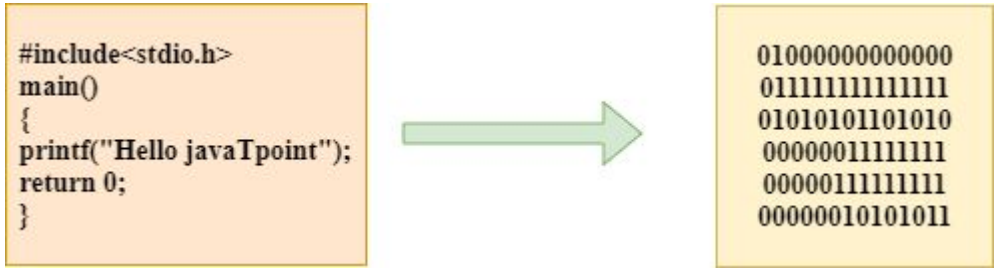


1. Simple
2. Machine Independent or Portable
3. Mid-level programming language
4. structured programming language
5. Rich Library
6. Memory Management
7. Fast Speed

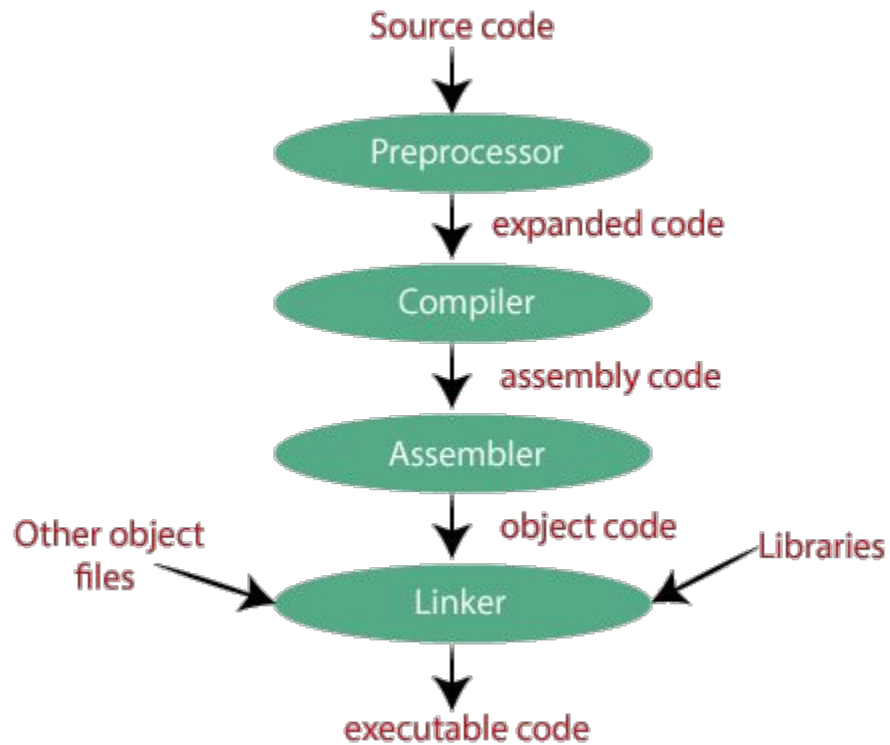
Compilation process in c

What is a compilation?

- The compilation is a process of converting the source code into object code. It is done with the help of the compiler. The compiler checks the source code for the syntactical or structural errors, and if the source code is error-free, then it generates the object code.



- The c compilation process converts the source code taken as input into the object code or machine code. The compilation process can be divided into four steps, i.e., Pre-processing, Compiling, Assembling, and Linking.


Source code
Preprocessor
expanded code
Compiler
assembly code
Assembler
object code
Other object files
Libraries
Linker
executable code

- Preprocessor

The source code is the code which is written in a text editor and the source code file is given an extension ".c". This source code is first passed to the preprocessor, and then the preprocessor expands this code. After expanding the code, the expanded code is passed to the compiler.

- Compiler

The code which is expanded by the preprocessor is passed to the compiler. **The compiler converts this code into assembly code.** Or we can say that the C compiler converts the pre-processed code into assembly code.

- Assembler

The assembly code is **converted into object code** by using an assembler. The name of the object file generated by the assembler is the same as the source file. The extension of the object file in DOS is '.obj,' and in UNIX, the extension is 'o'. If the name of the source file is **'hello.c',** then the name of the object file would be 'hello.obj'.

- Linker

Mainly, all the programs written in C use library functions. These library functions are pre-compiled, and the object code of these library files is stored with '.lib' (or '.a') extension.

The main working of the linker is to **combine the object code of library files with the object code of our program.**

Sometimes the situation arises when our program refers to the functions defined in other files; then linker plays a very important role in this. It links the object code of these files to our program.

Therefore, we conclude that the job of the linker is to link the object code of our program with the object code of the library files and other files.

The output of the linker is the executable file. The name of the executable file is the same as the source file but differs only in their extensions. In DOS, the extension of the executable file is '.exe', and in UNIX, the executable file can be named as 'a.out'. For example, if we are using printf() function in a program, then the linker adds its associated code in an output file.

Types of Variables in C

- There are many types of variables in c:

1. local variable
2. global variable
3. static variable
4. automatic variable
5. external variable

Local Variable

A variable that is declared inside the function or block is called a local variable.
It must be declared at the start of the block.

```
void function1(){
int x=10;//local variable
}
```

Global Variable

A variable that is declared outside the function or block is called a global variable. Any function can change the value of the global variable. It is available to all the functions.
It must be declared at the start of the block.

```
int value=20;//global variable
void function1(){
int x=10;//local variable
}
```

- Static Variable
- A variable that is declared with the static keyword is called static variable.
- It retains its value between multiple function calls.

```
void function1(){
int x=10;//local variable
static int y=10;//static variable
x=x+1;
y=y+1;
printf("%d,%d",x,y);
}
```

- If you call this function many times, the **local variable will print the same value** for each function call, e.g, 11,11,11 and so on. But the **static variable will print the incremented value** in each function call, e.g. 11, 12, 13 and so on.

- Automatic Variable
- All variables in C that are declared inside the block, are automatic variables by default. We can explicitly declare an automatic variable using **auto keyword**.

```
void main(){
int x=10;//local variable (also automatic)
auto int y=20;//automatic variable
}
```

- External Variable
- We can share a variable in multiple C source files by using an external variable. To declare an external variable, you need to use **extern keyword**.

1. **extern int** x=10;//external variable (also global)

- *program1.c*

```
1. #include "myfile.h"
2. #include <stdio.h>
3. void printValue(){
4.     printf("Global variable: %d", global_variable);
5. }
```