

# Lecture- 1

## Introduction

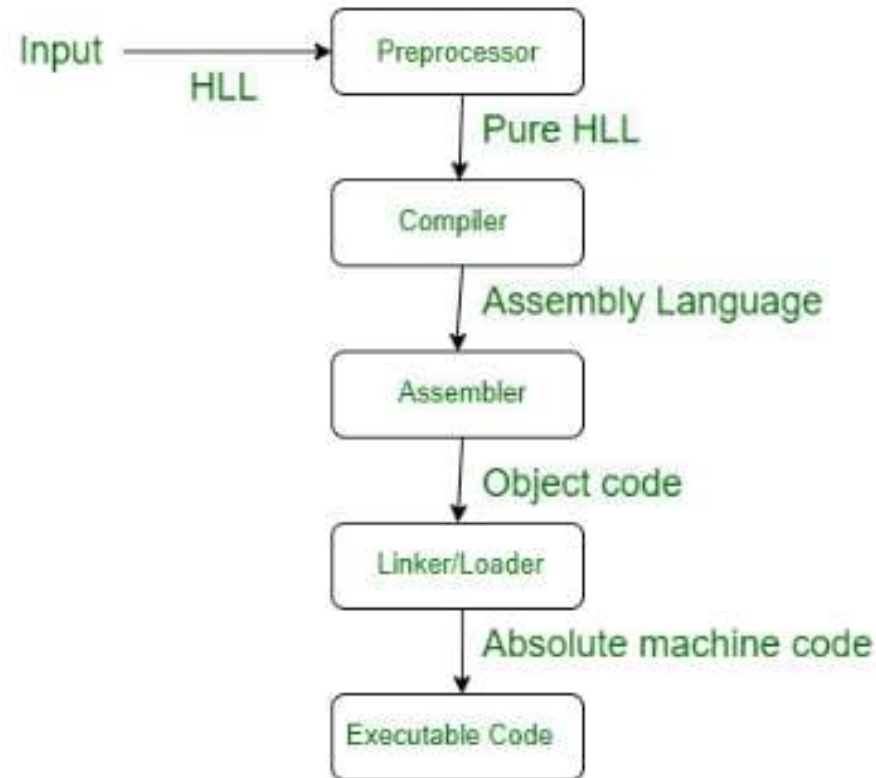
# Overview

- Introduction to Programming Language
- Translator Program
  - Assembler, compiler, interpreter
- Algorithm
- Flowchart
- Psudocode
- Introduction to C Programming Language
- Features of C Programming Languages
- IDE setup-> Codeblocks/ Microsoft Visual Studio

# Programming Language

- A programming language is a formal system of communication used to write instructions that a program and computer can understand and execute.
  - These languages are used by developers to create software programs, applications, and systems.
- High Level Language: A programming language that is closer to human languages and abstracts the details of the computer's hardware.
- Low Level Language: A programming language that is closer to machine code, providing little abstraction from the hardware.
- Assembly language: A low-level programming language that uses symbolic code to represent machine-level instructions.

# Language Preprocessing System

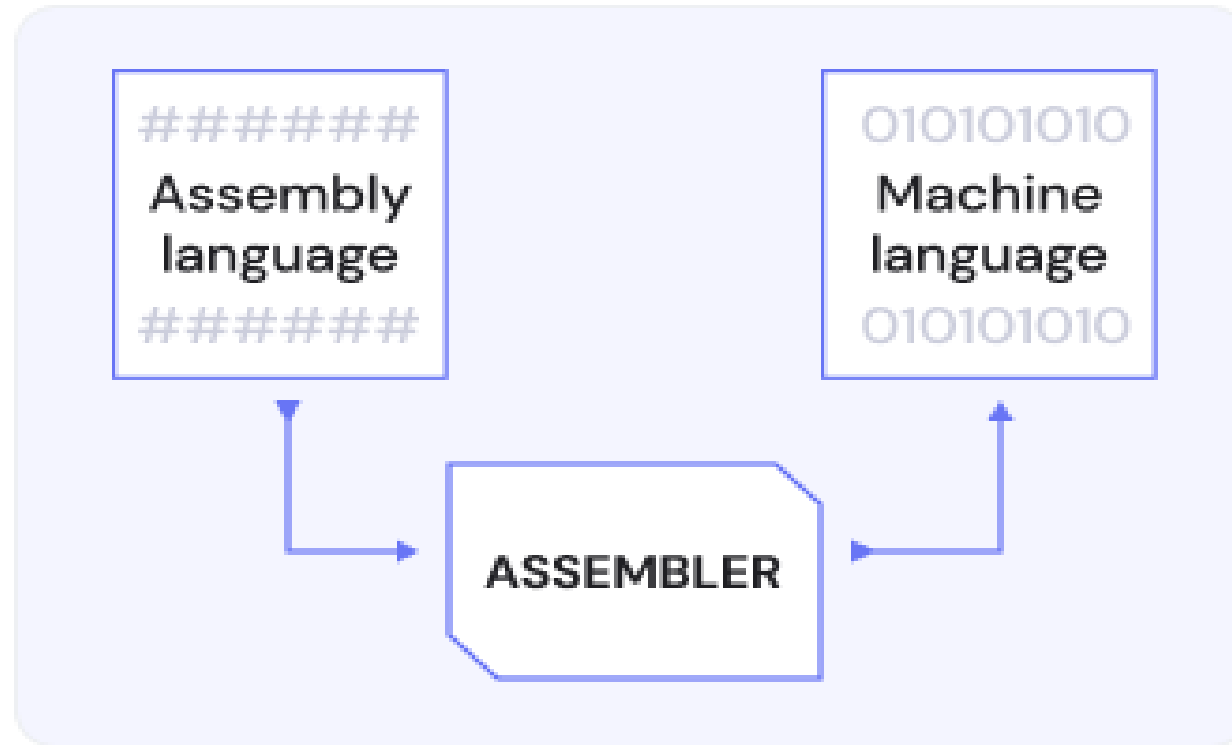


*Language Processing System*

# Translator

- Translator converts high level language to low level language
- Why high level language to low level language conversion is needed?
  - Because computer executes all the instructions in binary form. High level language is human understandable but not machine understandable.
- Then why don't we write codes directly in low level language?
  - Because low-level language is difficult for humans to understand and use all the time.
- Types of Translators
  - Assembler
  - Compiler
  - Interpreter

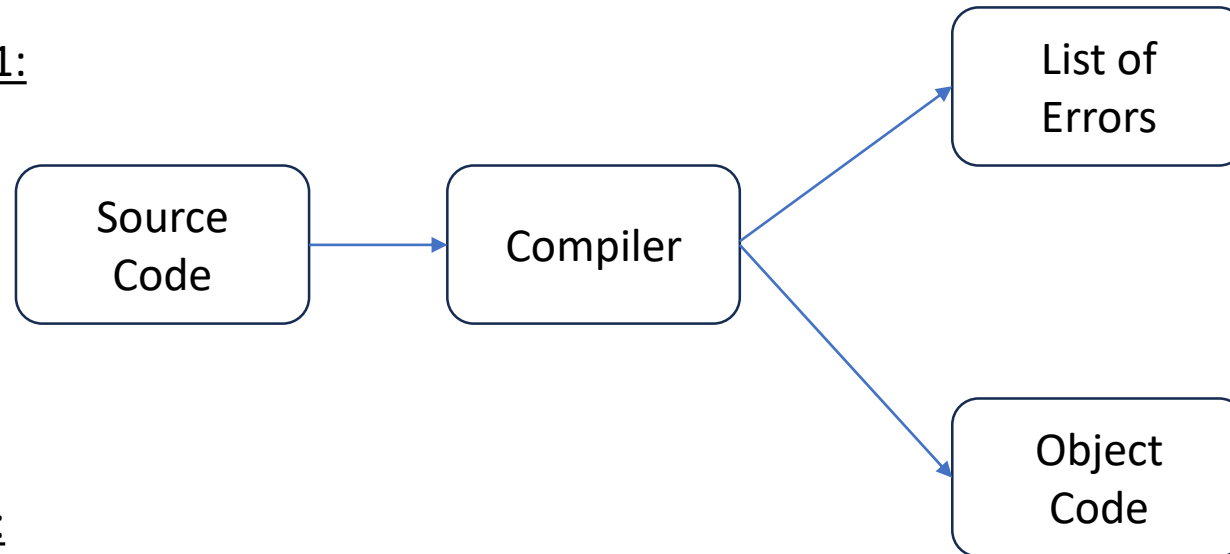
# Assembler



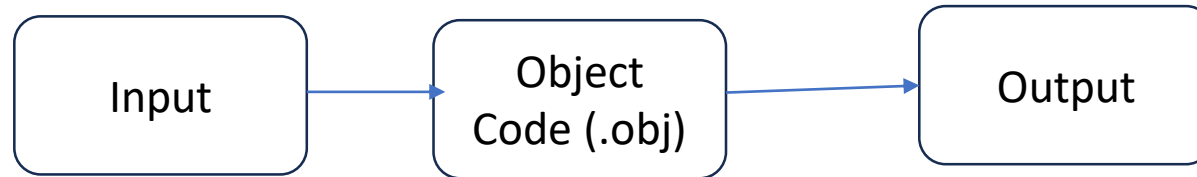
# Main 2 Steps in the Compiler's Function Procedure

In compiler, the full program is being translated at a time and then being executed

Step 1:

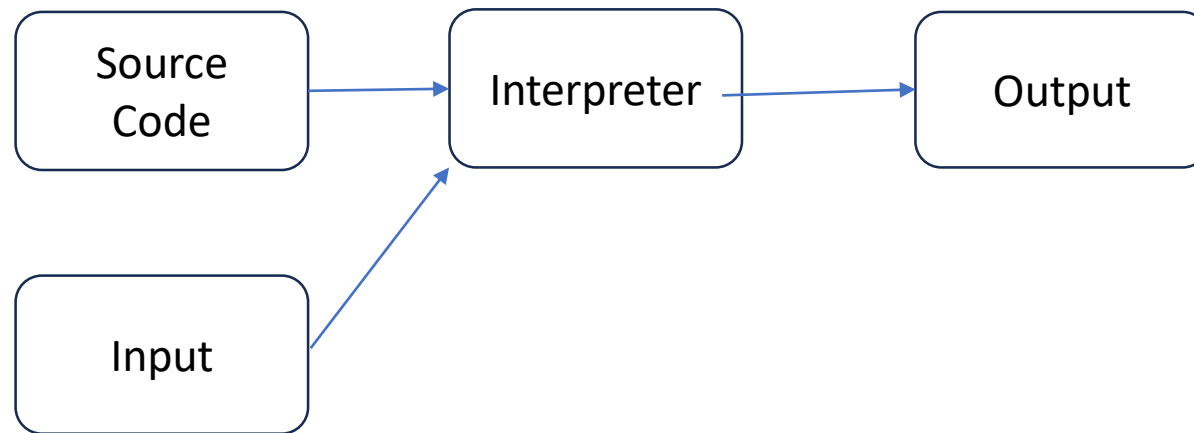


Step 2:



# Interpreter's Function Procedure

In interpreter, the full program is being translated and executed in a single step



Interpreter translates Line by Line and then executes



# Difference Between Compiler and Interpreter

Compiler	Interpreter
<ul style="list-style-type: none"><li>• A compiler takes the entire program in one go.</li></ul>	<ul style="list-style-type: none"><li>• An interpreter takes a single line of code at a time.</li></ul>
<ul style="list-style-type: none"><li>• The compiler generates an intermediate machine code.</li></ul>	<ul style="list-style-type: none"><li>• The interpreter never produces any intermediate machine code.</li></ul>
<ul style="list-style-type: none"><li>• The compiler is best suited for the production environment.</li></ul>	<ul style="list-style-type: none"><li>• An interpreter is best suited for a software development environment.</li></ul>
<ul style="list-style-type: none"><li>• The compiler is used by programming languages such as C, C ++, C #, Scala, Java, etc.</li></ul>	<ul style="list-style-type: none"><li>• An interpreter is used by programming languages such as Python, PHP, Perl, Ruby, etc.</li></ul>

# Algorithm

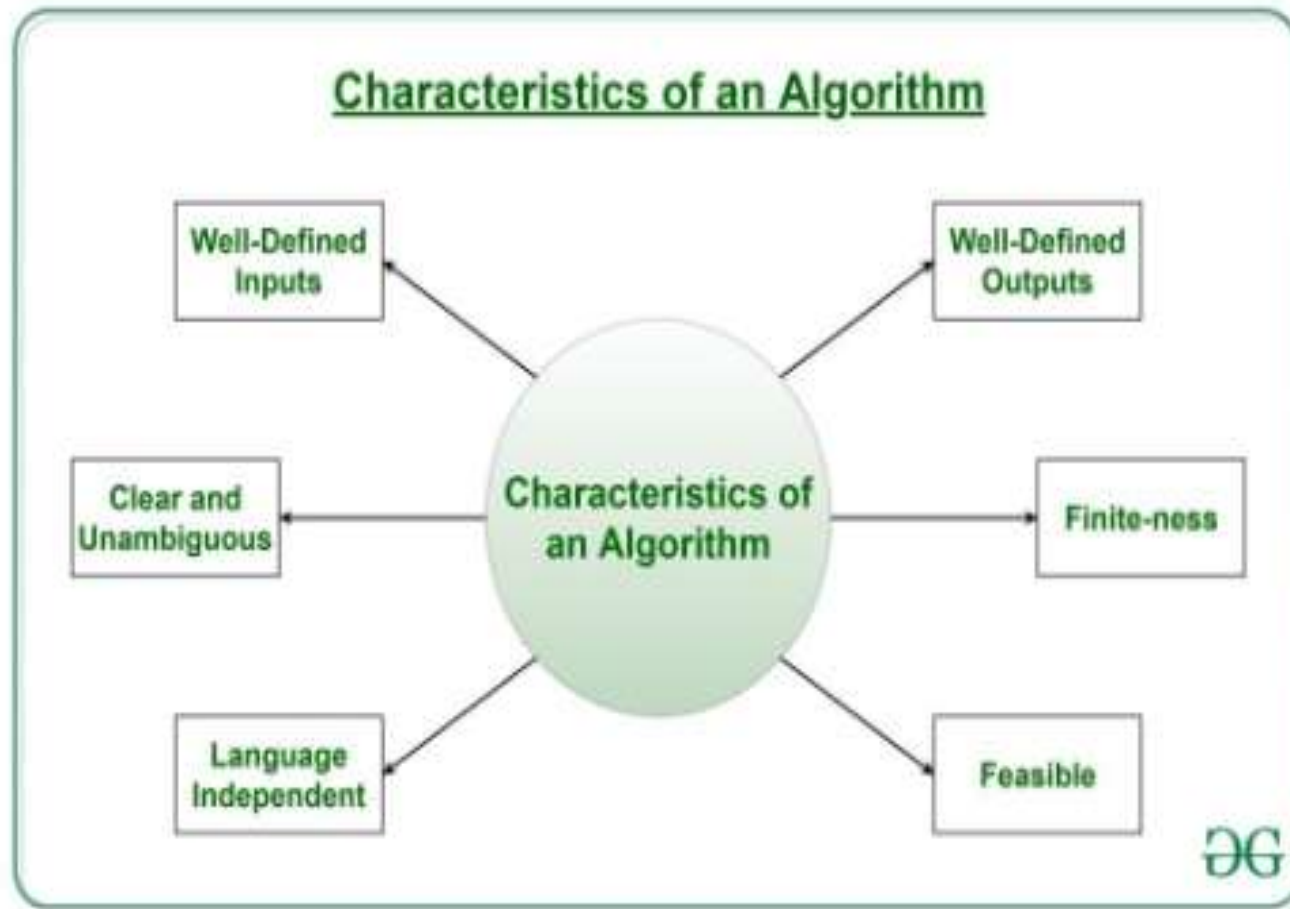
- **What is Algorithm?**

- A set of finite rules or instructions to be followed in calculations or other problem-solving operations

- **What is the need for algorithms?**

- Algorithms are necessary for solving complex problems efficiently and effectively.
- They help to automate processes and make them more reliable, faster, and easier to perform.
- Algorithms also enable computers to perform tasks that would be difficult or impossible for humans to do manually.
- They are used in various fields such as mathematics, computer science, engineering, finance, and many others to optimize processes, analyze data, make predictions, and provide solutions to problems.

# Characteristics of an Algorithm



# Example of Algorithm (Example 1)

- Algorithm: Add Two Integers
  - Input: Two integers, a and b.
  - Output: The sum of a and b.
- Algorithm
  - Start
  - Input the first integer a.
  - Input the second integer b.
  - Add the two integers:  $\text{sum} = a + b$ .
  - Output the result sum.
  - End






# Example of Algorithm (Example 2)

- Algorithm: Calculate and Print the Area of a Rectangle
  - Input: Length  $l$  and width  $w$  of the rectangle.
  - Output: Area of the rectangle.
- Algorithm
  - Step 1: Start
  - Step 2: Input the length  $l$  of the rectangle.
  - Step 3: Input the width  $w$  of the rectangle.
  - Step 4: Calculate the area of the rectangle using the formula:  $\text{Area} = l \times w$
  - Step 5: Output the calculated area.
  - Step 6: End

# Flowchart

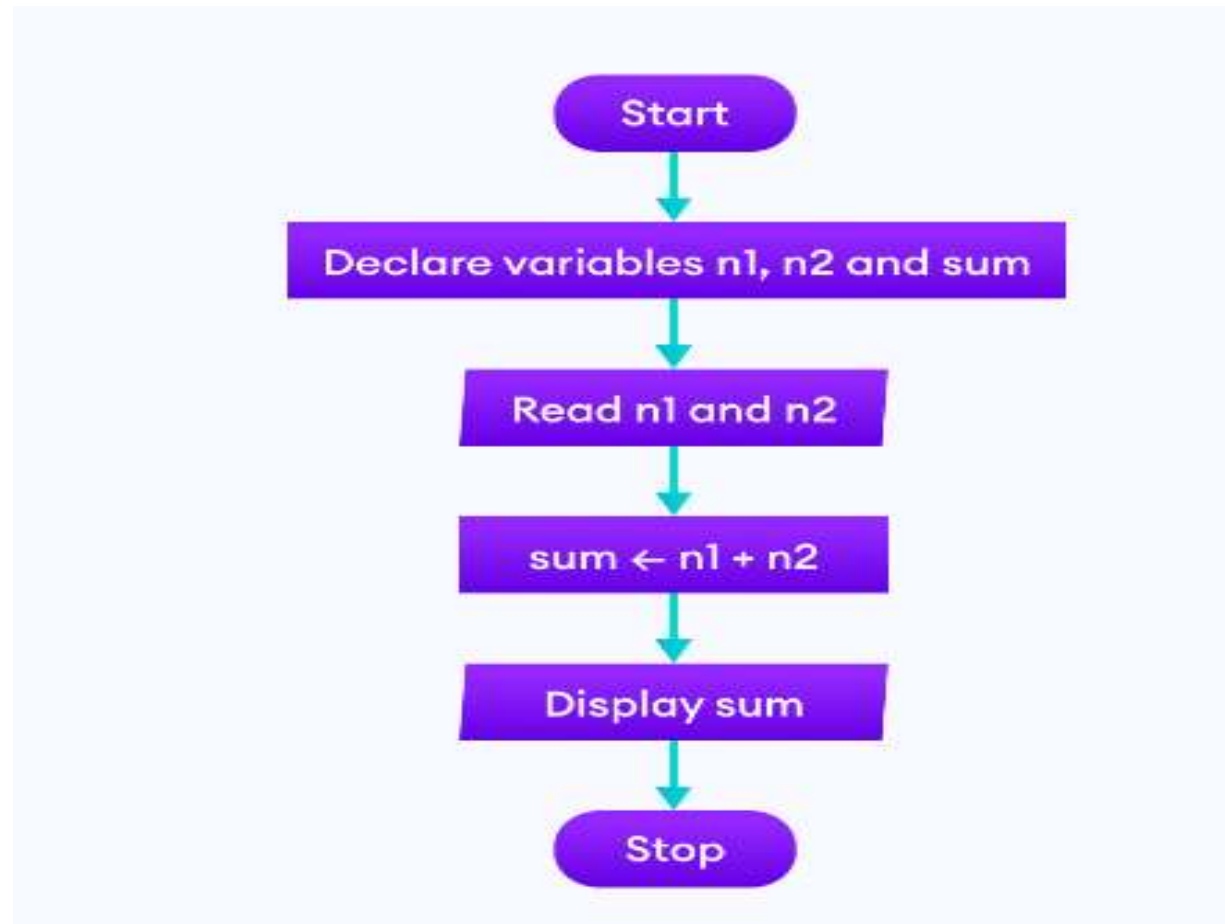
- A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

# Flowchart Basic Symbols

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

# Example of Flowchart (Example 1)

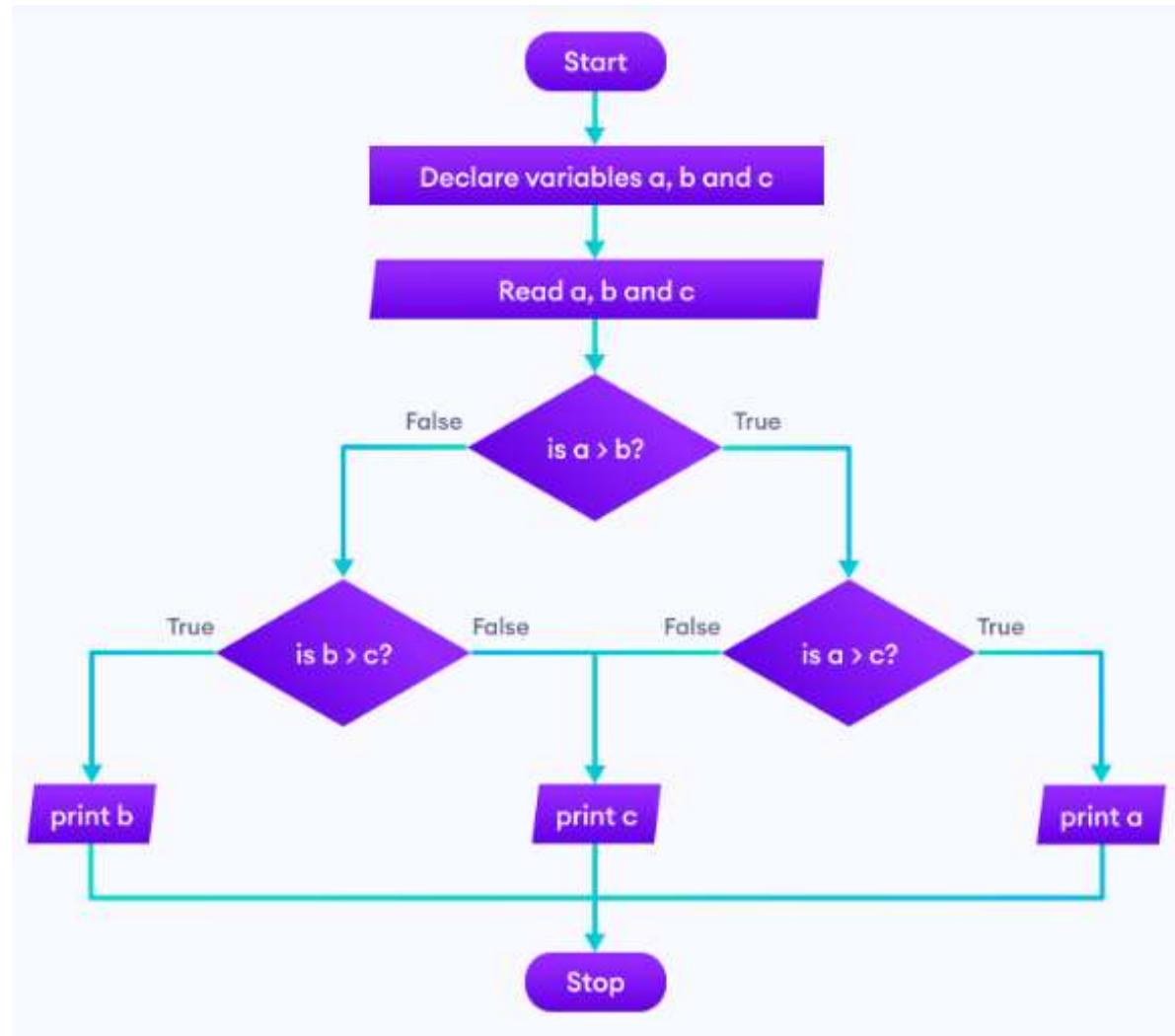
- Add two numbers entered by the user.





## Example of Flowchart (Example 2)

Find the largest among three different numbers entered by the user.



# Introduction to C Programming 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 a widely used professional language for various reasons –
  - Easy to learn
  - Structured language
  - It produces efficient programs
  - It can handle low-level activities
  - It can be compiled on a variety of computer platforms

# Features of C Programming Language

- I. C is a Procedural and Structured Language
- II. C is a General-Purpose Language
- III. C is a Fast Programming Language
- IV. C is Portable
- V. C is Extensible
- VI. Standard Libraries in C
- VII. Pointers in C
- VIII. C is a Mid-Level Programming Language
- IX. C Has a Rich Set of Built-in Operators
- X. Recursion in C
- XI. User-defined Data Types in C
- XII. Preprocessor Directives in C
- XIII. File Handling in C

# Features of C Programming Language (Explanation (1/4))

- C is a Procedural and Structured Language
  - C is described as procedure-oriented and structured programming language. It is procedural because a C program is a series of instructions that explain the procedure of solving a given problem. It makes the development process easier. In C, the logic of a process can be expressed in a structured or modular form with the use of function calls. C is generally used as an introductory language to introduce programming to school students because of this feature.
- C is a General-Purpose Language
  - The C language hasn't been developed with a specific area of application as a target. From system programming to photo editing software, the C programming language is used in various applications. Some of the common applications of C programming include the development of Operating Systems, databases, device drivers, etc.
- C is a Fast Programming Language
  - C is a compiler-based language which makes the compilation and execution of codes faster. The source code is translated into a hardware-specific machine code, which is easier for the CPU to execute, without any virtual machine, as some of the other languages like Java need. The fact that C is a statically typed language also makes it faster compared to dynamically typed languages. Being a compiler-based language, it is faster as compared to interpreter-based languages.

# Features of C Programming Language (Explanation (2/4))

- C is Portable
  - Another feature of the C language is its portability. C programs are machine-independent which means that you can compile and run the same code on various machines with none or some machine-specific changes. C programming provides the functionality of using a single code on multiple systems depending on the requirement.
- C is Extensible
  - C is an extensible language. It means if a code is already written, you can add new features to it with a few alterations. Basically, it allows adding new features, functionalities, and operations to an existing C program.
- Standard Libraries in C
  - Most of the C compilers are bundled with an extensive set of libraries with several built-in functions. It includes OS-specific utilities, string manipulation, mathematical functions, etc. Importantly, you can also create your user-defined functions and add them to the existing C libraries. The availability of such a vast scope of functions and operations allows a programmer to build a vast array of programs and applications using the C language.

# Features of C Programming Language (Explanation (3/4))

- **Pointers in C**
  - One of the unique features of C is its ability to manipulate the internal memory of the computer. With the use of pointers in C, you can directly interact with the memory. Pointers point to a specific location in the memory and interact directly with it. Using the C pointers, you can interact with external hardware devices, interrupts, etc.
- **C is a Mid-Level Programming Language**
  - High-level languages have features such as the use of mnemonic keywords, user-defined identifiers, modularity etc. C programming language, on the other hand, provides a low-level access to the memory. This makes it a mid-level language. As a mid-level programming language, it provides the best of both worlds. For instance, C allows direct manipulation of hardware, which high-level programming languages do not offer.
- **C Has a Rich Set of Built-in Operators**
  - C is perhaps the language with the most number of built-in operators which are used in writing complex or simplified C programs. In addition to the traditional arithmetic and comparison operators, its binary and pointer related operators are important when bit-level manipulations are required.

# Features of C Programming Language (Explanation (4/4))

- Recursion in C
  - C language provides the feature of recursion. Recursion means that a function can be created that can call itself multiple times until a given condition is true, just like the loops. Recursion in C programming provides the functionality of code reusability and backtracking.
- User-defined Data Types in C
  - C has three basic data types in int, float and char. However, C programming has the provision to define a data type of any combination of these three types, which makes it very powerful. In C, structures and union types can be also defined. Feature for declaring enumerated data types are also available.
- Preprocessor Directives in C
  - In C, we have preprocessor directives such as #include, #define, etc. They are not the language keywords. Preprocessor directives in C carry out some of the important roles such as importing functions from a library, defining and expanding the macros, etc.
- File Handling in C
  - C language doesn't directly manipulate files or streams. Handling file IO is not a part of the C language itself but instead is handled by libraries and their associated header files. File handling is generally implemented through high-level I/O which works through streams. C identifies stdin, stdout and stderr as standard input, output and error streams. These streams can be directed to a disk file to perform read/write operations.

# C Environment Setup

- Download and install one of the following IDEs (Latest Version)
  - Codeblocks
  - Microsoft Visual Studio



# C Program Example (Printing Hello World)

- Print Hello World

```
#include <stdio.h>

int main() {

    /* my first program in C */
    printf("Hello, World! \n");

    return 0;
}
```

# Steps involving in Execution of a C Program

- The first statement in the code (printing Hello world in the previous page) is the `#include` statement that imports the `stdio.h` file in the current C program. This is called a preprocessor directive. This header file contains the definitions of several library functions used for stand IO operations. Since we shall be calling the `printf()` function which is defined in the `stdio.h` library.
- Every C program must contain a `main()` function. The `main()` function in this program prints the "Hello World" message.
- In C, every function needs to have a return value. If the function doesn't return anything, its value is void. In the example above, the `main()` function has `int` as its return value. Since the `main()` function doesn't need to return anything, it is defined to return an integer "0". The "return 0" statement also indicates that the program has been successfully compiled and run.

# Introduction with scanf() Function

```
#include <stdio.h>

int main(){

    int price, qty, ttl;

    printf("Enter price and quantity: ");
    scanf("%d %d", &price, &qty);

    ttl = price * qty;

    printf("Total : %d", ttl);

    return 0;
}
```

- The standard library that is bundled with the C compiler includes stdio.h header file, whose library function scanf() is most commonly used to accept user input from the standard input stream. In addition, the stdio.h library also provides other functions for accepting input.
- When the given program is run, C waits for the user to enter the values –
  - Enter price and quantity:
- When the values are entered and you press Enter, the program proceeds to the subsequent steps.
  - Enter price and quantity: 100 5
  - Total : 500

# Pseudocode

- Pseudocode is a simplified, human-readable way of describing the steps of an algorithm without using the strict syntax of a programming language.
- It's used to outline logic clearly and concisely, making it easier to understand and translate into actual code.

# Example of a Pseudocode

- A simple pseudocode for subtracting two numbers and printing the result:

```
BEGIN
    // Input: Two numbers
    PRINT "Enter the first number: "
    READ firstNumber

    PRINT "Enter the second number: "
    READ secondNumber

    // Compute the subtraction
    result = firstNumber - secondNumber

    // Output the result
    PRINT "The result of the subtraction is: " + result
END
```

# Example of a Complete C Code

- Complete C code for subtracting two numbers and printing the result:

```
#include <stdio.h>

int main()
{
    float num1, num2, result;

    printf("Enter the first number: ");
    scanf("%f", &num1);

    printf("Enter the second number: ");
    scanf("%f", &num2);

    result = num1 - num2;

    printf("The result of the subtraction is: %.2f\n", result);

    return 0;
}
```

# Comments in C

- Using comments in a C program increases the readability of the code. You must intersperse the code with comments at appropriate places. As far as the compiler is concerned, the comments are ignored. In C, the comments are one or more lines of text, that the compiler skips while building the machine code.
  - A blocked comment or multi-line comment must be put inside `/*` and `*/` symbols,
  - a single-line comment starts with the `//` symbol and is effective till the end of the line.

# Reference

- [Geeksforgeeks](#)
- [GrayCode](#)
- [Programiz.com](#)
- [TutorialsPoint](#)



Thanks