

## **Programming Language**

As we know, to communicate with a person, we need a specific language, similarly to communicate with computers, programmers also need a language is called Programming language.

### **What is Language?**

Language is a mode of communication that is used to share ideas, opinions with each other. For example, if we want to teach someone, we need a language that is understandable by both communicators.

### **What is a Programming Language?**

A programming language is a computer language that is used by programmers (developers) to communicate with computers. It is a set of instructions written in any specific language ( C, C++, Java, Python) to perform a specific task.

A programming language is mainly used to develop desktop applications, websites, and mobile applications.

## **Types of programming language**

### **1. Low-level programming language**

Low-level language is machine-dependent (0s and 1s) programming language. The processor runs low-level programs directly without the need of a compiler or interpreter, so the programs written in low-level language can be run very fast.

Low-level language is further divided into two parts -

#### **i. Machine Language**

Machine language is a type of low-level programming language. It is also called as machine code or object code. Machine language is easier to read because it is normally displayed in binary or hexadecimal form (base 16) form. It does not require a translator to convert the programs because computers directly understand the machine language programs.

The advantage of machine language is that it helps the programmer to execute the programs faster than the high-level programming language.

## **ii. Assembly Language**

Assembly language (ASM) is also a type of low-level programming language that is designed for specific processors. It represents the set of instructions in a symbolic and human-understandable form. It uses an assembler to convert the assembly language to machine language.

The advantage of assembly language is that it requires less memory and less execution time to execute a program.

## **2. High-level programming language**

High-level programming language (HLL) is designed for developing user-friendly software programs and websites. This programming language requires a compiler or interpreter to translate the program into machine language (execute the program).

The main advantage of a high-level language is that it is easy to read, write, and maintain.

High-level programming language includes Python, Java, JavaScript, PHP, C#, C++, Objective C, Cobol, Perl, Pascal, LISP, FORTRAN, and Swift programming language.

## **Programming Models**

### **i. Procedural Oriented programming language**

Procedural Oriented Programming (POP) language is derived from structured programming and based upon the procedure call concept. It divides a program into small procedures called routines or functions.

Procedural Oriented programming language is used by a software programmer to create a program that can be accomplished by using a programming editor like IDE, Adobe Dreamweaver, or Microsoft Visual Studio.

The advantage of POP language is that it helps programmers to easily track the program flow and code can be reused in different parts of the program.

The advantage of POP language is that it helps programmers to easily track the program flow and code can be reused in different parts of the program.

Example: C, FORTRAN, Basic, Pascal, etc.

## **ii. Object-Oriented Programming language**

Object-Oriented Programming (OOP) language is based upon the objects. In this programming language, programs are divided into small parts called objects. It is used to implement real-world entities like inheritance, polymorphism, abstraction, etc in the program to makes the program reusable, efficient, and easy-to-use.

The main advantage of object-oriented programming is that OOP is faster and easier to execute, maintain, modify, as well as debug.

Note: Object-Oriented Programming language follows a bottom-up approach.

Example: C++, Java, Python, C#, etc.

## **iii. Natural language**

Natural language is a part of human languages such as English, Russian, German, and Japanese. It is used by machines to understand, manipulate, and interpret human's language. It is used by developers to perform tasks such as translation, automatic summarization, Named Entity Recognition (NER), relationship extraction, and topic segmentation.

The main advantage of natural language is that it helps users to ask questions in any subject and directly respond within seconds.

## **3. Middle-level programming language**

Middle-level programming language lies between the low-level programming language and high-level programming language. It is also known as the intermediate programming language and pseudo-language.

A middle-level programming language's advantages are that it supports the features of high-level programming, it is a user-friendly language, and closely related to machine language and human language.

Example: C, C++, language

## **Most commonly used Programming Language**

As we all know, the programming language makes our life simpler. Currently, all sectors (like education, hospitals, banks, automobiles, and more ) completely depend upon the programming language.

There are dozens of programming languages used by the industries. Some most widely used programming languages are given below -

### **1. Python**



Python is one of the most widely used user-friendly programming languages. It is an open-source and easy to learn programming language developed in the 1990s. It is mostly used in Machine learning, Artificial intelligence, Big Data, GUI based desktop applications, and Robotics.

#### **Advantages**

- Python is easy to read, easy to understand, and easy to write.
- It integrates with other programming languages like C, C++, and Java.
- Python executes code line-by-line, so it is easy for the programmer to find the error that occurred in the code.
- Python is platform-independent means you can write code once and run it anywhere.

#### **Disadvantages**

- Python is not suitable for developing mobile applications and games.
- Python works with the interpreter. That's why it is slower than other programming languages like C and C++.

### **2. Java**



Java is a simple, secure, platform-independent, reliable, architecture-neutral high-level programming language developed by Sun Microsystems in 1995. Now, Java is owned by Oracle. It is mainly used to develop bank, retail, information technology, android, big data, research community, web, and desktop applications.

## Advantages

- Java is easy to write, compile, learn, and debug as compared to other programming languages.
- It provides an ability to run the same program on different platforms.
- It is a highly secured programming language because in java, there is no concept of explicit pointers.
- It is capable of performing multiple tasks at the same time.

## Disadvantages

- Java consumes more memory and slower than other programming languages like C or C++.
- It does not provide a backup facility.

## 3. C



C is a popular, simple, and flexible general-purpose computer programming language. Dennis M Ritchie develops it in 1972 at AT&T. It is a combination of both low-level programming language as well as a high-level programming language. It is used to design applications like Text Editors, Compilers, Network devices, and many more.

## Advantages

- C language is easy to learn.
- It is fast, efficient, portable, easy to extend, powerful, and flexible programming language.
- It is used to perform complex calculations and operations such as MATLAB.
- It provides dynamic memory allocation to allocate memory at the run time.

## Disadvantages

- In the C programming language, it is very difficult to find the errors.
- C does not support the concepts of constructors, destructors, abstraction, polymorphism, encapsulation, and namespace like OOPs.

## 4. C++



C++ is one of the thousands of programming languages that we use to develop software. C++ programming language is developed by Bjarne Stroustrup in 1980. It is similar to the C programming language but also includes some additional features such as exception handling, object-oriented programming, type checking, etc.

### **Advantages**

- C++ is a simple and portable structured programming language.
- It supports OOPs features such as Abstraction, Inheritance, Encapsulation.
- It provides high-level abstraction and useful for a low-level programming language, and more efficient for general-purpose.
- C++ is more compatible with the C language.

### **Disadvantages**

- C++ programming language is not secured as compared to other programming languages like Java or Python.
- C++ can not support garbage collection.
- It is difficult to debug large as well as complex web applications.

## **5. C#**



C# (pronounced as C sharp) is a modern, general-purpose, and object-oriented programming language used with XML based Web services on the .NET platform. It is mainly designed to improve productivity in web applications. It is easier to learn for those users who have sufficient knowledge of common programming languages like C, C++, or Java.

### **Advantages**

- C# is a modern, type-safe, easy, fast, and open-source programming language that is easily integrated with Windows.
- The maintenance of C# (C sharp) is lower than the C++ programming language.
- C# is a pure object-oriented programming language.
- C# includes a strong memory backup facility. That's why it avoids the problem of memory leakage.

## Disadvantages

- C# is less flexible because it is completely based on Microsoft .Net framework.
- In C#, it is difficult to write, understand, debug, and maintain multithreaded applications.

## 6. JavaScript



JavaScript is a type of scripting language that is used on both client-side as well as a server-side. It is developed in the 1990s for the Netscape Navigator web browser. It allows programmers to implement complex features to make web pages alive. It helps programmers to create dynamic websites, servers, mobile applications, animated graphics, games, and more.

### Advantage

- JavaScript helps us to add behavior and interactivity on the web page.
- It can be used to decrease the loading time from the server.
- It has the ability to create attractive, dynamic websites, and rich interfaces.
- JavaScript is a simple, versatile, and lightweight programming language.
- JavaScript and its syntax are easy to understand.

### Disadvantage

- JavaScript is completely based on the browser.
- It does not support multiple inheritance.
- It is less secure compared to other programming languages.

## 7. R



Currently, R programming is one of the popular programming languages that is used in data analytics, scientific research, machine learning algorithms, and statistical computing. It is developed in 1993 by Ross Ihaka and Robert Gentleman. It helps marketers and data scientists to easily analyze, present, and visualize data.

### Advantages

- R programming provides extensive support for Data Wrangling.
- It provides an easy-to-use interface.

- It runs on any platform like Windows, Linux, and Mac.
- It is an open-source and platform-independent programming language.

### **Disadvantages**

- R programming does not support 3D graphics.
- It is slower than other programming languages.

## **8. PHP**



PHP stands for Hypertext Preprocessor. It is an open-source, powerful server-side scripting language mainly used to create static as well as dynamic websites. It is developed by Rasmus Laird in 1994. Inside the php, we can also write HTML, CSS, and JavaScript code. To save php file, file extension .php is used.

### **Advantages**

- PHP is a more secure and easy-to-use programming language.
- It supports powerful online libraries.
- It can be run on a variety of operating systems such as Windows, Linux, and Mac.
- It provides excellent compatibility with cloud services.

### **Disadvantages**

- PHP is not capable of handling a large number of applications and not suitable for large applications.
- It is quite difficult to maintain.

## **9. Go**



Go or Golang is an open-source programming language. It is used to build simple, reliable, and efficient software. It is developed by Robert Griesemer, Rob Pike, and Ken Thompson in 2007.

### **Advantages**

- Go language is easy-to-learn and use.
- It comes with the in-built testing tools.
- Go is a fast programming language.

### **Disadvantages**

- Go language does not support generics.



- It does not support error handling.
- It supports a lack of frameworks.

## **10. Ruby**



Ruby is an open-source, general-purpose, and pure object-oriented programming language released in 1993. It is used in front-end and back-end web development. It is mainly designed to write CGI (Common Gateway Interface) scripts.

### **Advantages**

- Ruby supports various GUI (Graphical User Interface) tools like GTK and OpenGL.
- It is used to develop both internet as well as intranet applications.
- The code written in Ruby is small and contains less number of lines.

### **Disadvantages**

- Ruby is slower than other programming languages.
- It is very difficult for programmers to debug the code written in Ruby.

### **What is a program?**

A computer program is a set of instructions and as a term it can be used as a verb as well as a noun. In terms of a verb, it is used as a process of creating a software program by using programming language. In terms of a noun, an application, program, or application software is used to perform a specific task on the computer. For example, Microsoft PowerPoint is an application, which provides a way to create documents related to the presentation. Furthermore, a browser is also an application, which allows us to browse any website.

### **Difference between Applications and programs**

All applications can be called a program, but a program cannot be an application. An application is a collection of programs that are designed to help the end-users to achieve a purpose. These programs communicate with each other to perform tasks or activities. It cannot exist without a program and functions to carry out end-user commands. Whereas, a program is a collection of instructions that describes the computer what task to perform.

### **What is the purpose of a program?**

The program enables the computer to perform a particular operation. As without application software (programs), a computer is able to operate with the operating system, but it cannot perform any specific task. For example, if you want to create a Word document, you have to install Microsoft Word on your computer. It is a program or application software that instructs the computer how to create, edit, and save a document or a file.

### **Basic functions of a program**

The function of a program depends upon the type of program. For example, the function of the Microsoft Excel program is to create, edit, and view documents related to calculation and data analysis, etc. The function of an internet browser is to find information on the World Wide Web and display it on the screen. Basically, a program is designed to execute a particular task or function. For example, an Excel program is able to create a document, but it cannot find the information on the World Wide Web like a browser.

### **What was the first program?**

Tom Kilburn wrote the first software program to hold in electronic memory. It was successfully executed at the University of Manchester, England, on 21 June 1948. This program was computed as the greatest factor of the integer  $2^{18} = 262,144$ . The computer was called the small Scale Experimental Machine (SSEM), which was known as the Manchester Baby. This occurrence is considered as the birth of the first software.

### **Examples of computer programs**

Today, there are various types of programs available for mobile phones, computers, and other devices. The below table contains some examples of programs with their category and brief description.

Program	Category	Description
Google Chrome	Internet Browser	It is an internet browser that was introduced by Google on 11 December 2008. It is used to retrieve the information available on the World Wide Web and display it on the device screen. It provides various types of features to help the users, such as tabbed browsing, synchronization with Google services and accounts, and

		spell check and automatic translation of web pages. Additionally, it has a search bar or Omnibox, which allows users to search for any query.
C	Programming Language	<p>It is a general-purpose programming language, which is used to develop the software. It was released in 1972 after it was developed at Bell Labs by Dennis Ritchie. It is widely used for writing complex programs such as Python, Git, Oracle database, etc.</p> <p>Furthermore, it includes more features such as simple and efficient, portability, rich library, extensible, high-speed, and more.</p>
Skype	Chat and VoIP	Skype is a program that allows users to chat and make VOIP (voice over internet protocol) calls anywhere in the World. A Skype user can call for free to another Skype user anywhere in the world.
Adobe Photoshop	Photo Editor	It is an image editing program, runs on MacOS or Windows computers. It supports all types of file formats as well as JPEG, Targa, GIF, BMP, HEIF, etc. It provides users many tools to create, edit, and enhance the quality of an image, including a real-life painting, create an animated GIF from an image or short video files.
Microsoft Word	Word processor	<p>It is a word processor program. It was developed by Charles Simonyi and Richard Brodie, and published by Microsoft. It was introduced on 25 October 1983. You can use the Word program on Microsoft Windows, Android, Apple iOS, and Apple macOS.</p> <p>Furthermore, it can also be run on the Linux OS with the help of WINE.</p>

FileZilla	FTP	The FileZilla is an open-source software program that allows users to transfer files from a local computer to a remote computer. It is usable as a client version as well as a server version. It includes more features, including the important features such as Transfer Queue, Site Manager, File, and Folder View, Directory Comparison.
Microsoft Excel	Spreadsheet	It is a software program, which provides a spreadsheet to create documents related to calculation, data analysis, and more. It is developed by Microsoft on 30 September 1985. When it was in the developing phase, its code name was Odyssey. If you want to create a monthly budget report, salary sheet, Bill order, and more, you can use the Microsoft Excel program.
Microsoft PowerPoint	Presentation	It is a part of Microsoft Office that is bundled with Microsoft Word and Excel. It is used to create a presentation by creating different types of slides. It is widely used in school and business presentations. For example, if you want to create a presentation of your document to show at your college or at any organization, you can use the Microsoft PowerPoint program.
Mozilla Thunderbird	E-mail client	It is an open-source e-mail client that allows users to send, receive, and manage their e-mail on Microsoft Windows, Linux, MacOS, and other supported systems. It provides users the option to retrieve e-mail from their e-mail provider with the help of IMAP or POP3, and users can send an e-mail by using Simple Mail Transfer Protocol (SMTP).
Norton Anti-Virus	Antivirus	It is an anti-virus software product that is developed for computer security by Symantec Corporation in 1991. It uses heuristics and signatures to detect viruses. Furthermore, it is distributed by

		Symantec as a download, copy, a box, and OEM software.
Audacity	Audio software	It is an open-source software program, which enables users to record sound, including edit sound clips. It can run on the MacOS, Linux, and Windows operating systems. It is available for free to use as per the General Public License (GPL).
Adobe Acrobat	PDF reader	It is an application software introduced by Adobe, which is used to create, view, manage, print, and manipulates files in PDF (Portable Document Format).
Comm Central	Fax/Voice/Phone	It is a program, which enables users to receive faxes, including receiving voicemail on their personal computers.
Adobe Dreamweaver	HTML editor	It is a software program that is used to design web pages and released by Macromedia in 1997. It is a full-fledged HTML and programming editor, which offers users WYSIWYG (what you see is what you get) user-interface to create and edit web pages. It supports HTML, CSS, JavaScript, and XML as well as human languages such as English, French, Spanish, Chinese, Japanese, Russian, etc.

## Compiler vs Interpreter: Complete Difference Between Compiler and Interpreter

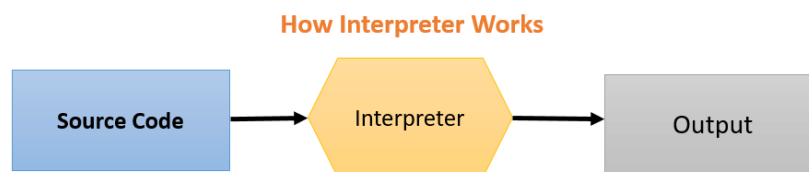
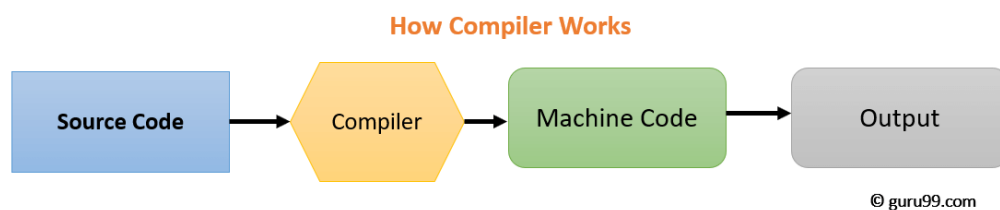
### What is Compiler?

A compiler is a computer program that transforms code written in a high-level programming language into the machine code. It is a program which translates the human-readable code to a language a computer processor understands (binary 1 and 0 bits). The computer processes the machine code to perform the corresponding tasks.

A compiler should comply with the syntax rule of that programming language in which it is written. However, the compiler is only a program and cannot fix errors found in that program. So, if you make a mistake, you need to make changes in the syntax of your program. Otherwise, it will not compile.

### What is Interpreter?

An interpreter is a computer program, which converts each high-level program statement into the machine code. This includes source code, pre-compiled code, and scripts. Both compiler and interpreters do the same job which is converting higher level programming language to machine code. However, a compiler will convert the code into machine code (create an exe) before program run. Interpreters convert code into machine code when the program is run.



### KEY DIFFERENCE

- Compiler transforms code written in a high-level programming language into the machine code, at once, before program runs, whereas an Interpreter converts each high-level program statement, one by one, into the machine code, during program run.
- Compiled code runs faster while interpreted code runs slower.
- Compiler displays all errors after compilation, on the other hand, the Interpreter displays errors of each line one by one.
- Compiler is based on translation linking-loading model, whereas Interpreter is based on Interpretation Method.
- Compiler takes an entire program whereas the Interpreter takes a single line of code.

### Difference Between Compiler and Interpreter

Basis of difference	Compiler	Interpreter
Programming Steps	<p>Create the program.</p> <p>Compiler will parse or analyses all of the language statements for its correctness. If incorrect, throws an error</p> <p>If no error, the compiler will convert source code to machine code.</p> <p>It links different code files into a runnable program(know as exe)</p> <p>Run the Program</p>	<p>Create the Program</p> <p>No linking of files or machine code generation</p> <p>Source statements executed line by line DURING Execution</p>
Advantage	The program code is already translated into machine code. Thus, its code execution time is less.	Interpreters are easier to use, especially for beginners.
Disadvantage	You can't change the program without going back to the source code.	Interpreted programs can run on computers that have the corresponding interpreter.
Machine code	Store machine language as machine code on the disk	Not saving machine code at all.
Running time	Compiled code runs faster	Interpreted code runs slower
Model	It is based on language translation linking-loading model.	It is based on Interpretation Method.
Program generation	Generates output program (in the form of exe) which can be run independently from the original program.	Do not generate output program. So they evaluate the source program at every time during execution.
Execution	Program execution is separate from the compilation. It is performed only after the entire output program is compiled.	Program Execution is a part of Interpretation process, so it is performed line by line.

Basis of difference	Compiler	Interpreter
Memory requirement	Target program execute independently and do not require the compiler in the memory.	The interpreter exists in the memory during interpretation.
Best suited for	Bounded to the specific target machine and cannot be ported. C and C++ are a most popular a programming language which uses compilation model.	For web environments, where load times are important. Due to all the exhaustive analysis is done, compiles take relatively larger time to compile even small code that may not be run multiple times. In such cases, interpreters are better.
Code Optimization	The compiler sees the entire code upfront. Hence, they perform lots of optimizations that make code run faster	Interpreters see code line by line, and thus optimizations are not as robust as compilers
Dynamic Typing	Difficult to implement as compilers cannot predict what happens at turn time.	Interpreted languages support Dynamic Typing
Usage	It is best suited for the Production Environment	It is best suited for the program and development environment.
Error execution	Compiler displays all errors and warning at the compilation time. Therefore, you can't run the program without fixing errors	The interpreter reads a single statement and shows the error if any. You must correct the error to interpret next line.
Input	It takes an entire program	It takes a single line of code.
Output	Compiler generates intermediate machine code.	Interpreter never generates any intermediate machine code.
Errors	Display all errors after, compilation, all at the same time.	Displays all errors of each line one by one.
Pertaining Programming languages	C, C++, C#, Scala, Java all use compiler.	PHP, Perl, Ruby uses an interpreter.



## Role of Compiler

- Compilers reads the source code, outputs executable code
- Translates software written in a higher-level language into instructions that computer can understand. It converts the text that a programmer writes into a format the CPU can understand.
- The process of compilation is relatively complicated. It spends a lot of time analyzing and processing the program.
- The executable result is some form of machine-specific binary code.

## Role of Interpreter

- The interpreter converts the source code line-by-line during RUN Time.
- Interpreter completely translates a program written in a high-level language into machine level language.
- Interpreter allows evaluation and modification of the program while it is executing.
- Relatively less time spent for analyzing and processing the program
- Program execution is relatively slow compared to compiler

## HIGH-LEVEL LANGUAGES

High-level languages, like C, C++, JAVA, etc., are very near to English. It makes programming process easy. However, it must be translated into machine language before execution. This translation process is either conducted by either a compiler or an interpreter. Also known as **source code**.

## MACHINE CODE

Machine languages are very close to the hardware. Every computer has its machine language. A machine language programs are made up of series of binary pattern. (Eg. 110110) It represents the simple operations which should be performed by the computer. Machine language programs are executable so that they can be run directly.

## OBJECT CODE

On compilation of source code, the machine code generated for different processors like Intel, AMD, an ARM is different. To make code portable, the source code is first converted to Object Code. It is an

intermediary code (similar to machine code) that no processor will understand. At run time, the object code is converted to the machine code of the underlying platform.

### **Java is both Compiled and Interpreted.**

To exploit relative advantages of compilers and interpreters some programming language like Java are both compiled and interpreted. The Java code itself is compiled into Object Code. At run time, the JVM interprets the Object code into machine code of the target computer.

### **Modular Approach in Programming**

Modular programming is the process of subdividing a computer program into separate sub-programs. A module is a separate software component. It can often be used in a variety of applications and functions with other components of the system.

Some programs might have thousands or millions of lines and to manage such programs it becomes quite difficult as there might be too many of syntax errors or logical errors present in the program, so to manage such type of programs concept of modular programming approached.

Each sub-module contains something necessary to execute only one aspect of the desired functionality.

Modular programming emphasis on breaking of large programs into small problems to increase the maintainability, readability of the code and to make the program handy to make any changes in future or to correct the errors.

Points which should be taken care of prior to modular program development:

- Limitations of each and every module should be decided.
- In which way a program is to be partitioned into different modules.
- Communication among different modules of the code for proper execution of the entire program.

### **Advantages of Using Modular Programming Approach –**

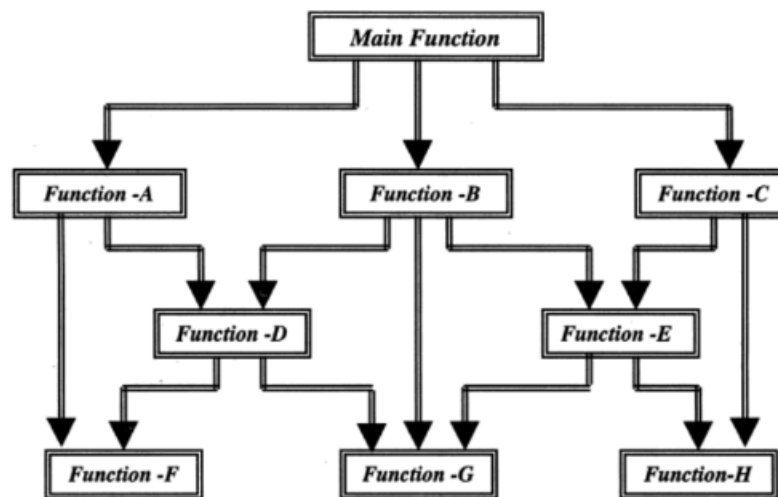
- **Ease of Use** : This approach allows simplicity, as rather than focusing on the entire thousands and millions of lines code in one go we can access it in the form of modules. This allows ease in debugging the code and prone to less error.

- **Reusability** : It allows the user to reuse the functionality with a different interface without typing the whole program again.
- **Ease of Maintenance** : It helps in less collision at the time of working on modules, helping a team to work with proper collaboration while working on a large application.

### Example of Modular Programming in C

C is called a structured programming language because to solve a large problem, C programming language divides the problem into smaller modules called functions or procedures each of which handles a particular responsibility. The program which solves the entire problem is a collection of such functions.

Module is basically a set of interrelated files that share their implementation details but hide it from the outside world. Each function defined in C by default is globally accessible. This can be done by including the header file in which implementation of the function is defined.



### Structured Programming Approach with Advantages and Disadvantages

Structured Programming Approach, as the word suggests, can be defined as a programming approach in which the program is made as a single structure. It means that the code will execute the instruction by instruction one after the other. It doesn't support the possibility of jumping from one instruction to some other with the help of any statement like GOTO, etc.

Therefore, the instructions in this approach will be executed in a serial and structured manner.

The languages that support Structured programming approach are:

- C
- C++
- Java
- C#
- ..etc

On the contrary, in the Assembly languages like Microprocessor 8085, etc, the statements do not get executed in a structured manner. It allows jump statements like GOTO. So the program flow might be random.

The structured program mainly consists of three types of elements:

- Selection Statements
- Sequence Statements
- Iteration Statements

The structured program consists of well structured and separated modules. But the entry and exit in a Structured program is a single-time event. It means that the program uses single-entry and single-exit elements. Therefore a structured program is well maintained, neat and clean program. This is the reason why the Structured Programming Approach is well accepted in the programming world.

#### **Advantages of Structured Programming Approach:**

- Easier to read and understand
- User Friendly
- Easier to Maintain
- Mainly problem based instead of being machine based
- Development is easier as it requires less effort and time
- Easier to Debug

- Machine-Independent, mostly.

#### **Disadvantages of Structured Programming Approach:**

- Since it is Machine-Independent, So it takes time to convert into machine code.
- The converted machine code is not the same as for assembly language.
- The program depends upon changeable factors like data-types. Therefore it needs to be updated with the need on the go.
- Usually the development in this approach takes longer time as it is language-dependent. Whereas in the case of assembly language, the development takes lesser time as it is fixed for the machine.

#### **C Language**

The C Language is developed by Dennis Ritchie for creating system applications that directly interact with the hardware devices such as drivers, kernels, etc.

C programming is considered as the base for other programming languages, that is why it is known as mother language.

#### **It can be defined by the following ways:**

- Mother language
- System programming language
- Procedure-oriented programming language
- Structured programming language
- Mid-level programming language

#### **1) C as a mother language**

C language is considered as the mother language of all the modern programming languages because most of the compilers, JVMs, Kernels, etc. are written in C language, and most of the programming languages follow C syntax, for example, C++, Java, C#, etc.

It provides the core concepts like the array, strings, functions, file handling, etc. that are being used in many languages like C++, Java, C#, etc.

## **2) C as a system programming language**

A system programming language is used to create system software. C language is a system programming language because it can be used to do low-level programming (for example driver and kernel). It is generally used to create hardware devices, OS, drivers, kernels, etc. For example, Linux kernel is written in C.

It can't be used for internet programming like Java, .Net, PHP, etc.

## **3) C as a procedural language**

- A procedure is known as a function, method, routine, subroutine, etc. A procedural language specifies a series of steps for the program to solve the problem.
- A procedural language breaks the program into functions, data structures, etc.
- C is a procedural language. In C, variables and function prototypes must be declared before being used.

## **4) C as a structured programming language**

A structured programming language is a subset of the procedural language. Structure means to break a program into parts or blocks so that it may be easy to understand.

In the C language, we break the program into parts using functions. It makes the program easier to understand and modify.

## **5) C as a mid-level programming language**

C is considered as a middle-level language because it supports the feature of both low-level and high-level languages. C language program is converted into assembly code, it supports pointer arithmetic (low-level), but it is machine independent (a feature of high-level).

A Low-level language is specific to one machine, i.e., machine dependent. It is machine dependent, fast to run. But it is not easy to understand.

A High-Level language is not specific to one machine, i.e., machine independent. It is easy to understand.

## C Program

**File: main.c**

```
#include <stdio.h>
int main() {
    printf("Hello C Programming\n");
    return 0;
}
```

## History of C Language



- History of C language is interesting to know. Here we are going to discuss a brief history of the c language.
- 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.
- It was developed to overcome the problems of previous languages such as B, BCPL, etc.
- Initially, C language was developed to be used in UNIX operating system. It inherits many features of previous languages such as B and BCPL.
- Let's see the programming languages that were developed before C language.

Language	Year	Developed By
----------	------	--------------

Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee



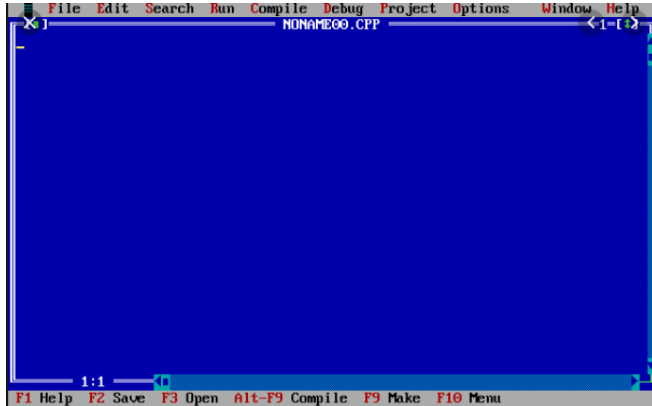
## C Compilers

A compiler transforms and translates a high-end language to machine (low level) understandable language. The compiling process does basic translation mechanisms and also error detection. The front end compilation includes lexical, syntax, and semantic analysis. And at the back end, the compilation does the code generation and optimization. So there are many compilers in C like BDS, Clang, GCC, Interactive C, Lattice, Portable C Compiler, Visual Express, etc.

### Top Compilers of C:

#### 1) Borland Turbo C

Turbo C is one of the basic and popular compilers for the C programming language. This was first introduced in 1987; it was popular for its small size, compilation speed, and low price. Once Turbo C++ got released in 1990, both the compilers are merged and the name Turbo C got discontinued. In 2006, Embarcadero Technologies had re-released Turbo C as freeware.



#### 2) Tiny C Compiler

The Tiny C Compiler is designed to work on slow computers with little disk space. This is an ARM processor C compiler. This compiler started its support to Windows from 2005.

Its file size is small and according to the owner of this compiler (Fabrice Bellard). The fastness of this compiler is around nine times faster than GCC. The compilation, assembling and linking of code were the main attributes considered for measuring the fastness of this compiler.

This compiler had included many compiler-specific features to boost up the optional memory, bound checker and had greater code stability.

This compiler allows automatic execution of programs during the compile-time only using command line arguments. This way, programs are executed under UNIX, using shell scripts. The latest version was released on December 2017.

### **3) Portable C Compiler**

The Portable C Compiler (PCC) was a very early used and established compiler for the C programming language that is almost around mid-1970. This compiler had a long life span. This was prevalent during a period in such a way that many of the C compilers were based on it. The advantages of PCC depended on its capabilities and probability predictions. PCC compiler was made such that source files were machine-dependent, not all but only a few of them. It can detect syntax errors and can perform perfect validity checks. A new version of PCC was released on 10 December 2014.

### **4) GCC**

GNU Compiler Collection is the compiler produced by the GNU Project. This supports many programming languages and it is a free software foundation under the General Public License. This compiler was first released in 1987 and it supported only C- Programming language during the start. Slowly it expanded to C++, Java, Android, and IOS. Here, each of the different language compilers has its own program that reads the code written and sends the machine code as the output. All of these have a common internal structure. When a high-level language is written, as per the language it is written, the compiler parses the code in that language and produces an abstract syntax tree. GCC uses LALR parsers, but slowly switched to recursive-descent parsers for C in 2006. Coming to the optimization part, as already known this can occur during any phase of the compilation. However, here the bulk optimizations are performed

before the code generation and after the syntax, semantic analysis. Below are a few of the optimizations performed by GCC.

1. It can eliminate the Dead Code pieces.
2. It can eliminate the redundancy at the code level.
3. Replacement of Aggregates with respect to the scalar level.
4. Can perform optimizations with Arrays.

In GCC back end is specified by preprocessor macros and functions specific to a particular architecture. This code is generally built by first calling a small snippet code which is associated with each pattern and generate instructions from the instruction set. It is done using registers, offsets, and addresses that are chosen during the re-load phase. The current version of GCC is 9.2, which was released on August 12, 2019.

## **5) Clang**

Clang; including C, is also a compiler for C++, Objective-C, and objective-C++ programming languages. This compiler uses LLVM for the back end code related compilations. This compiler has been designed to act as a replacement for the GCC by supporting many of its compilation flags and language extensions.

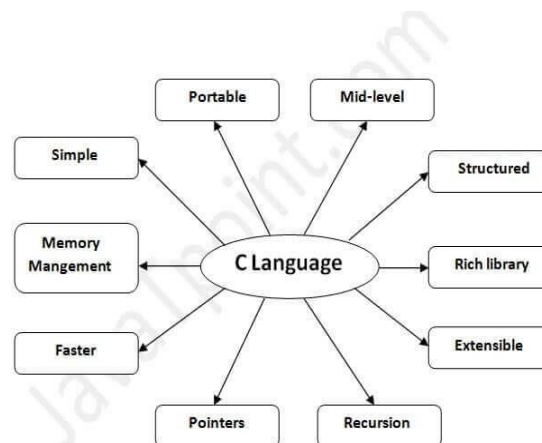
Clang has many contributors including Apple, Microsoft, Google, Sony, and Intel. It is open-source software. LLVM was first used by GCC for the front end compilation, but GCC had caused some problems for developers at Apple, as the source code is large and difficult to use. So, they had come up with Clang.

One of the major goals for Clang is to provide library-based architecture. It is designed to keep more information during the compilation process than GCC. This also helps to preserve the overall shape of the original code.

The error report generated by Clang during compilation is always in a detailed and specific in a machine-readable format. Clang had always aimed to reduce the over usage of memory space

and increase the compilation speed as compared with GCC, and due to these qualities, it had become one of the fastest-growing used compilers during a point of time. But over a period the performance of Clang started to come down. The reports told the performance had lagged with almost large differences as compared with GCC and started to have slower performance. The most recent comparisons indicate that both the compilers had come up and increased their performance and once again creating great competition between them. **Yet, GCC remains to top the list.**

### Features of C Language



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

- Simple
- Machine Independent or Portable
- Mid-level programming language
- structured programming language
- Rich Library
- Memory Management
- Fast Speed
- Pointers
- Recursion
- Extensible

#### 1) Simple

C is a simple language in the sense that it provides a structured approach (to break the problem into parts), the rich set of library functions, data types, etc.

## **2) Machine Independent or Portable**

Unlike assembly language, c programs can be executed on different machines with some machine specific changes. Therefore, C is a machine independent language.

## **3) Mid-level programming language**

Although, C is intended to do low-level programming. It is used to develop system applications such as kernel, driver, etc. It also supports the features of a high-level language. That is why it is known as mid-level language.

## **4) Structured programming language**

C is a structured programming language in the sense that we can break the program into parts using functions. So, it is easy to understand and modify. Functions also provide code reusability.

## **5) Rich Library**

C provides a lot of inbuilt functions that make the development fast.

## **6) Memory Management**

It supports the feature of dynamic memory allocation. In C language, we can free the allocated memory at any time by calling the free() function.

## **7) Speed**

The compilation and execution time of C language is fast since there are lesser inbuilt functions and hence the lesser overhead.

## **8) Pointer**

C provides the feature of pointers. We can directly interact with the memory by using the pointers. We can use pointers for memory, structures, functions, array, etc.

## **9) Recursion**

In C, we can call the function within the function. It provides code reusability for every function. Recursion enables us to use the approach of backtracking.

## **10) Extensible**

C language is extensible because it can easily adopt new features.

### **Where is C used?**

- 'C' language is widely used in embedded systems.
- It is used for developing system applications.
- It is widely used for developing desktop applications.
- Most of the applications by Adobe are developed using 'C' programming language.
- It is used for developing browsers and their extensions. Google's Chromium is built using 'C' programming language.
- It is used to develop databases. MySQL is the most popular database software which is built using 'C'.
- It is used in developing an operating system. Operating systems such as Apple's OS X, Microsoft's Windows, and Symbian are developed using 'C' language. It is used for developing desktop as well as mobile phone's operating system.
- It is used for compiler production.
- It is widely used in IOT applications.

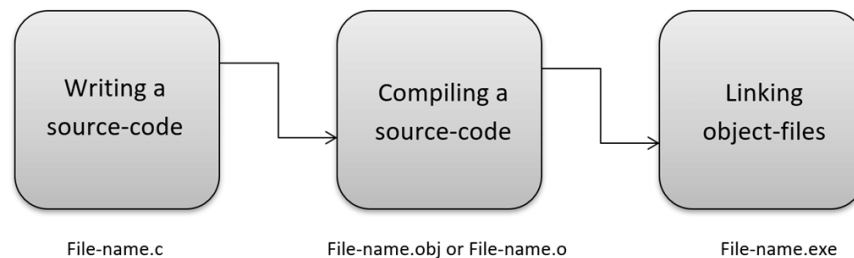
### **Why learn C Language?**

- 'C' is a base language for many programming languages. So, learning 'C' as the main language will play an important role while studying other programming languages. It shares the same concepts such as data types, operators, control statements and many more. 'C' can be used widely in various applications. It is a simple language and provides faster execution. There are many jobs available for a 'C' developer in the current market.

- 'C' is a structured programming language in which program is divided into various modules. Each module can be written separately and together it forms a single 'C' program. This structure makes it easy for testing, maintaining and debugging processes.
- 'C' contains 32 keywords, various data types and a set of powerful built-in functions that make programming very efficient.
- Another feature of 'C' programming is that it can extend itself. A 'C' program contains various functions which are part of a library. We can add our features and functions to the library. We can access and use these functions anytime we want in our program. This feature makes it simple while working with complex programming.
- Various compilers are available in the market that can be used for executing programs written in this language.
- It is a highly portable language which means programs written in 'C' language can run on other machines. This feature is essential if we wish to use or execute the code on another computer.

### How C Programming Language Works?

C is a compiled language. A compiler is a special tool that compiles the program and converts it into the object file which is machine readable. After the compilation process, the linker will combine different object files and creates a single executable file to run the program. The following diagram shows the execution of a 'C' program



### Structure of a C Program

There are seven main sections to a basic c program.

The six sections are,

- Documentation
- Link
- Definition
- Global Declarations
- Main functions
- Subprograms

### **Documentation Section**

The documentation section is the part of the program where the programmer gives the details associated with the program. He usually gives the name of the program, the details of the author and other details like the time of coding and description. It gives anyone reading the code the overview of the code.

Example

```
/**  
* File Name: Helloworld.c  
* Author: Manthan Naik  
* date: 09/08/2019  
* description: a program to display hello world  
*           no input needed  
*/
```

Moving on to the next bit of this basic structure of a C program article,

### **Link Section**

This part of the code is used to declare all the header files that will be used in the program. This leads to the compiler being told to link the header files to the system libraries.

Example



```
1    #include<stdio.h>
```

### **Definition Section**

In this section, we define different constants. The keyword define is used in this part.

```
1    #define PI=3.14
```

### **Global Declaration Section**

This part of the code is the part where the global variables are declared. All the global variable used are declared in this part. The user-defined functions are also declared in this part of the code.

```
1    float area(float r);
```

```
2    int a=7;
```

### **Main Function Section**

Every C-programs needs to have the main function. Each main function contains 2 parts. A declaration part and an Execution part. The declaration part is the part where all the variables are declared. The execution part begins with the curly brackets and ends with the curly close bracket. Both the declaration and execution part are inside the curly braces.

```
int main(void)

{

int a=10;

printf(" %d", a);

return 0;

}
```

### **Sub Program Section**

All the user-defined functions are defined in this section of the program.

```
int add(int a, int b)

{

return a+b;

}
```

### **Sample Program**

The C program here will find the area of a circle using a user-defined function and a global variable pi holding the value of pi

```
* File Name: areaofcircle.c

* Author: Manthan Naik

* date: 09/08/2019

* description: a program to calculate area of circle

*user enters the radius

**/

#include<stdio.h>//link section

#define pi 3.14;//defination section

float area(float r);//global declaration

int main();//main function

{

float r;

printf(" Enter the radius:n");
```

```
scanf("%f",&r);

printf("the area is: %f",area(r));

return 0;

}

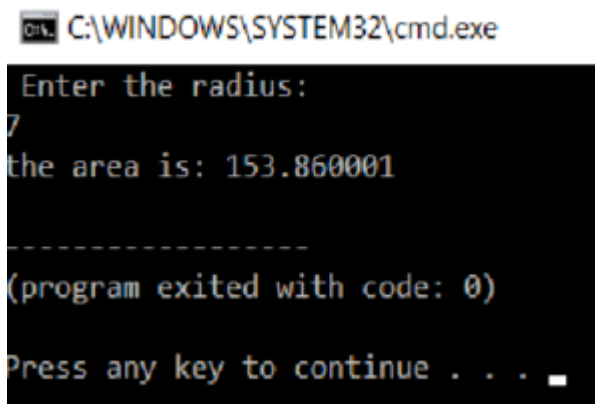
float area(float r)

{

return pi * r * r;//sub program

}
```

### Output



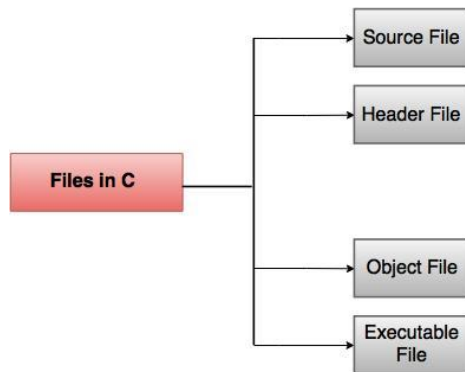
```
C:\WINDOWS\SYSTEM32\cmd.exe

Enter the radius:
7
the area is: 153.860001

-----
(program exited with code: 0)
Press any key to continue . . .
```

### Files used in C Programming

A C program uses four types of files as follows:



### Source Code File

- This file includes the source code of the program.
- The extension for these kind of files are '.c'. It defines the main and many more functions written in C.
- `main()` is the starting point of the program. It may also contain other source code files.

### Header Files

They have an extension '.h'. They contain the C function declarations and macro definitions that are shared between various source files.

### Advantages of header files:

At times the programmer may want to use the same subroutines for different programs. To do this, he would just compile the code of the subroutine once and link to the resulting object file in any file in which the functionalities of this subroutine are required.

At times the programmer may want to change or add the subroutines and reflect those changes in all the programs. For doing this, he will have to only change the source file for the subroutines, recompile the source code and then recompile and re-link the program.

This tells us that including a header file will make it easier at all levels of the program. If we need to modify anything then changes are made only in the subroutines after which all the changes will be reflected.

### **Standard header files**

**C provides us with some standard header files which are available easily.**

**Common standard header files are:**

- string.h – used for handling string functions.
- stdlib.h – used for some miscellaneous functions.
- stdio.h – used for giving standardized input and output.
- math.h – used for mathematical functions.
- alloc.h – used for dynamic memory allocation.
- conio.h – used for clearing the screen.

The header files are added at the start of the source code so that they can be used by more than one function of the same file.

### **Object files**

- They are the files that are generated by the compiler as the source code file is processed.
- These files generally contain the binary code of the function definitions.
- The object file is used by the linker for producing an executable file for combining the object files together. It has a '.o' extension.

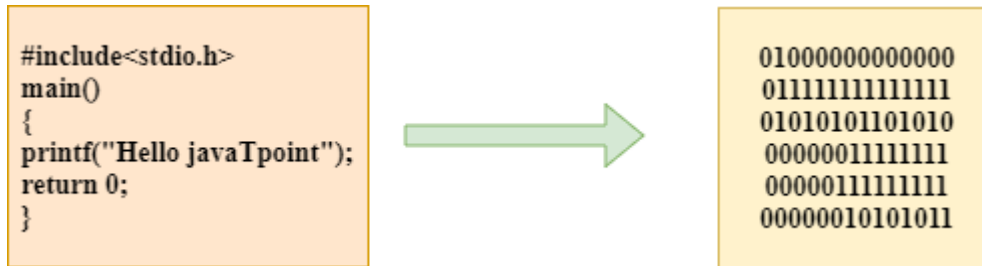
### **Executable file**

- This file is generated by the linker.
- Various object files are linked by the linker for producing a binary file which will be executed directly.
- They have an '.exe' extension.

### **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.

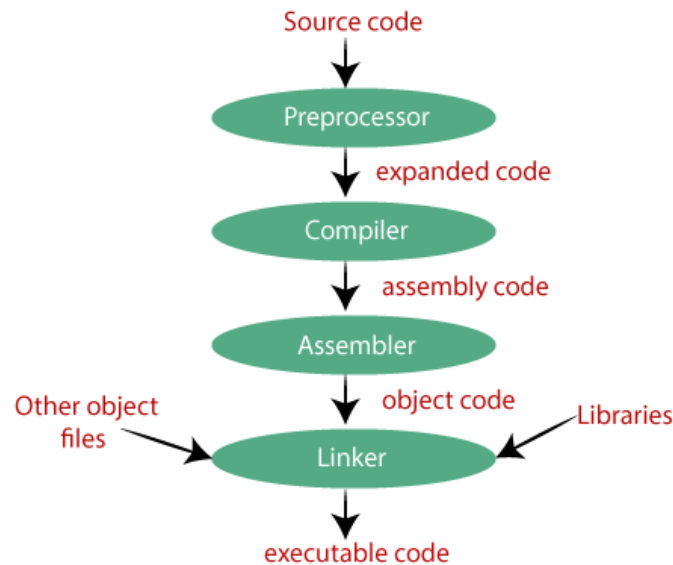


The c compilation process converts the source code taken as input into the object code or machine code. The compilation process can be divided into four steps, i.e., Pre-processing, Compiling, Assembling, and Linking.

The preprocessor takes the source code as an input, and it removes all the comments from the source code. The preprocessor takes the preprocessor directive and interprets it. For example, if `<stdio.h>`, the directive is available in the program, then the preprocessor interprets the directive and replace this directive with the content of the 'stdio.h' file.

The following are the phases through which our program passes before being transformed into an executable form:

- Preprocessor
- Compiler
- Assembler
- Linker



### **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.

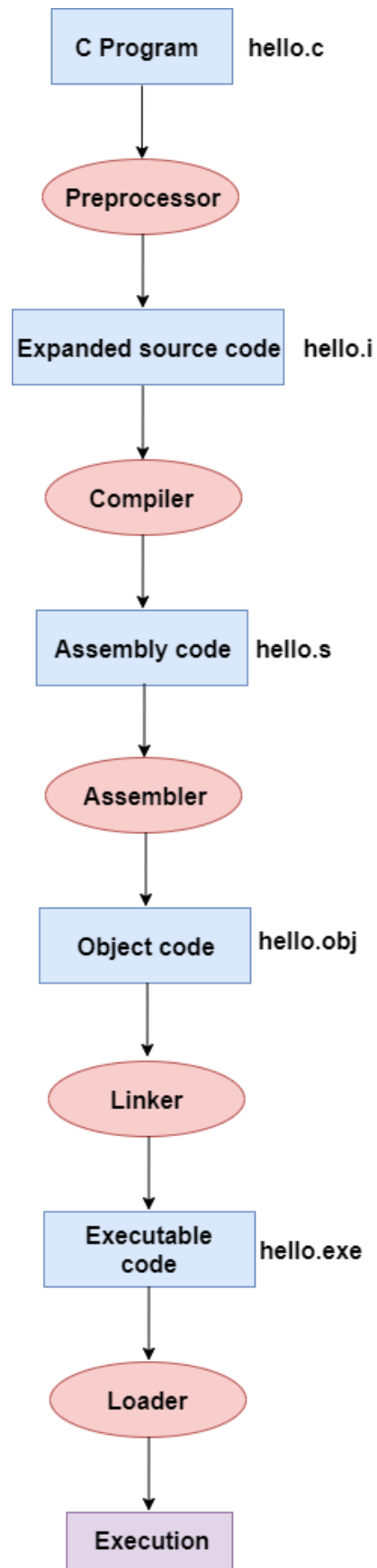
**Let's understand through an example.**

**hello.c**

```
#include <stdio.h>
int main()
{
    printf("Hello javaTpoint");
    return 0;
}
```

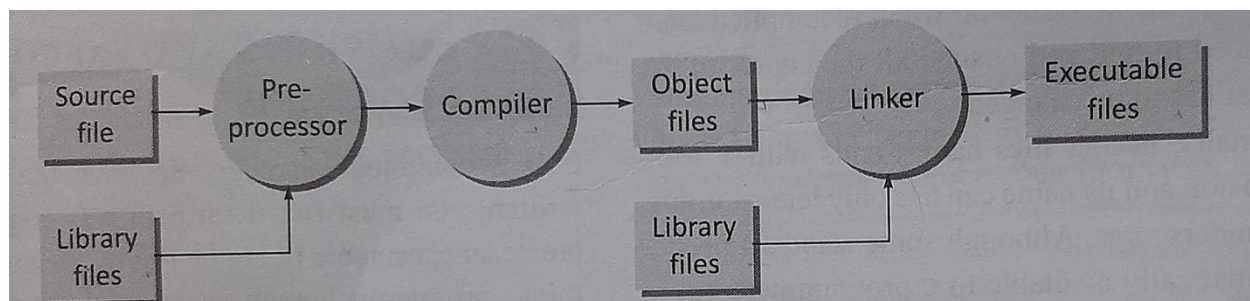
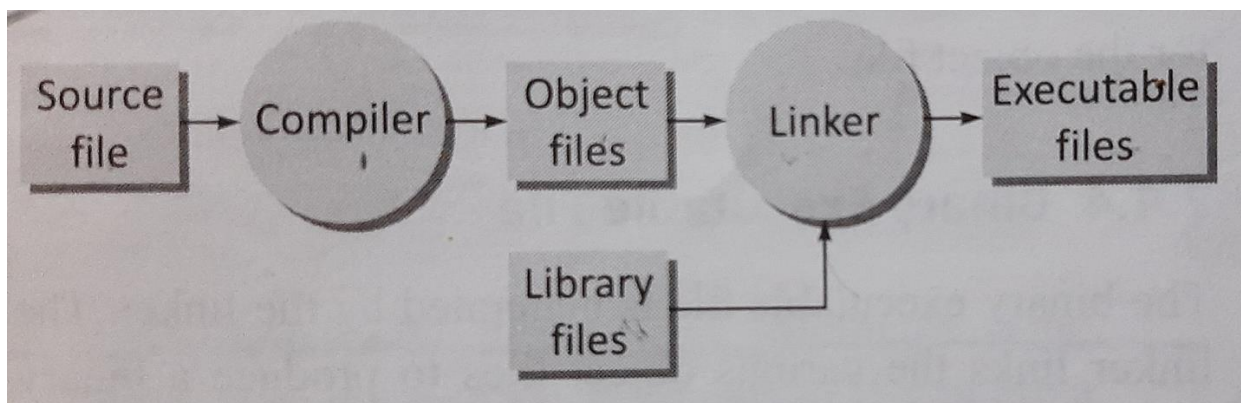
**Now, we will create a flow diagram of the above program:**

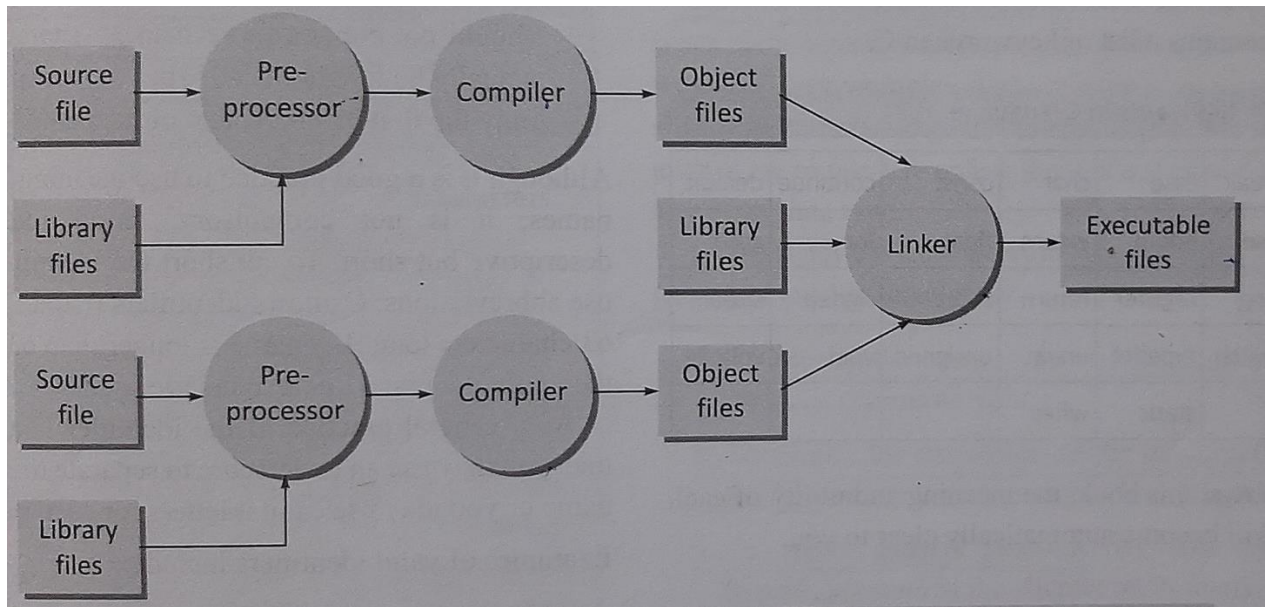




**In the above flow diagram, the following steps are taken to execute a program:**

- Firstly, the input file, i.e., hello.c, is passed to the preprocessor, and the preprocessor converts the source code into expanded source code. The extension of the expanded source code would be hello.i.
- The expanded source code is passed to the compiler, and the compiler converts this expanded source code into assembly code. The extension of the assembly code would be hello.s.
- This assembly code is then sent to the assembler, which converts the assembly code into object code.
- After the creation of an object code, the linker creates the executable file. The loader will then load the executable file for the execution.





## Escape Sequence in C

An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character.

It is composed of two or more characters starting with backslash \. For example: \n represents new line.

## List of Escape Sequences in C

Escape Sequence	Meaning
\a	Alarm or Beep
\b	Backspace
\f	Form Feed
\n	New Line

\r	Carriage Return
\t	Tab (Horizontal)
\v	Vertical Tab
\\	Backslash
\'	Single Quote
\"	Double Quote
\?	Question Mark
\nnn	octal number
\xhh	hexadecimal number
\0	Null

### Escape Sequence Example

```
#include<stdio.h>
int main(){
    int number=50;
    printf("You\nare\nlearning\n\'c\' language\n\"Do you know C language\");
    return 0;
}
```

Output:

```
You
are
learning
'c' language
"Do you know C language"
```

**2.**

```
// C program to illustrate \a escape sequence
#include <stdio.h>
int main(void)
{
    printf("My mobile number "
           "is 7\a8\a7\a3\a9\a2\a3\a4\a0\a8\a");
    return (0);
}
```

**3.**

```
// C program to illustrate \b escape sequence
#include <stdio.h>
int main(void)
{
    // \b - backspace character transfers
    // the cursor one character back with
    // or without deleting on different
    // compilers.
    printf("Hello ICS \b\b\b\b Nutan Nagar");
    return (0);
}
```

**4.**

```
// C program to illustrate \n escape sequence
#include <stdio.h>
int main(void)
{
    // Here we are using \n, which
    // is a new line character.
    printf("Hello\n");
    printf("ICS Nutan Nagar");
    return (0);
}
```

**5.**

```
// C program to illustrate \t escape sequence
#include <stdio.h>
int
main(void)
{
    // Here we are using \t, which is
```

```

        // a horizontal tab character.
        // It will provide a tab space
        // between two words.
        printf("Hello \t ICS");
        return (0);
    }
6.

```

```

// C program to illustrate \v escape sequence
#include <stdio.h>
int main(void)
{
    // Here we are using \v, which
    // is vertical tab character.
    printf("Hello friends");

    printf("\v Welcome to ICS");

    return (0);
}
7.

```

```

// C program to illustrate \r escape sequence
#include <stdio.h>
int main(void)
{
    // Here we are using \r, which
    // is carriage return character.
    printf("Hello fri \r ends");
    return (0);
}
8.

```

```

// C program to illustrate \\(Backslash) escape sequence to print backslash.
#include <stdio.h>
int main(void)
{
    // Here we are using \,
    // It contains two escape sequence
    // means \ and \n.
    printf("Hello\\ICS");
    return (0);
}

```

**9.**

// C program to illustrate \' escape sequence/ and \" escape sequence to print single quote and double //quote.

```
#include <stdio.h>
int main(void)
{
    printf("\' Hello ICS\n");
    printf("\" Hello ICS");
    return 0;
}
```

**10.**

// C program to illustrate \? escape sequence

```
#include <stdio.h>
int main(void)
{
    // Here we are using \?, which is
    // used for the presentation of trigraph
    // in the early of C programming. But
    // now we don't have any use of it.
    printf("\? \?! \n");
    return 0;
}
```

**11.**

// C program to illustrate \OOO escape sequence

```
#include <stdio.h>
int main(void)
{
    // we are using \OOO escape sequence, here
    // each O in "OOO" is one to three octal
    // digits(0....7).
    char* s = "A\0725";
    printf("%s", s);
    return 0;
}
```

**12.**

// C program to illustrate \XHH escape

// sequence

```
#include <stdio.h>
int main(void)
```

```

{
    // We are using \xhh escape sequence.
    // Here hh is one or more hexadecimal
    // digits(0....9, a...f, A...F).
    char* s = "B\x4a";
    printf("%s", s);
    return 0;
}

```

### 13.

```

#include <stdio.h>
int main()
{
    printf("vignesh\krishnakumar \n");
    printf("new line \n next line \n");
    printf("welcome \t\t concolidated\t \v example \n");
    printf("\v");
    printf("\learning is fun\n ");
    printf("\r");
    printf("\n\text surrounded with single quotation\' ");
    printf("\n\"double quotes surrounded text\" ");
    printf("\n whats your fathers name\t? ");
    printf("\n E:\\test\\test1\\test2 ");
    char* b = "B\124";
    printf("\n%s", b);
    char* s = "B\x5b";
    printf("\n %s", s);
    return 0;
}

```

### Character set of C

**Character** : It denotes any alphabet, digit or special symbol used to represent information.

**Use** : These characters can be combined to form variables. C uses constants, variables, operators, keywords and expressions as building blocks to form a basic C program.



**Character set** : The character set is the fundamental raw material of any language and they are used to represent information. Like natural languages, computer language will also have well defined character set, which is useful to build the programs.

The characters in C are grouped into the following two categories:

**1. Source character set**

- a. Alphabets
- b. Digits
- c. Special Characters
- d. White Spaces

**2. Execution character set**

- a. Escape Sequence

**Source character set**

**ALPHABETS**

Uppercase letters	A-Z
Lowercase letters	a-z

**DIGITS** 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

**SPECIAL CHARACTERS**

~	tilde	%	percent sign
	vertical bar	@	at symbol
+	plus sign	<	less than
_	underscore	-	minus sign
>	greater than	^	caret
#	number sign	=	equal to
&	ampersand	\$	dollar sign
/	slash	(	left parenthesis
*	asterisk	\	back slash
)	right parenthesis	'	apostrophe

:	colon	:	colon	[	left bracket
"	quotation mark	"	quotation mark	;	semicolon
]	right bracket	]	right bracket	!	exclamation mark
,	comma	,	comma	{	left flower brace
?	Question mark	?	Question mark	.	dot operator
}	right flower brace	}	right flower brace		

## WHITESPACE CHARACTERS

\b	blank space	\t	horizontal tab
\v	vertical tab	\r	carriage return
\f	form feed	\n	new line
\\	Back slash	\'	Single quote
\"	Double quote	\?	Question mark
\0	Null	\a	Alarm (bell)

## Execution Character Set

Certain ASCII characters are unprintable, which means they are not displayed on the screen or printer. Those characters perform other functions aside from displaying text. Examples are backspacing, moving to a newline, or ringing a bell.

They are used in output statements. Escape sequence usually consists of a backslash and a letter or a combination of digits. An escape sequence is considered as a single character but a valid character constant.

These are employed at the time of execution of the program. Execution characters set are always represented by a backslash (\) followed by a character. Note that each one of character constants represents one character, although they consist of two characters. These characters combinations are called as escape sequence.

### Backslash character constants

Character	ASCII value	Escape Sequence	Result
-----------	-------------	-----------------	--------

Null	000	\0	Null
Alarm (bell)	007	\a	Beep Sound
Back space	008	\b	Moves previous position
Horizontal tab	009	\t	Moves next horizontal tab
New line	010	\n	Moves next Line
Vertical tab	011	\v	Moves next vertical tab
Form feed page	012	\f	Moves initial position of next
Carriage return	013	\r	Moves beginning of the line
Double quote	034	\"	Present Double quotes
Single quote	039	\'	Present Apostrophe
Question mark	063	\?	Present Question Mark
Back slash	092	\\	Present back slash
Octal number	\000		
Hexadecimal number	\x		