

**GLS UNIVERSITY**

**Faculty of Computer Applications &  
Information Technology**

**Integrated Msc(IT)**

**Semester-I**

**221601102 PROBLEM SOLVING THROUGH  
PROGRAMMING  
(Core Course)**

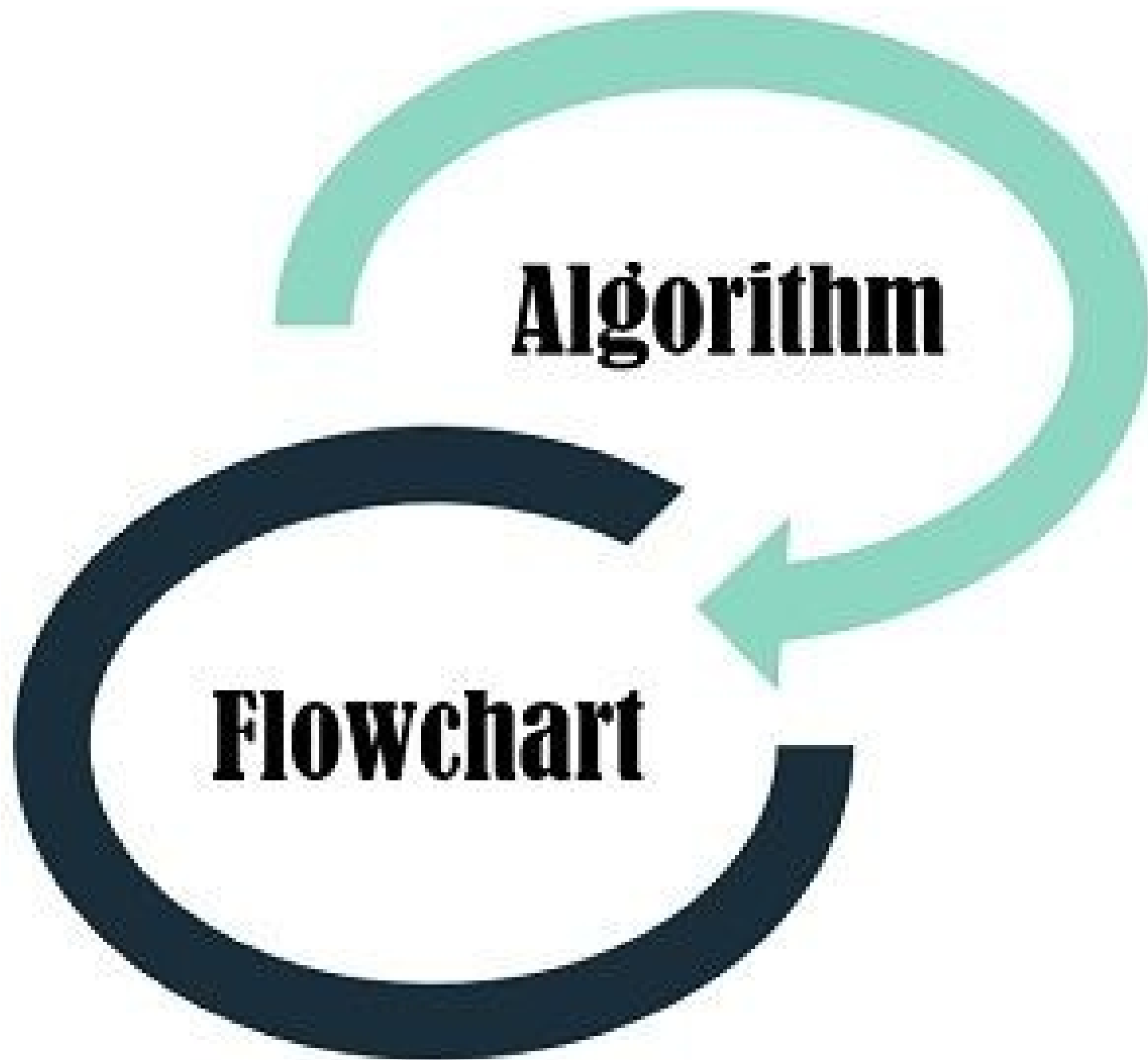
## Introduction to Algorithms and Flowcharts & C Language & Programming

- ☐ Introduction to Algorithms and Flowcharts
- ☐ Basic Concept of Logic
- ☐ Features of Algorithm & Flowcharts
- ☐ Criteria to be followed by an Algorithm & Flowcharts
- ☐ Symbols of Flowcharts
- ☐ Simple flowcharts & Algorithm with examples
- ☐ Steps to convert algorithm to program
- ☐ Introduction to C language
- ☐ History of C language
- ☐ Generation of Language
  
- ☐ Types of Languages
  - o Machine Language
  - o Assembly Language
  - o Higher Level Language
- ☐ Compiler, Assembler, Interpreter
- ☐ Introduction to ANSI Standards
- ☐ Introduction to Procedural Programming language
- ☐ Advantages and Disadvantages
- ☐ Introduction to Programming
- ☐ What is a Program?
- ☐ Structure of Program
- ☐ Compiling a Program
- ☐ Link and Run the Program

# Basic Concept of Logic

## Steps to solve problem:

- Define the Problem
- Identify the input, output & constraints
- Find various alternative solutions
- Select the best possible alternative
- Prepare detailed stepwise result for the identified alternative
- Compute the required result using the identified set of instructions
- Check the correctness of the answer obtained



# What is flow chart?

A flow chart shows the break down of a task into separate steps

They can be used to represent how programs work





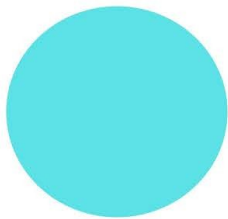
**Start / End**



**Input / Output**



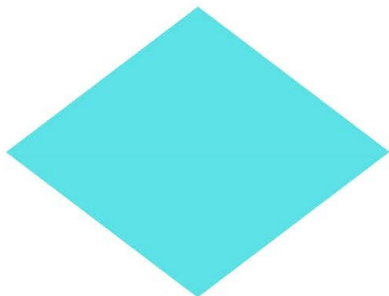
**Process**







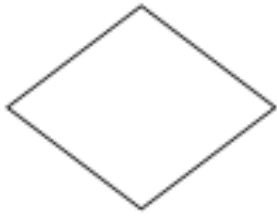

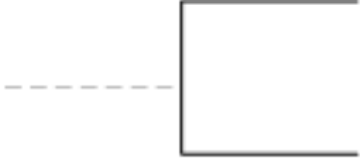
**Connector**

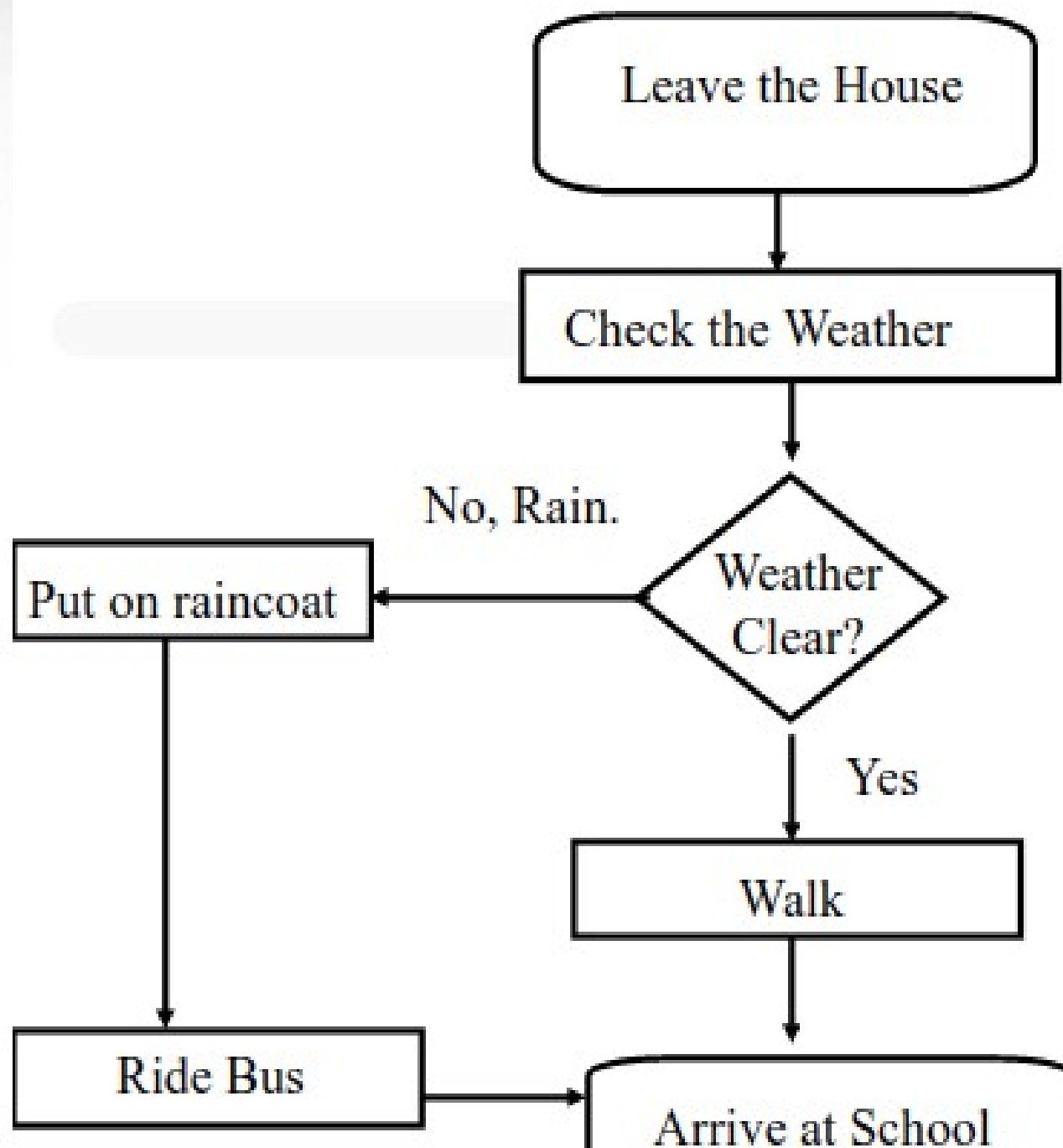


**Flow Direction**

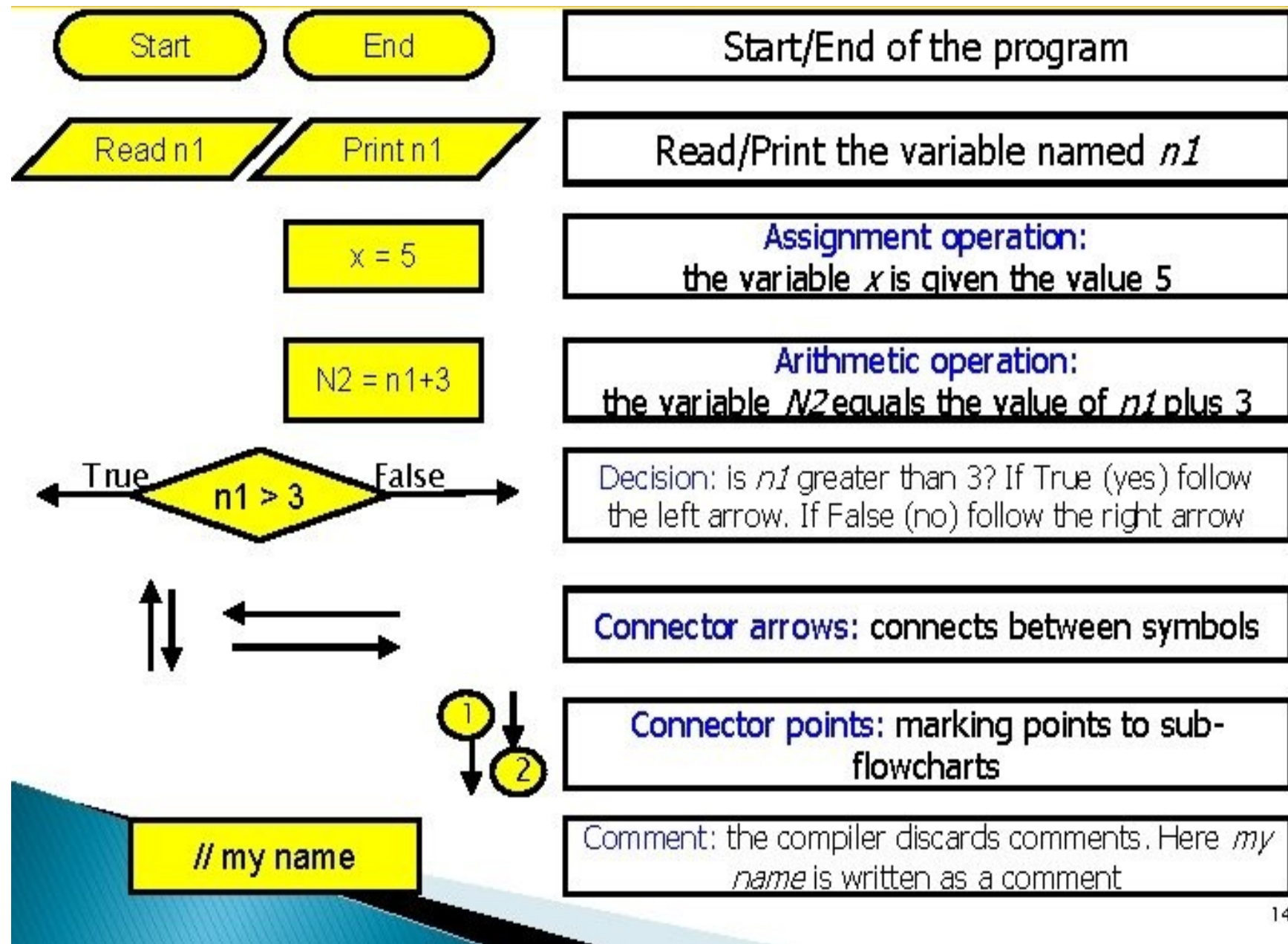


**Decision**

Flowchart Symbol	Symbol Name	Description
	Terminal (Start or Stop)	Terminals (Oval shapes) are used to represent start and stop of the flowchart.
	Flow Lines or Arrow	Flow lines are used to connect symbols used in flowchart and indicate direction of flow.
	Input / Output	Parallelograms are used to read input data and output or display information
	Process	Rectangles are generally used to represent process. For example, Arithmetic operations, Data movement etc.
	Decision	Diamond shapes are generally used to check any condition or take decision for which there are two answers, they are, yes (true) or no (false).
	Connector	It is used connect or join flow lines.
	Annotation	It is used to provide additional information about another flowchart symbol in the form of comments or remarks.





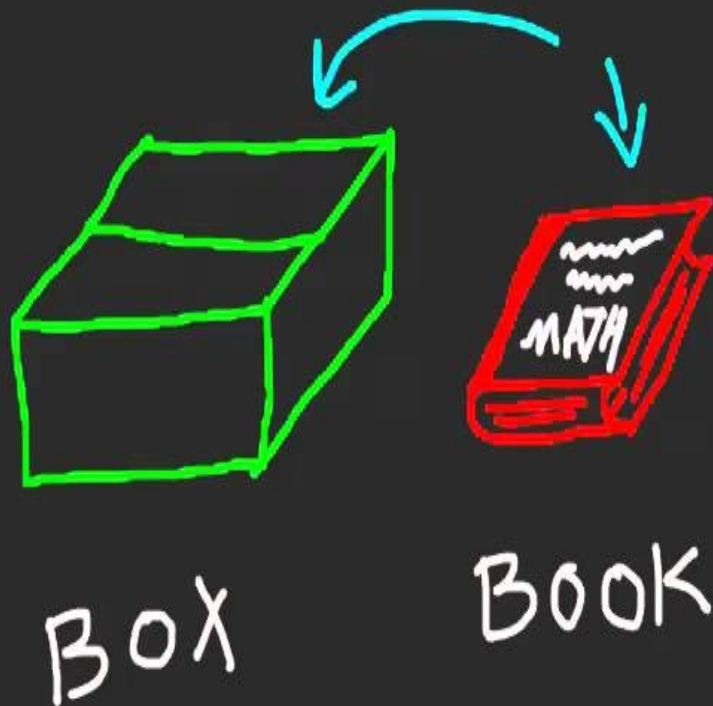


# Introduction to Algorithms

---



# What is an Algorithm?



Put the book in the box

- 1) Open the box
- 2) Pick up the book
- 3) Put it

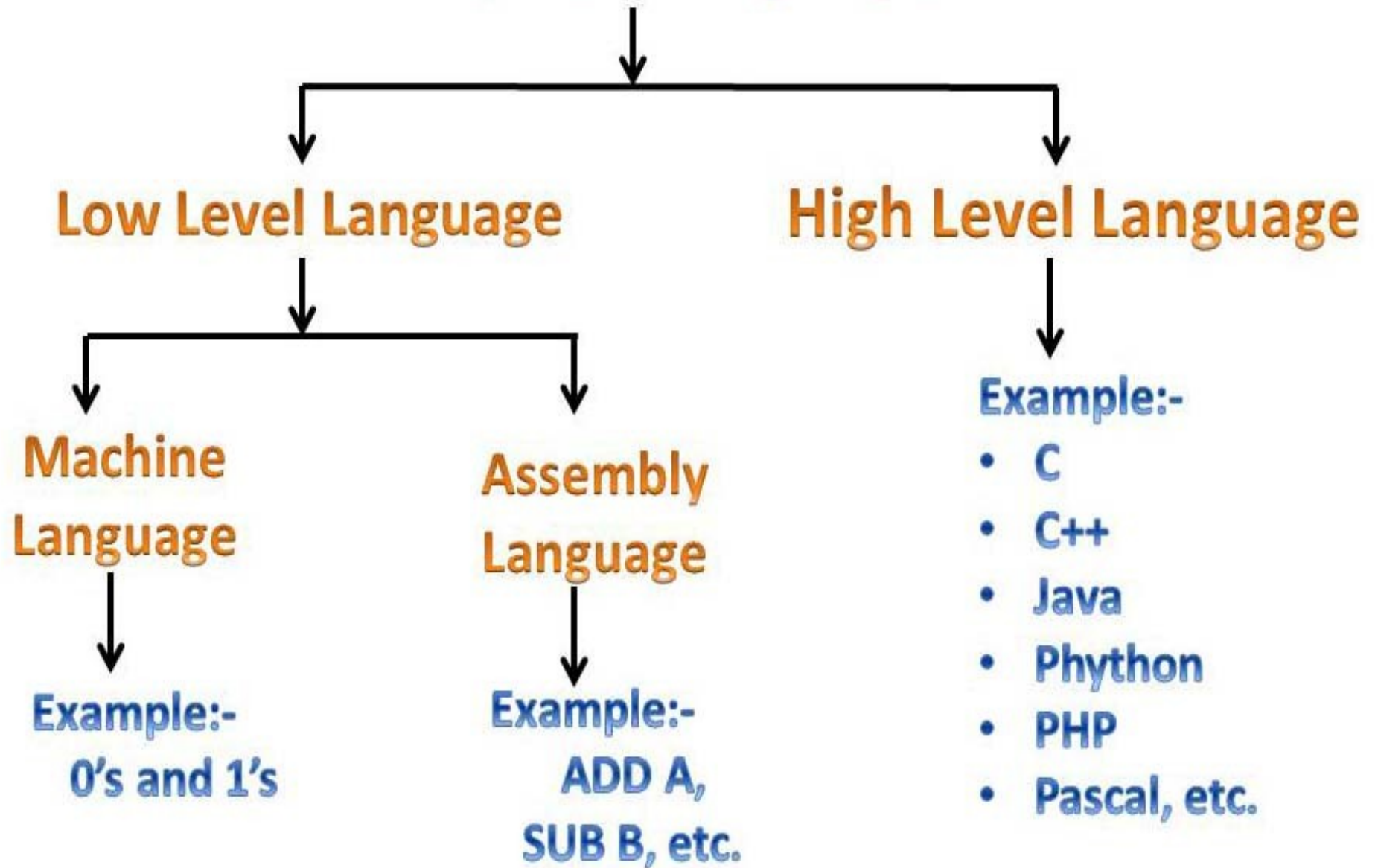
# Algorithm - Definition

- An algorithm is a set of instructions for solving a problem.
- When the instructions are followed, it must eventually stop with an answer.

**Write an algorithm to add two numbers entered by user.**

```
Step 1: Start
Step 2: Declare variables num1,
Step 3: Read values num1 and num2
Step 4: Add num1 and num2 and
        sum ← num1 + num2
Step 5: Display sum
Step 6: Stop
```

# Computer Languages





# ***Machine Code***

10011101000110100000  
01100011010001110110  
10000010111101101110  
11110110001011011000  
10000010011100011011  
10010011000111000000

# Machine Language

- It is also referred to as machine code or object code.
- Its the lowest level programming language
- It is a collection of binary digits or bits that the computer reads and interprets.
- Machine language is the only language a computer is capable of understanding.
- Consists entirely of numbers so almost impossible for the humans to understand



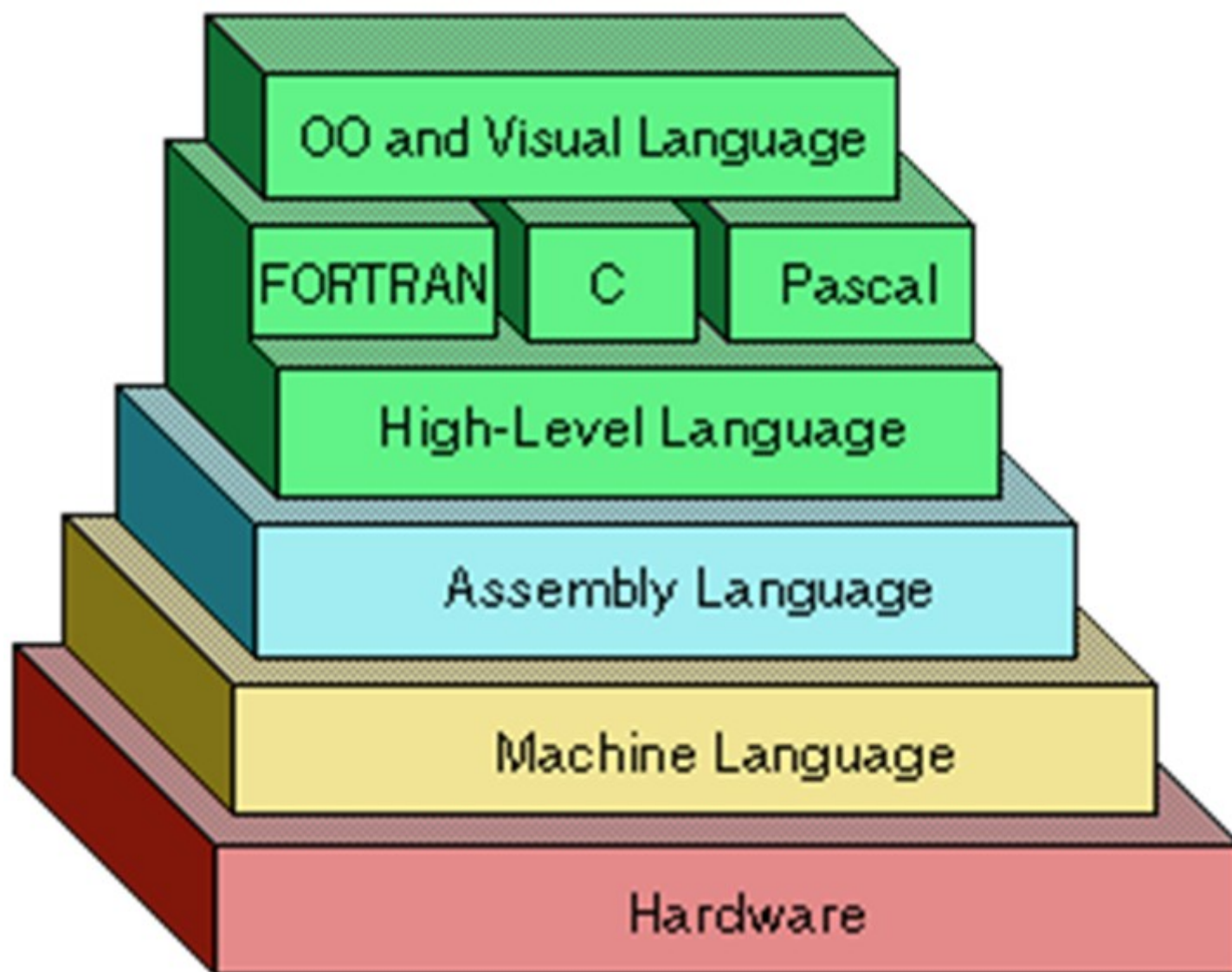
# Assembly

# Language

- Assembly languages have the same structure and set of commands as machine languages, but they enable a programmer to use names instead of numbers.
- Each type of CPU has its own machine language and assembly language, so an assembly language program written for one type of CPU won't run on another.
- Assembly language is the most basic programming language available for any processor.
- language is still useful for programmers when speed is necessary or when they need to carry out an operation that is not possible in high-level languages.

# High Level Language

- A high-level language is a programming language
- EG. C, FORTRAN, or Pascal
- They enable a programmer to write programs that are more or less independent of a particular type of computer.
- Such languages are considered high-level because they are closer to human languages.
- They are easier to read, write, and maintain.
- The first high-level programming languages were designed in the 1950s
- High-level language doesn't require addressing hardware constraints to a greater extent when developing a program



High-level  
Language

$(a+b)/2$

Source  
code



Translator  
Software

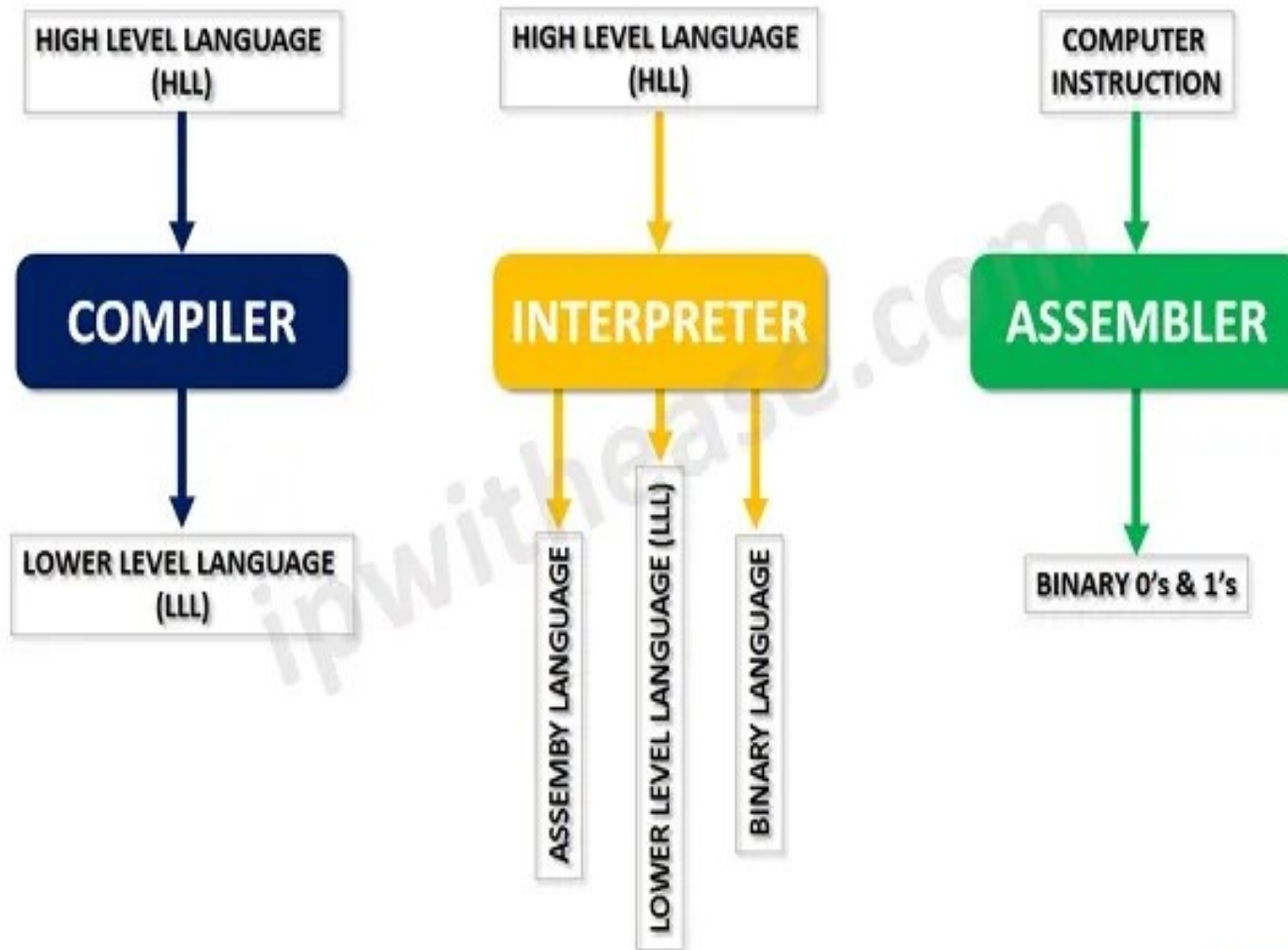


Machine  
Language

000110101  
101000110  
101000110

Object  
code

# Compiler vs Interpreter vs Assembler



PARAMETERS	COMPILER	INTERPRETER	ASSEMBLER
Conversion	It converts the high-defined programming language into Machine language or binary code.	It also converts the program-developed code into machine language or binary code.	It converts programs written in the assembly language to the machine language or binary code.
Scanning	It scans the entire program before converting it into binary code.	It translates the program line by line to the equivalent machine code.	It converts the source code into the object code then converts it into the machine code.
Error Detection	Gives the full error report after the whole scan.	Detects error line by line. And stops scanning until the error in the previous line is solved.	It detects errors in the first phase, after fixation the second phase starts.
Code generation	Intermediate code generation is done in the case of Compiler.	There is no intermediate code generation.	There is an intermediate object code generation.
Execution time	It takes less execution time comparing to an interpreter.	An interpreter takes more execution time than the compiler.	It takes more time than the compiler.
Examples	C, C#, Java, C++	Python, Perl, VB, PostScript, LISP, etc...	GAS, GNU

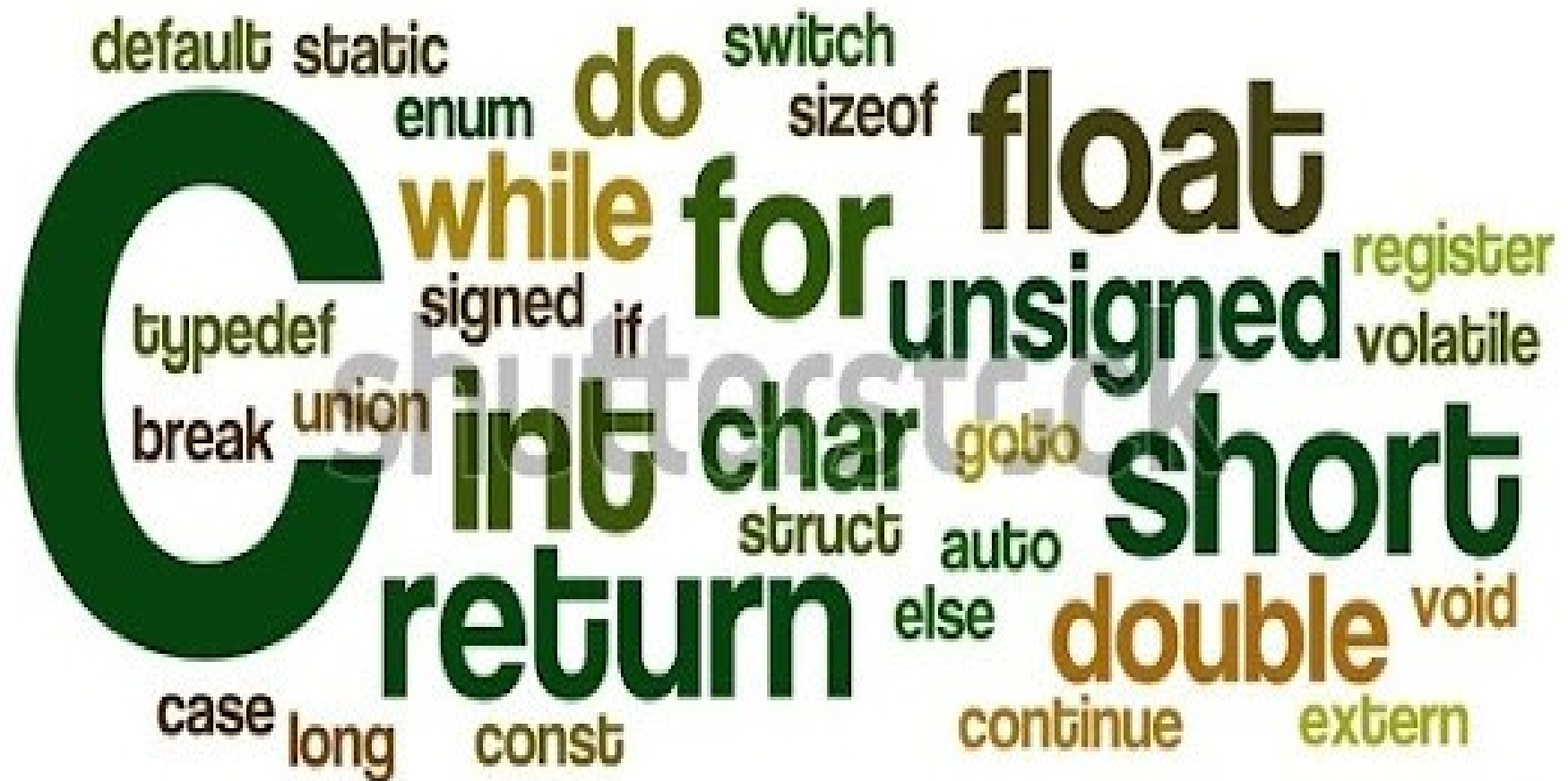




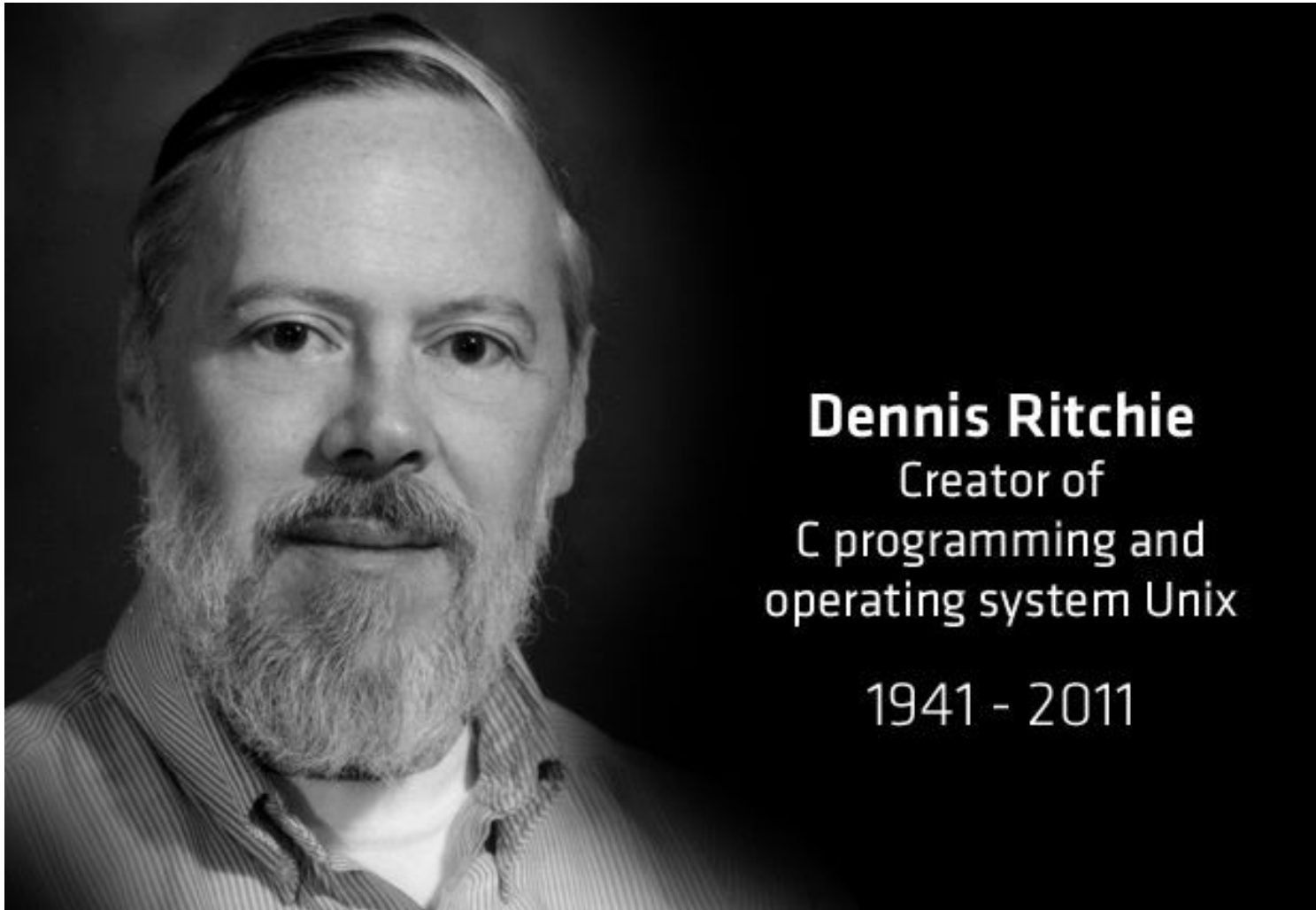
# Introduction

## C Programming

```
#include <stdio.h>
int main() {
    printf("Introduction to C!");
}
```







**Dennis Ritchie**  
Creator of  
C programming and  
operating system Unix

1941 - 2011

# 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.

# Why C language is so important?

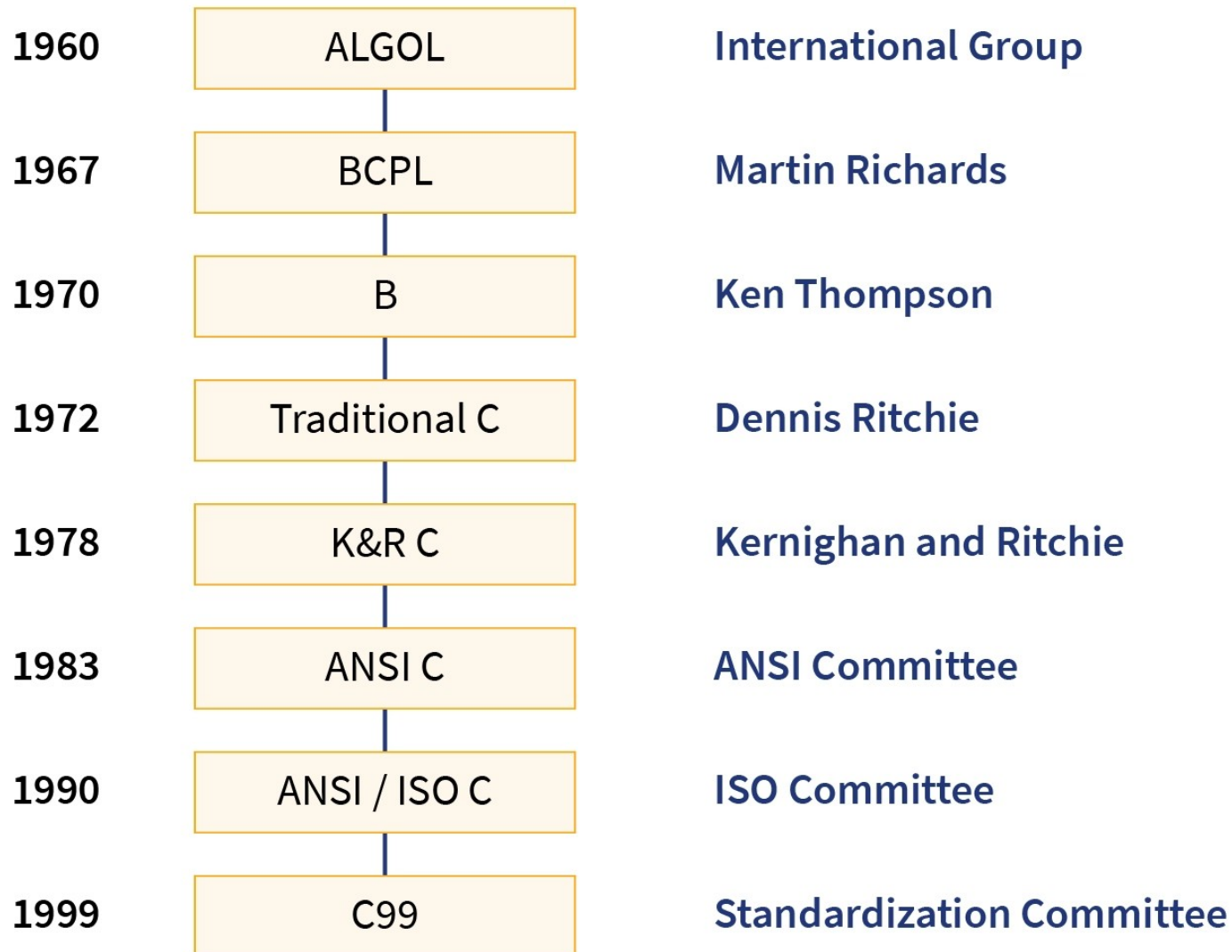
*Worth to know about C language*

- — *Oracle is written in C.*
- — *Core libraries of android are written in C*
- — *MySQL is written in C*
- — *Almost every device driver is written in C*
- — *Major part of web browser is written in C*
- — *Unix operating system is developed in C*
- — *C is the world's most popular programming language*

*For students*

- — *C is important to build programming skills*
- — *C covers basic features of all programming language*
- — *Campus recruitment process*
- — *C is the most popular language for hardware dependent programming*

# Generation of C language



# Introduction to ANSI Standard

The American National Standards Institute (ANSI) is a private, non-profit organization that administers and coordinates the U.S. voluntary standards and conformity assessment system. Founded in 1918, the Institute works in close collaboration with stakeholders from industry and government to identify and develop standards- and conformance-based solutions to national and global priorities.

Together, standards and technical regulations impact up to 93% of global trade. Globally relevant standards and the conformance measures that assure their effective use help to increase efficiency, open markets, boost consumer confidence, and reduce costs. And ANSI is the U.S. leader in fostering that potential for the benefit of businesses across every industry and consumers around the world.

# What is Procedural Programming?

- Procedural Programming is a programming language that follows a step-by-step approach to break down a task into a collection of variables and routines (or subroutines) through a sequence of instructions. In procedural oriented programming, each step is executed in a systematic manner so that the computer can understand what to do.
- The programming model of the procedural oriented programming is derived from structural programming. The concept followed in the procedural oriented programming is called the "procedure". These procedures consist several computational steps that are carried out during the execution of a program. Examples of procedural oriented programming language include – C, Pascal, ALGOL, COBOL, BASIC, etc.

# Advantages and Disadvantages of Procedure Oriented Programming

## Advantages

- We can access the same code without repeating it at various points in the software program.
- With the help of procedure oriented programming we will monitor the performance of the program.

## Disadvantage

- The code re-usability feature is not present in the procedure oriented programming. we have to write the same programming code to many times .
- We can not perform encapsulation ,inheritance etc in the procedure oriented programming.

# What is Programme ?

A computer program is a set of instructions for a computer to perform a specific task.



# Structure of C Programme

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

```
int main()
```

```
{
```

```
    printf("Hello, World! \n");
```

```
    return 0;
```

```
}
```

# Explanation

- 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 `void main()` is the main function where the program execution begins.
- The next line `printf(...)` is another function available in C which causes the message "Hello, World!" to be displayed on the screen.

# Compile and Run C Program

- 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.

# Comments in C language

- Single-line Comment in C. A single-line comment in C starts with ( // ) double forward slash. It extends till the end of the line and we don't need to specify its end. ...
- Multi-line Comment in C. The Multi-line comment in C starts with a forward slash and asterisk ( /\* ) and ends with an asterisk and forward slash ( \*/ ).
- The comments 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.

# Structure of C Programme

## Single Line Comments

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

```
int main()
```

```
{
```

```
    //my first program in C
```

```
    printf("Hello, World! \n");
```

```
    return 0;
```

```
}
```

# Structure of C Programme

## Multi Line Comments

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

```
int main()
```

```
{
```

```
    /* hello
```

```
        below is the first C program
```

```
        my first program in C
```

```
    */
```

```
    printf("Hello, World! \n");
```

```
    return 0;
```

```
}
```

Documentation section

---

Link section

---

Definition section

---

Global declaration section

---

main () Function section

{

Declaration part
------------------

Executable part
-----------------

}

---

Subprogram section

Function 1
------------

Function 2
------------

.....
-------

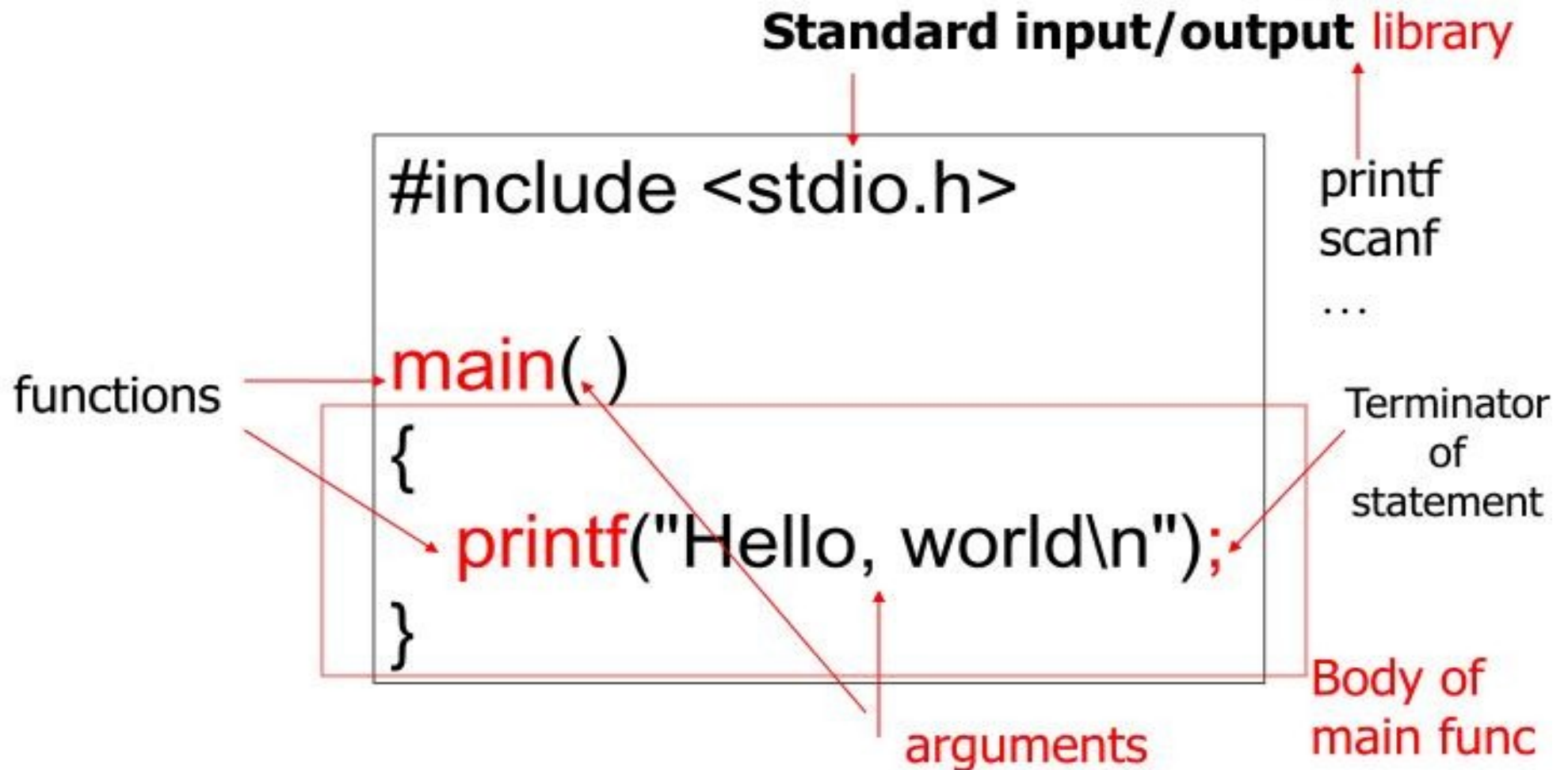
.....
-------

Function n
------------

(User defined functions)



# Example 1



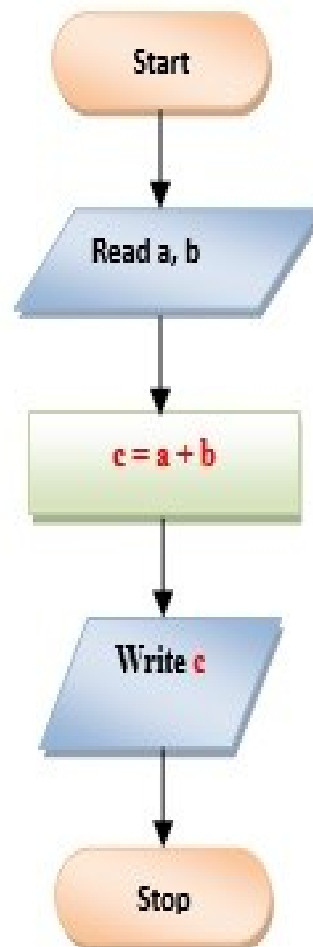


## To find sum of two numbers

### Algorithm

1. Start
2. Read a, b
3.  $c = a + b$
4. Print or display c
5. Stop

### Flowchart



### Program

```
#include<stdio.h>

int main()
{
    int a, b, c;

    printf("Enter value of a: ");
    scanf("%d", &a);

    printf("Enter value of b: ");
    scanf("%d", &b);
    c = a+b;

    printf("Sum of given two numbers is: %d", c);

    return 0;
}
```