# PROGRAMMING

## FOR PROBLEM SOLVING

## { C LANGUAGE }

# Module :- 1

## Start

# Module 1 :- Introduction To Programming


- Introduction to components of a computer system (disks, memory, processor, where a program is stored and executed, operating system, compilers etc ).
- Idea of algorithm : steps to solve logical and numerical problems. Representation of algorithm : flowchart/pseudo code with examples. From algorithms to programs;
- Source code, variables (with data types) variable and memory locations, type casting/type conversion, run time environment (static, dynamic location),
- Storage classes (auto, register, static, extern)
- Syntax and logical errors in compilation, object and executable code.

# -: Computer :-

**C** – Commonly

**O** – Operating

**M** – Machine

**P** – Particularly

**U** – Used for

**T** – Technical

**E** – Educational

**R** – Research

# Computer :-

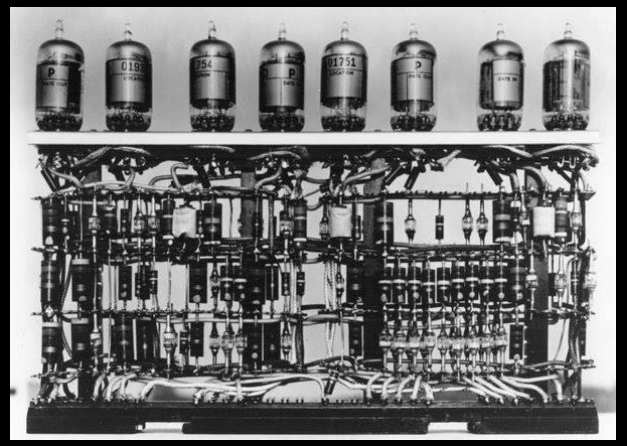
- Computer is a machine or device that perform processes, calculations and operations based on instruction provided by a software or hardware program.
- It has the ability to accept data (input), process it, and then produce outputs.
- Computer → Latin word → Computare

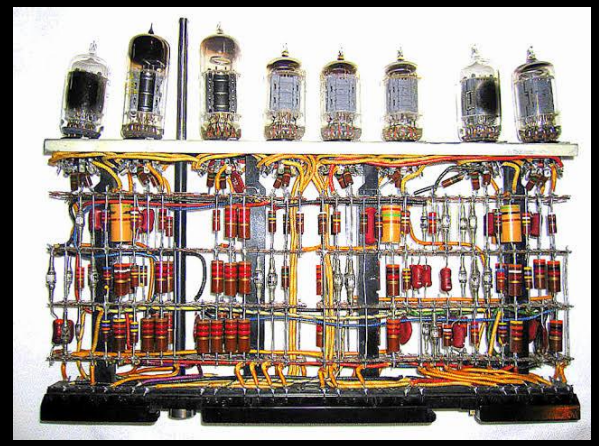
# History of computer :-


1. **ABACUS** :- arithmetic calculations
2. **Napier's bones** :- to perform multiplication and division
3. **Pascaline** :- addition and subtraction
4. **Stepped reckoner or Leibniz wheel** :- multiplication , division
5. **Difference engine** :- Computer :- by charles Babbage (father of modern computer). it can perform simple mathematical calculation

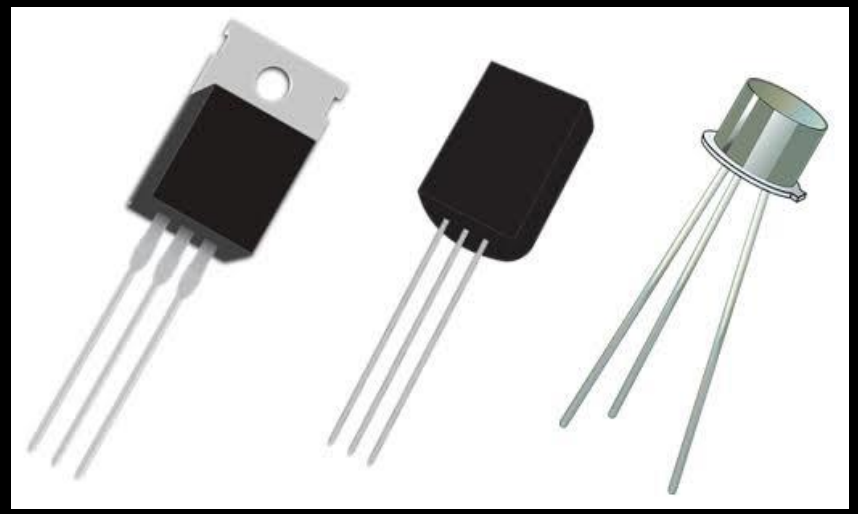
# Generation of computer :-

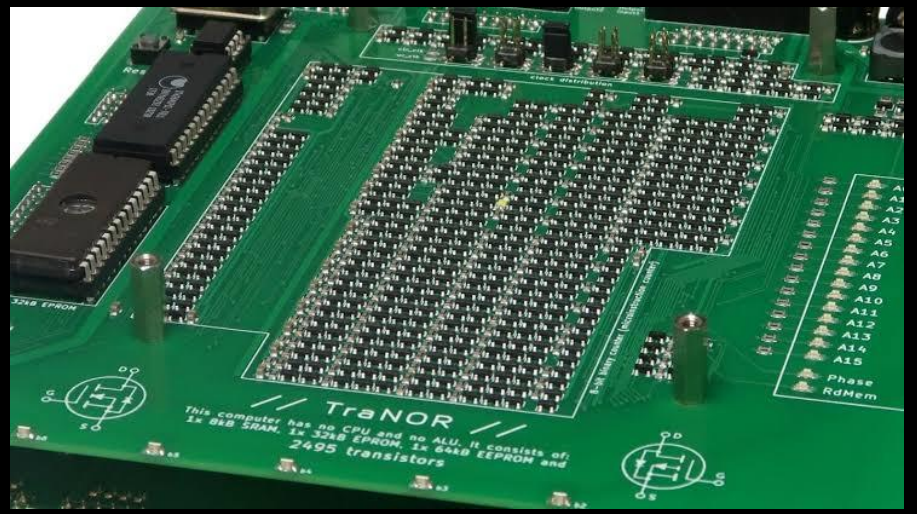
**First generation (1946 - 1959)** :- using vacuum tubes → machine language



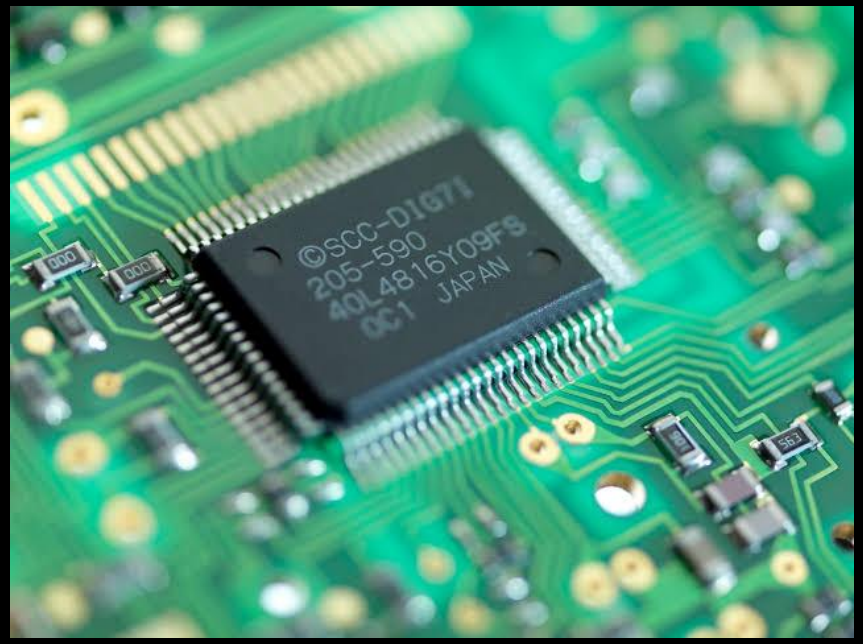


**Second generation (1959 - 1965)** :- using transistor → assembly language





**Third generation (1965 - 1971)** :- using integrated circuit → high level language




Third generation Computer

Integrated Circuit

**Fourth generation (1971-1980)** :- using microprocessors → very large scale integrated → high level language

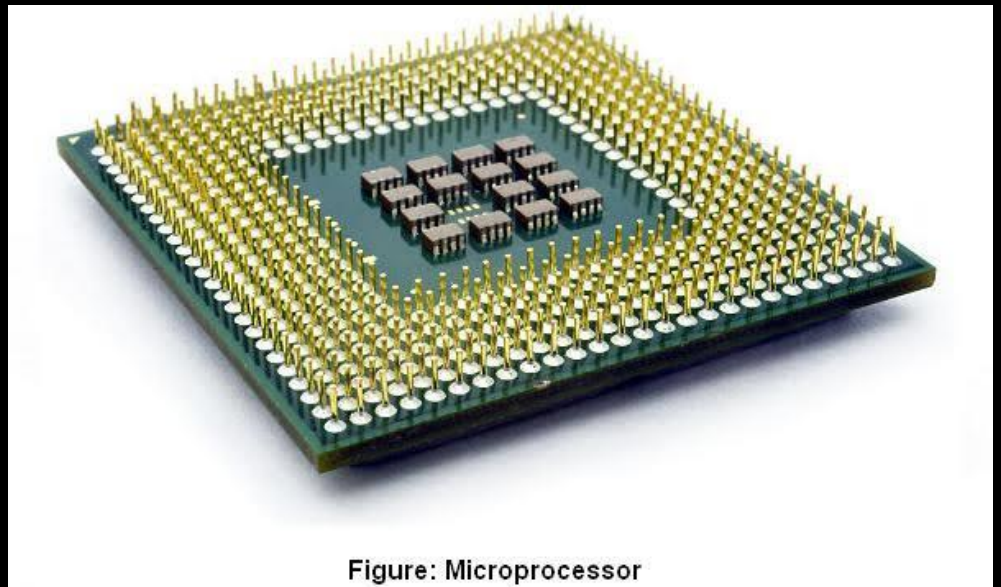
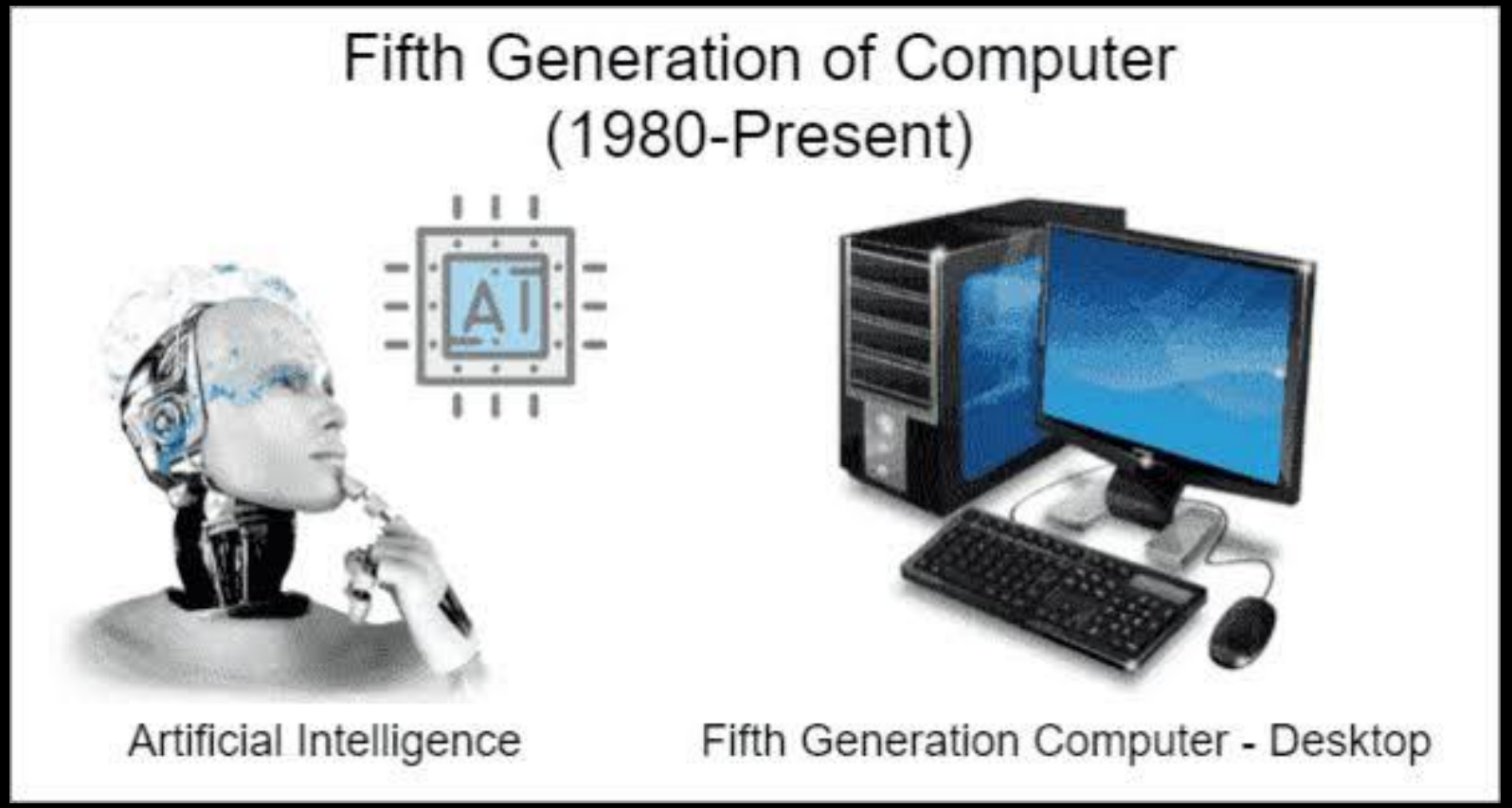


Figure: Microprocessor

**Fifth generation (1980 - present)** :- using artificial intelligence → ultra large scale integrated → high level language



# Generations of Computers

Vacuum Tubes

1st Generation

Transistors

2nd Generation

Integrated Circuits

3rd Generation

Very large scale integration

4nd Generation

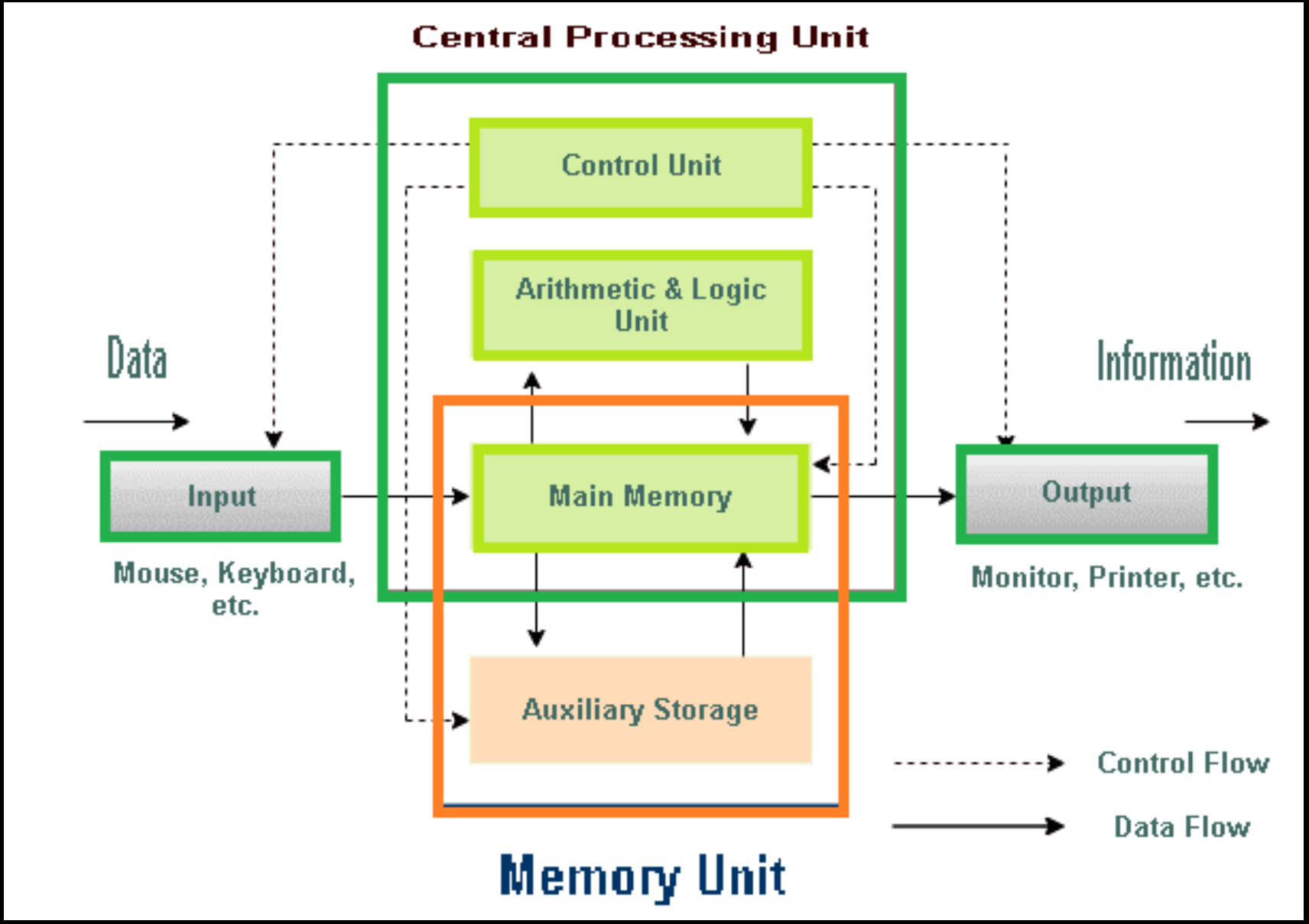

Ultra large scale Integration

5th Generation

Time

# Digital Computer :-

Block diagram



**Digital computer** :-

a digital computer is consider to be a calculating device that can perform arithmetic operations at enormous speed it is defined as a device that operate upon information / data to be able to process data

**Input unit** :-

computer need to receive data and instruction in order to solve any problem, therefore we need to input the data and instruction into the computer

example :- keyboard ,mouse ,scanner etc..

**Storage unit** :-

the storage unit of the computer holds data and instruction that are entered through the input unit

**Output unit** :-

the output unit of a computer provides the information and results of a computation to outside world

example :- printer , display ,speaker , projector etc..

**Arithmetic logical unit** :-

all calculation are performed in the arithmetic logical unit (ALU) of the computer it also does comparison and takes decisions the ALU can perform basic operations such as addition, subtraction, multiplication, division etc..

**Control unit** :-

it controls all other unit in the computer the control unit instructs the input unit where to store the data after receiving it from the user it controls the flow of data and instruction from the storage unit to ALU it also control the flow of results from the ALU to the storage unit the control unit is generally prefered as the central nervous system of the computer that control and synchronizes its working

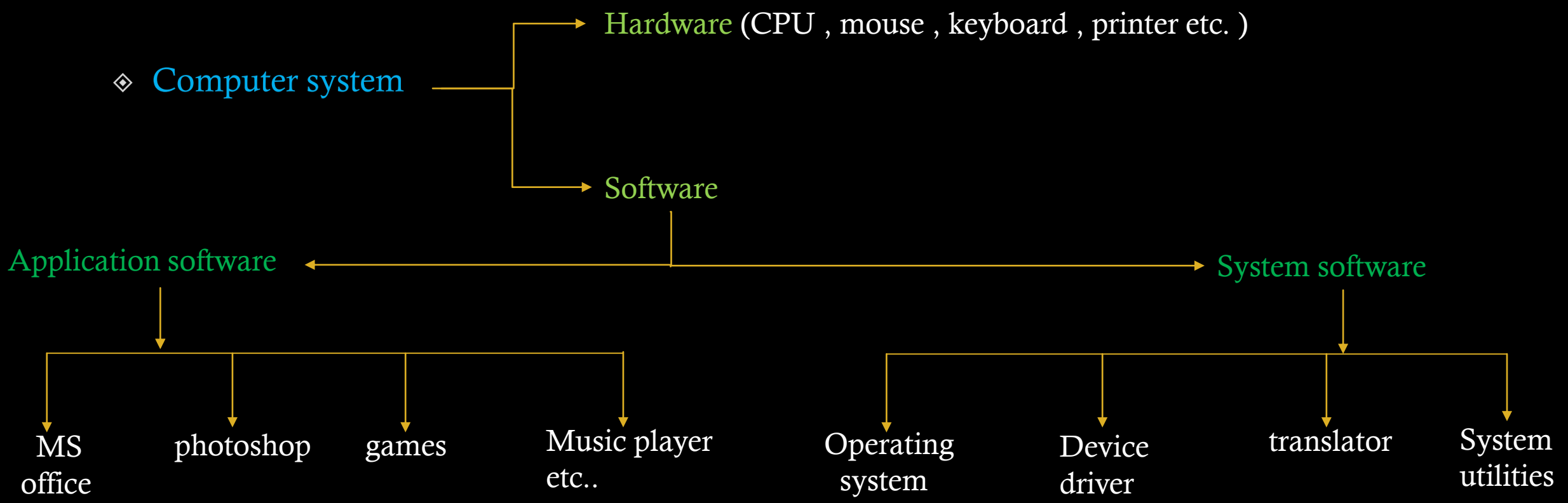
**Central processing unit (CPU) :-**

the control unit and arithmetic logical unit of the computer are together known as central processing unit

The CPU is like brain performs the following functions

1. It perform all calculations
2. It takes all decisions
3. It control all unit of the computer

# Components of computer :-



# Programming for problem solving :-

**Programming for problem solving** →

Means, problem solving with the help of programming

Example :- find average of three number

( x + y + z )/3

1. Get first number
2. Get second number
3. Get third number
4. Add first number, second number, third number = sum
5. Divide sum by the number of number 3
6. Display result

# Programming


- Letting the computer know how to solve a problem
  - Instructing the machine
  - Steps to follow
- Set of instruction to be provided to a computer to solve a problem

# Average of 20 number


## Method 1

1. Get first number
2. Get second number
3. Get third number

21. Add first number, second number, third number……twenty numbers = sum
22. Divide sum by the number of number 20 = result
23. Display result

## Method 2

1. Sum = 0
2. For 20 number
   repeat
   get number
   add the number with the current sum
   end
3. Result = sum/20
4. Display result

# Steps to be followed to solve a program

## Algorithm

How can we express algorithm ?

**Algorithm** :-

sequence of steps to be followed to solved a problem

**Expressed in some way**

Pseudo code :- English like language

**Flow chart** :- diagram

# How to write algorithm :-

Step 1: start

Step 2: always read a variable which you want to use.

Step 3: always read a variable which you are using a variable for input.

Step 4: for processing

Step 5: displaying a variable

Step 6: stop

# Example

1. Write a algorithm to add two number

Step 1: start
Step 2: get a , b
Step 3: sum = a + b
Step 4: display sum
Step 5: stop

2. Write a algorithm to swap two number using two variable

Step 1: start
Step 2: get a , b
Step 3: a = a + b
b = a - b
a = a - b
Step 4: display a & b
Step 5: stop

3. Write a algorithm to calculate simple interest (SI)

Step 1: start
Step 2: get p , r & t
Step 3: S.I = (p*r*t)/100
Step 4: display S.I
Step 5: stop

4. Write a algorithm to calculate average of three number

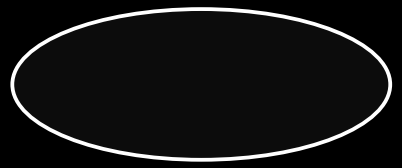
Step 1: start
Step 2: get a , b & c
Step 3: avg = (a+b+c)/3
Step 4: display avg
Step 5: stop

5. Write a algorithm to calculate sum and product of two number

Step 1: start
Step 2: get a , b
Step 3: sum = a+b
prod = a*b
Step 4: display sum & prod
Step 5: stop

# Flow chart :-

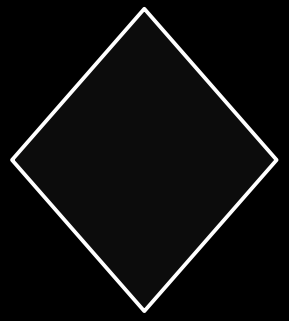
- A pictorial representation of an algorithm is called flow chart . In flow chart the step in algorithm are represented in the formal different shape.



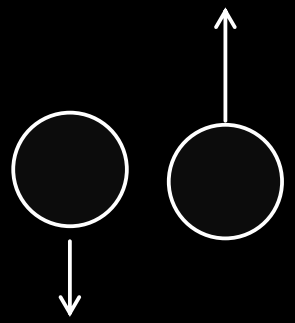
Start / stop

processing

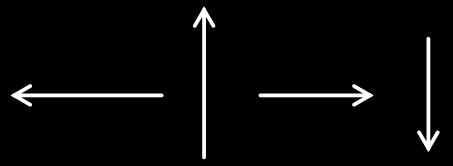
Input / output



Condition / decision



connector

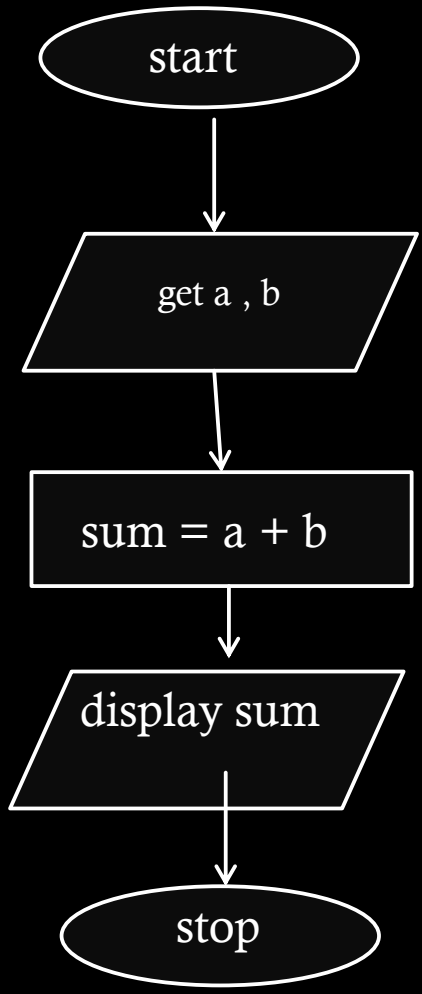


Direction of Flow

# Example

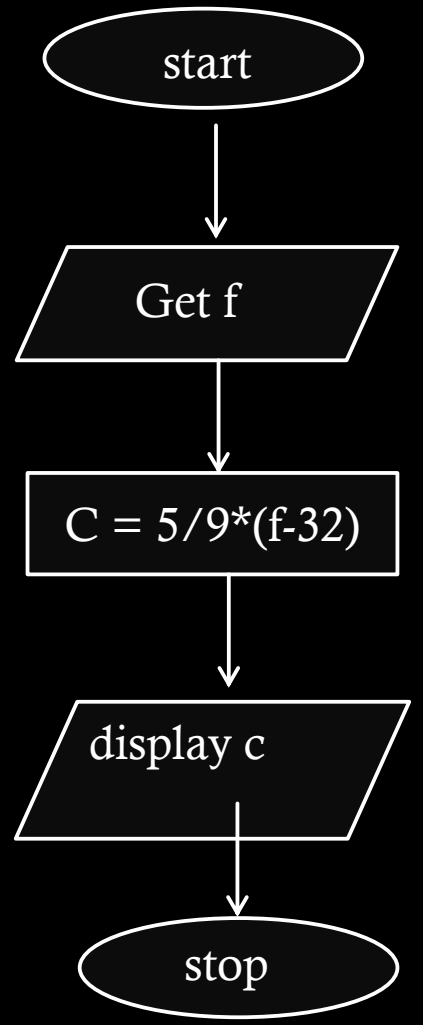
1. Write a algorithm & flow chart to add two number

Step 1: start
Step 2: get a, b value
Step 3: sum = a + b
Step 4: display sum
Step 5: stop



2. Write a algorithm & flow chart to convert temperature from Fahrenheit to celcius

Step 1: start
Step 2: get f value
Step 3: c = 5/9*(f-32)
Step 4: display c
Step 5: stop



# Pseudo code :- English like language


Example : calculate simple interest

```
Begin
Read p, n, r
Calculate Si
Si=(p*n*r)/100
Display Si
end
```


# Examples On Algorithm, Flow Chart, Pseudo Code

# Find area of rectangle

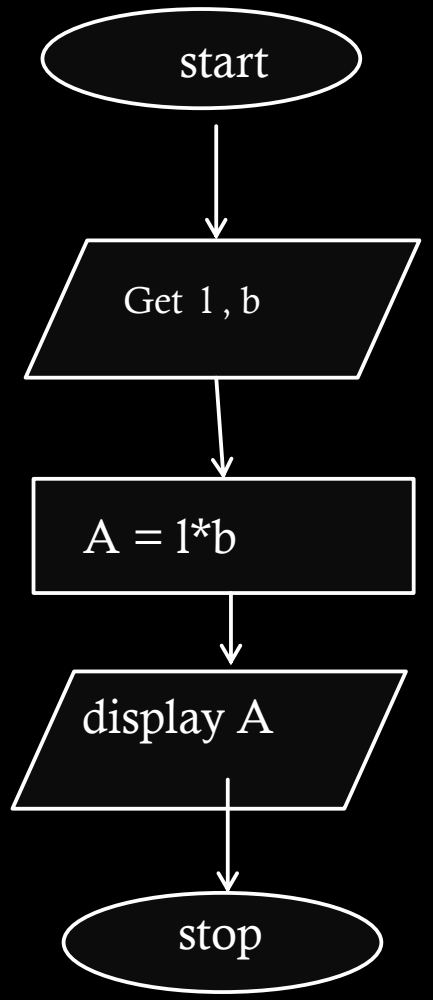
## Algorithm

Step 1: start
Step 2: get l, b value
Step 3: calculate A = l*b
Step 4:display A
Step 5: stop

## Pseudo code

Begin
Read l, b
Calculate A=l*b
Display A
end

## Flow chart



# Find area and circumference of circle

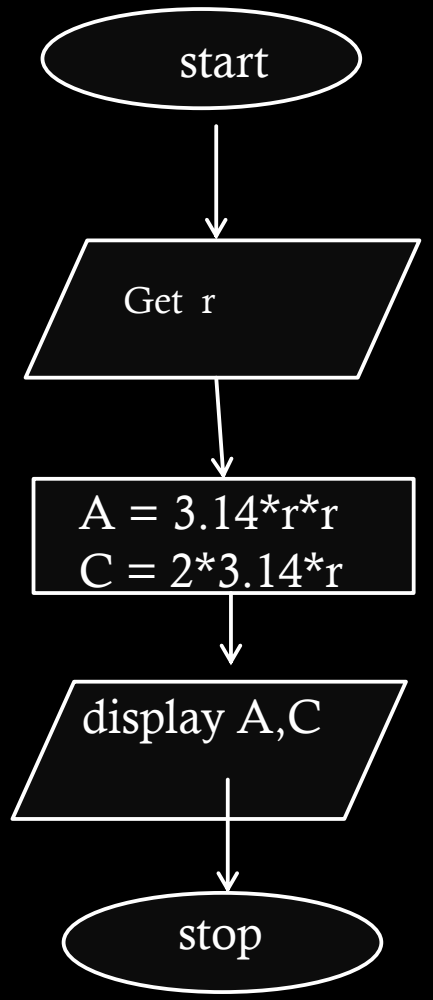
## Algorithm

Step 1: start
Step 2: get r value
Step 3: calculate A = 3.14*r*r
Step 4: calculate C = 2*3.14*r
Step 5: display A,C
Step 6: stop

## Pseudo code

Begin
Read r
Calculate A and C
A = 3.14*r*r
C = 2*3.14*r
Display A,C
end

## Flow chart



# Find simple interest

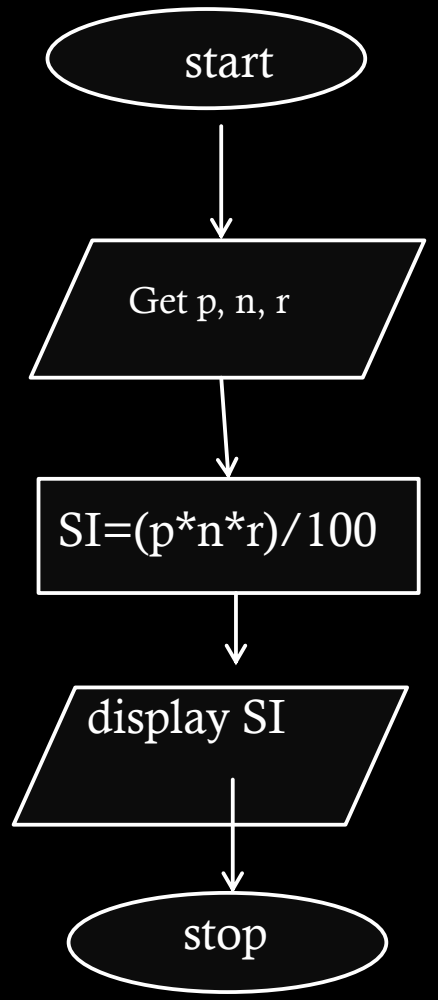
## Algorithm

Step 1: start
Step 2: get p, n, r value
Step 3: calculate SI=(p*n*r)/100
Step 4: display SI
Step 5: stop

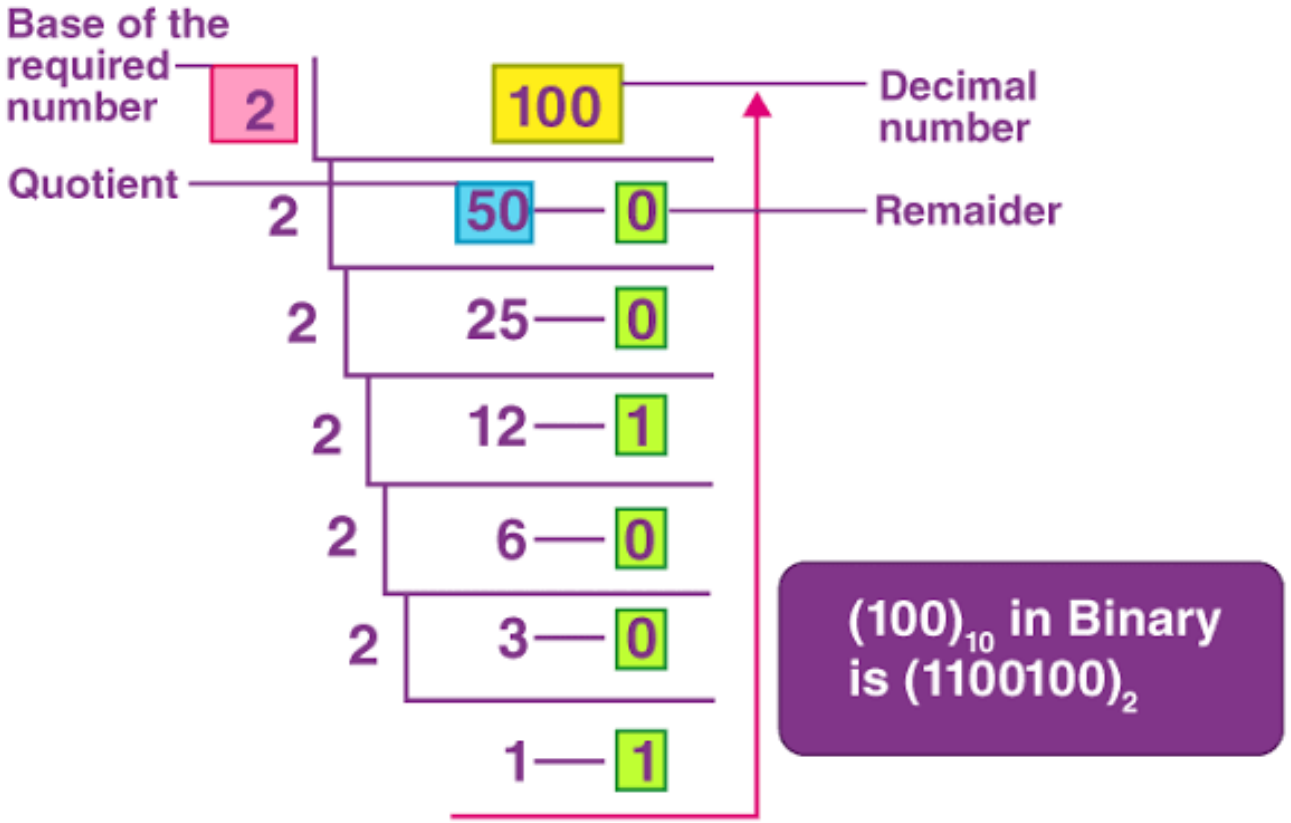
## Pseudo code

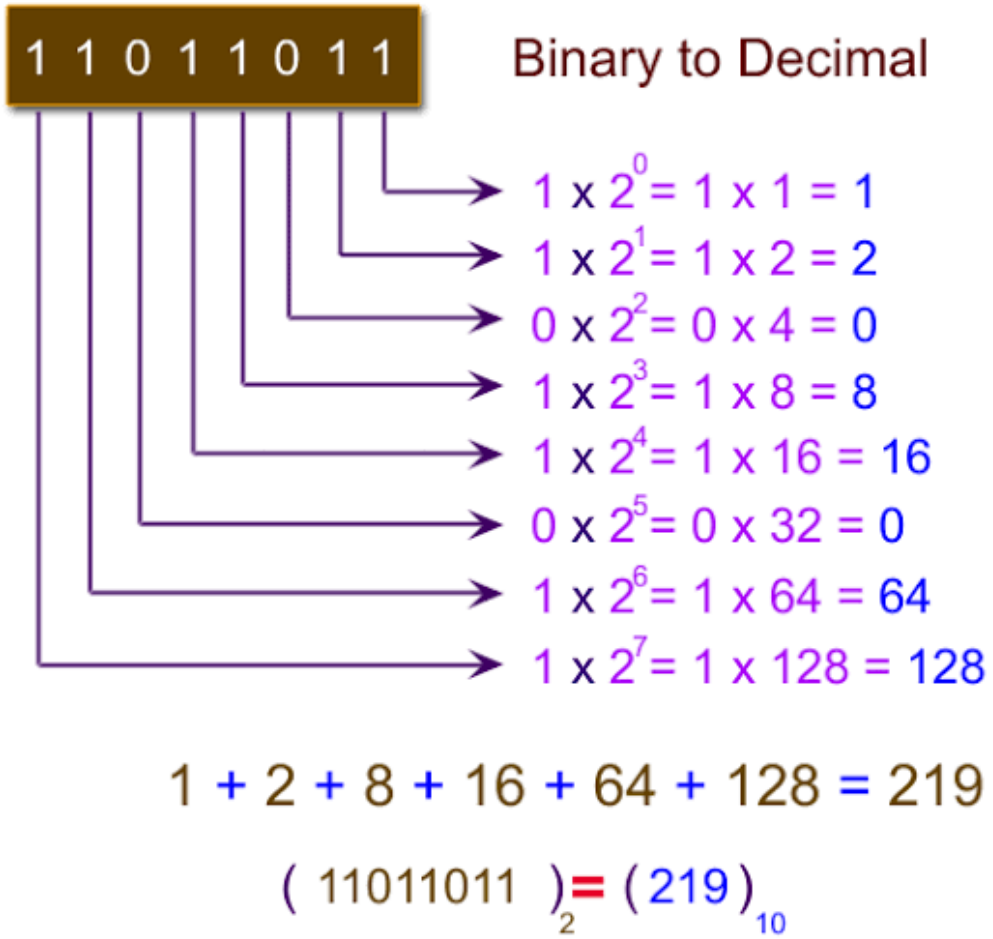
Begin
Read p, n, r
Calculate SI
SI=(p*n*r)/100
Display SI
end

## Flow chart



# Number system :-




# Algorithm

programmer

Program

Language understood by a computer

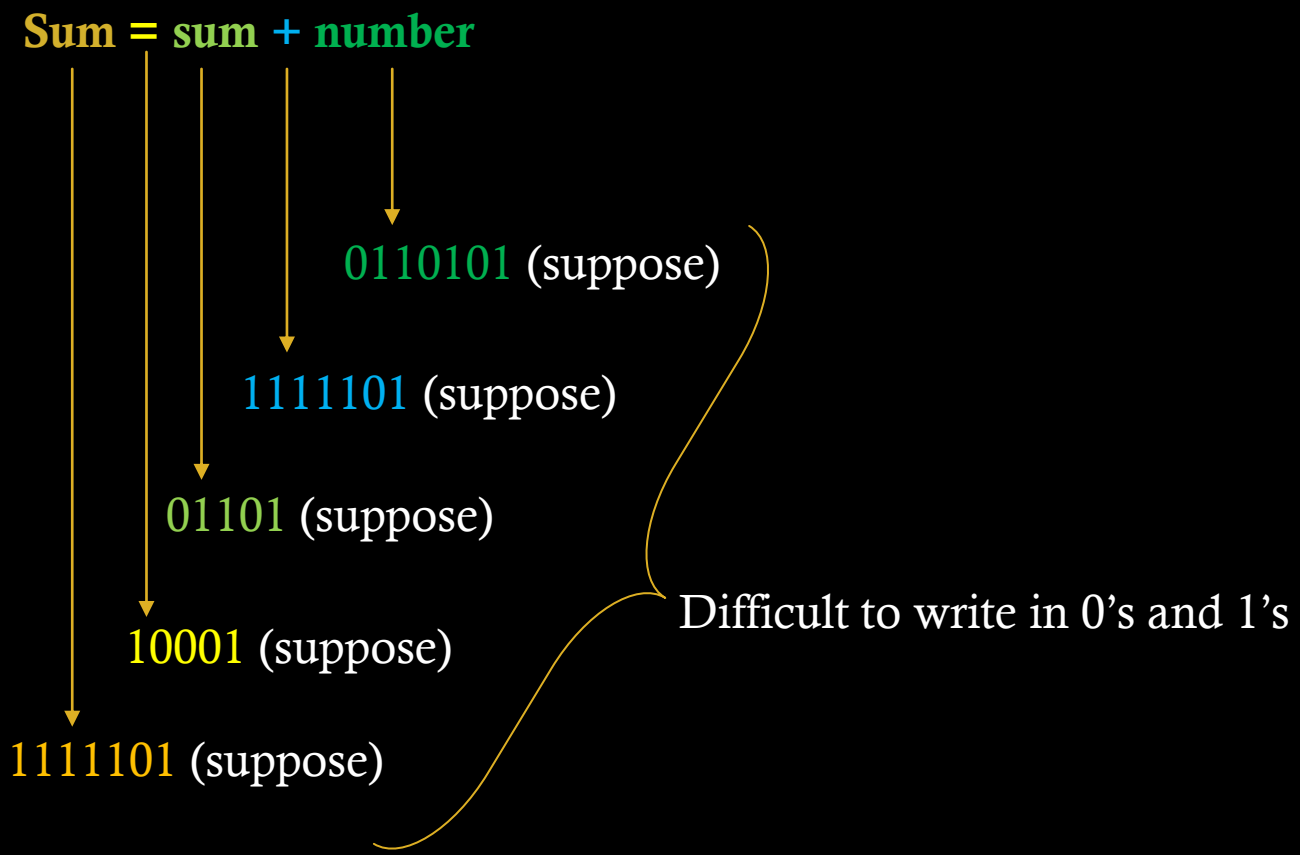
0's and 1's (switch is on or off)

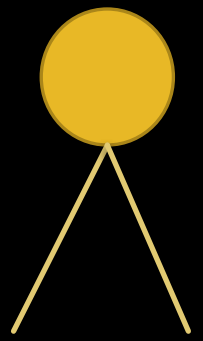
binary

Expressed in some way

Pseudo code

Flow chart


Sum = sum + number
0110101 (suppose)
1111101 (suppose)
01101 (suppose)
10001 (suppose)
1111101 (suppose)
Difficult to write in 0's and 1's



Flow char

Pseudo code
(English)

Computer
(Machine)

Pattern of 0's and 1's

Machine language

Translator

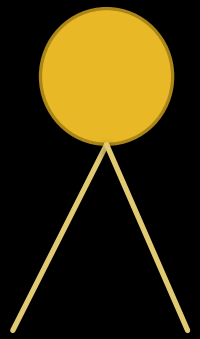
Go to the bank ? Mean what for withdrawn or debit
English me ek word ka matlav alag alag hota hai
Isliye English me likhna beneficial nahi hoga

# Programming language

**Low level language :-** sequence of 0's and 1's

**Assembly language :-** it is directly communicate with computer's hardware

**High level language :-** it refers to the programming language that allow the use of symbolic operators eg. + , - , * , /  etc..



Computer
(Machine)

Pattern of 0's and 1's

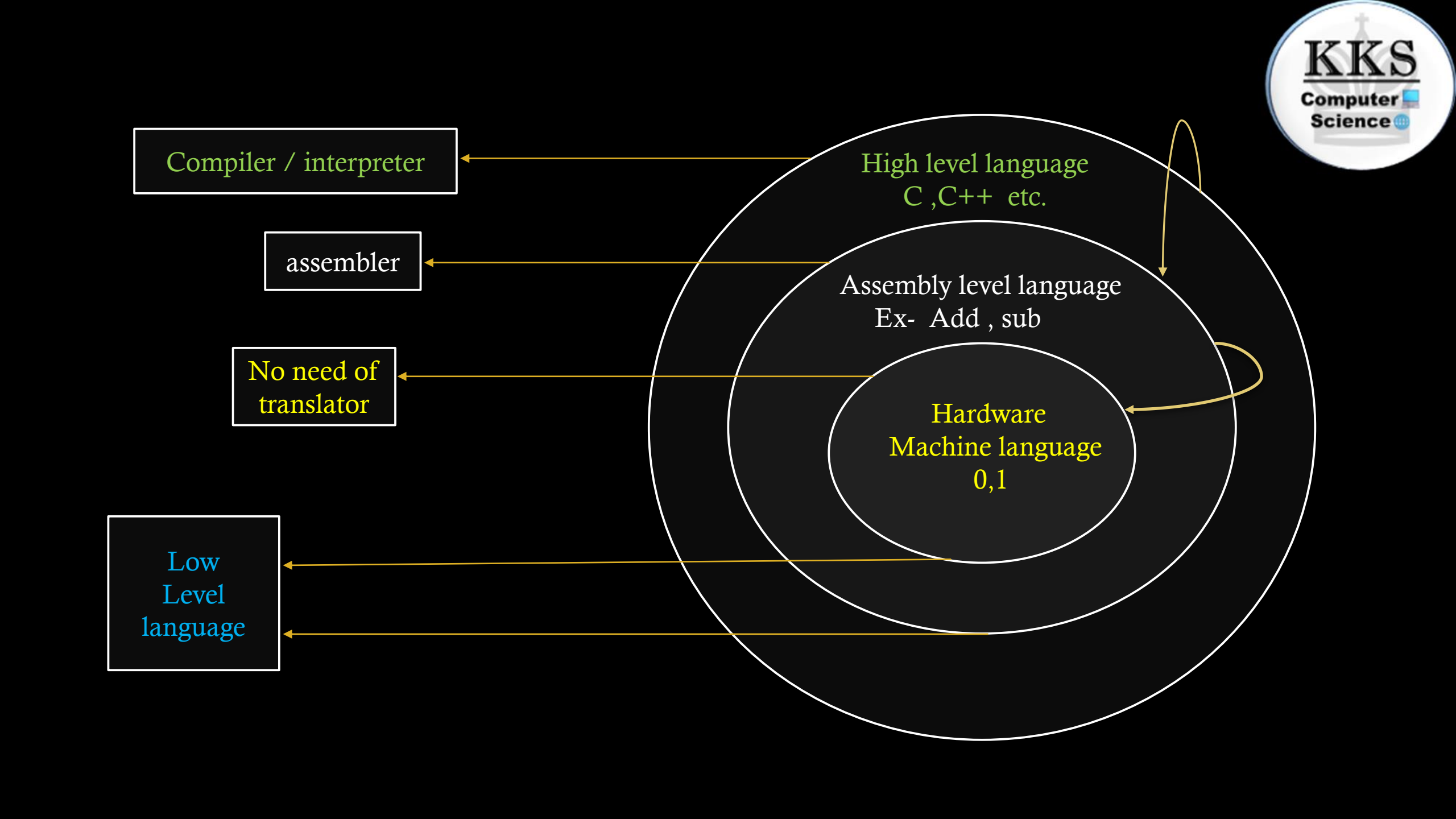
High level language → Translator → Machine language

C → C compiler
C++ → C++ compiler
java → java interpreter
Etc..

It convert high level language to machine level language

# Translator :-

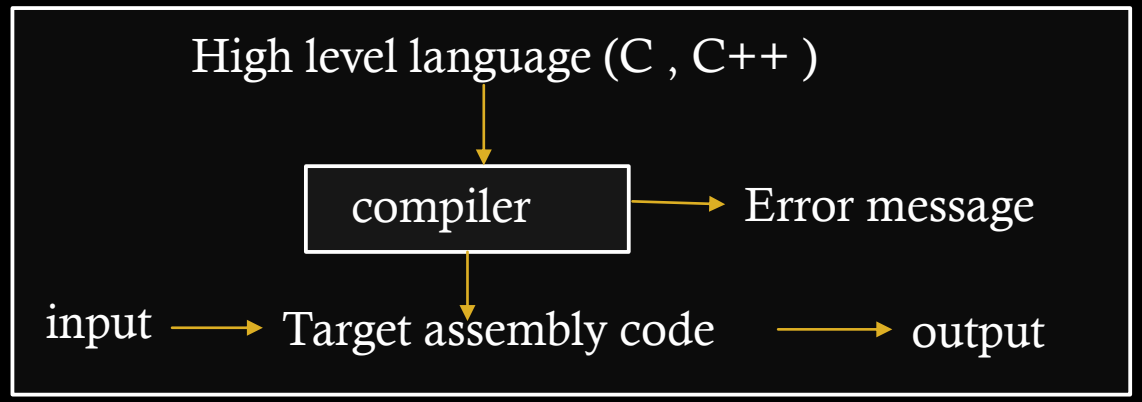

1. **Assembler** :- It convert assembly language to machine level language
2. **Compiler** :- It convert high level language to machine level language
3. **Interpreter** :- It convert high level language to machine level language


Compiler / interpreter
assembler
No need of translator
Low Level language
High level language
C ,C++ etc.
Assembly level language
Ex- Add , sub
Hardware
Machine language
0,1

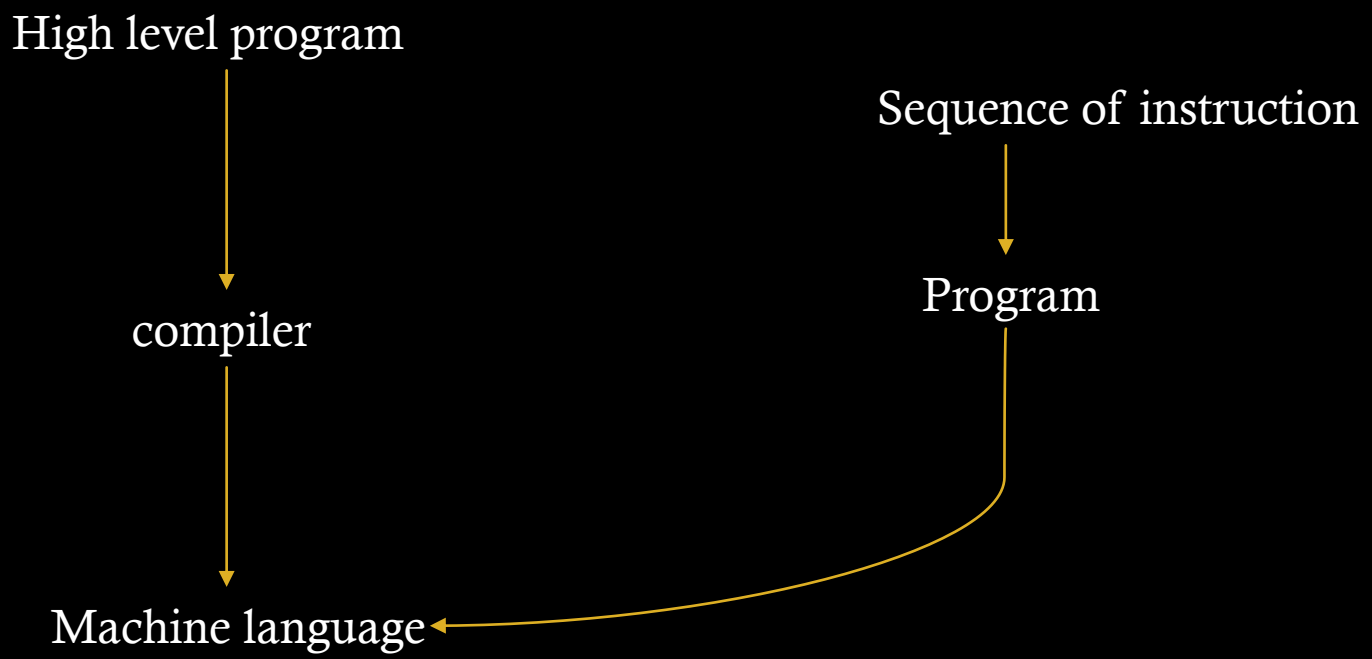
# Compiler V/s interpreter

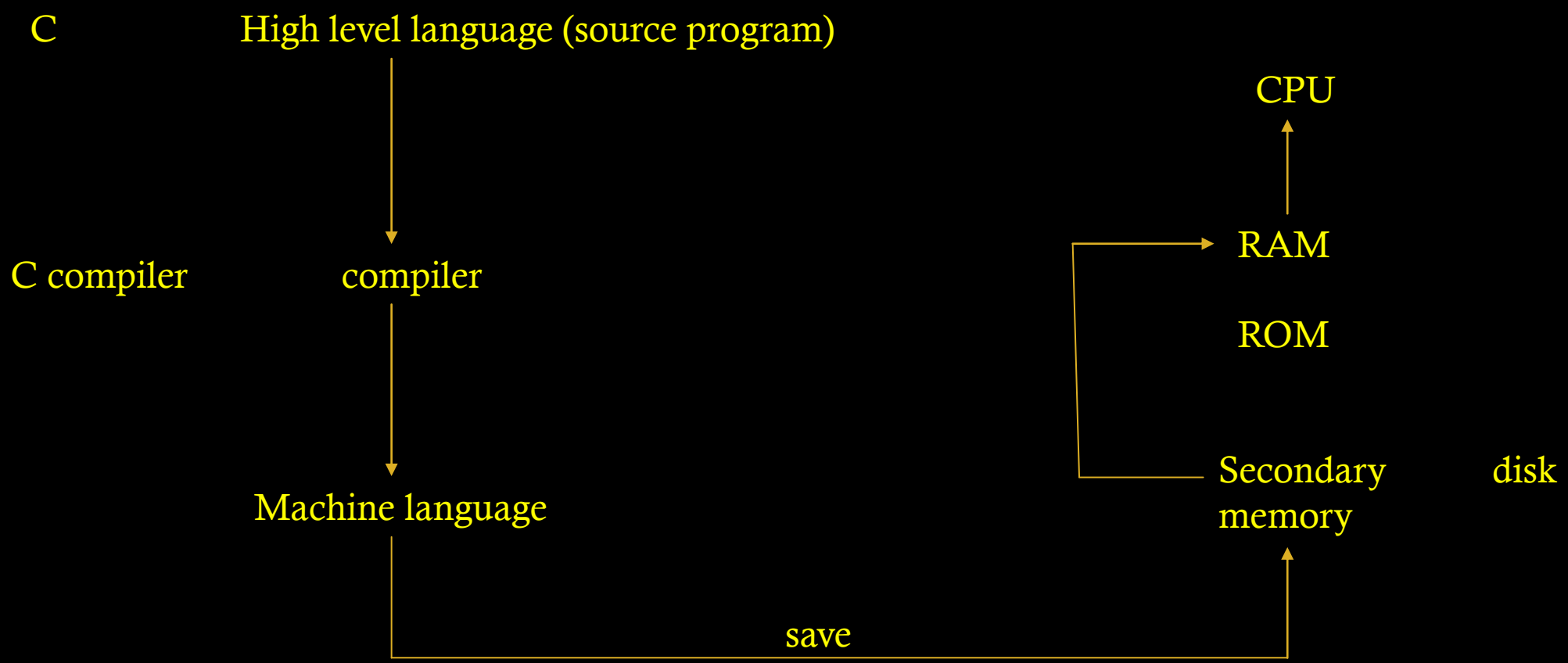
1. Compiler translate source code in to object code as whole
2. It create object file
3. Execution is fast
4. Program not required to translate each time to run the program
5. Most high level language uses compiler

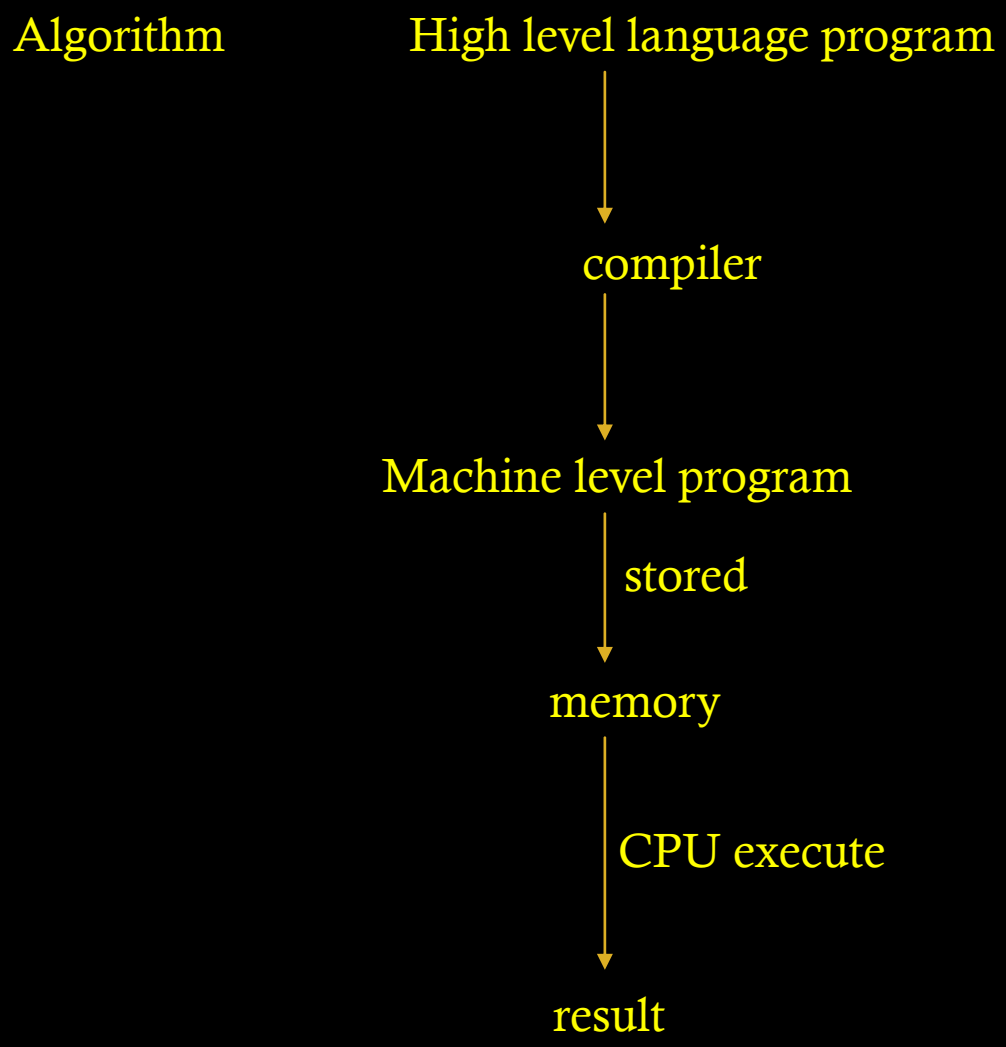
Eg. C , C++ , JAVA etc..




1. Interpreter statement of the source code one by one an execute immediately
2. It does not create object file
3. Execution is slow
4. Program required to translate each time to run
5. Easy to correct the mistake in source code
6. Few languages uses interpreter
   Eg. LISP , Pythod , basic etc..


High level program
compiler
Machine language
Sequence of instruction
Program


C
High level language (source program)
C compiler
compiler
Machine language
save
CPU
RAM
ROM
Secondary memory
disk

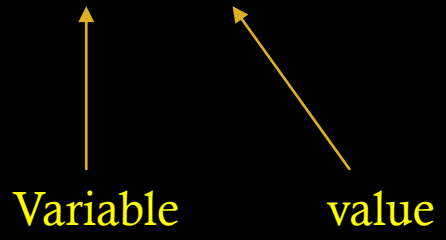

Algorithm
High level language program
compiler
Machine level program
stored
memory
CPU execute
result

# Variable

Sum = Sum + number

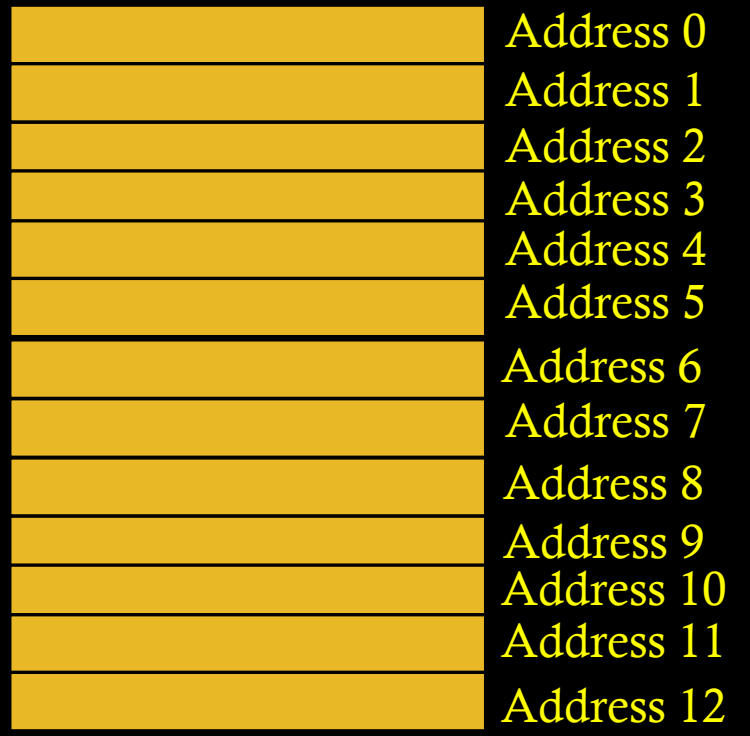
Max = number

Area = 5.2



# Memory map

Every variable is mapped to a particular memory address



High language program

Source code → compiler → machine language

Sum = 25

A variable can have different value

Sum 37

Sum = 25

X = 56

Sum = sum + x

---

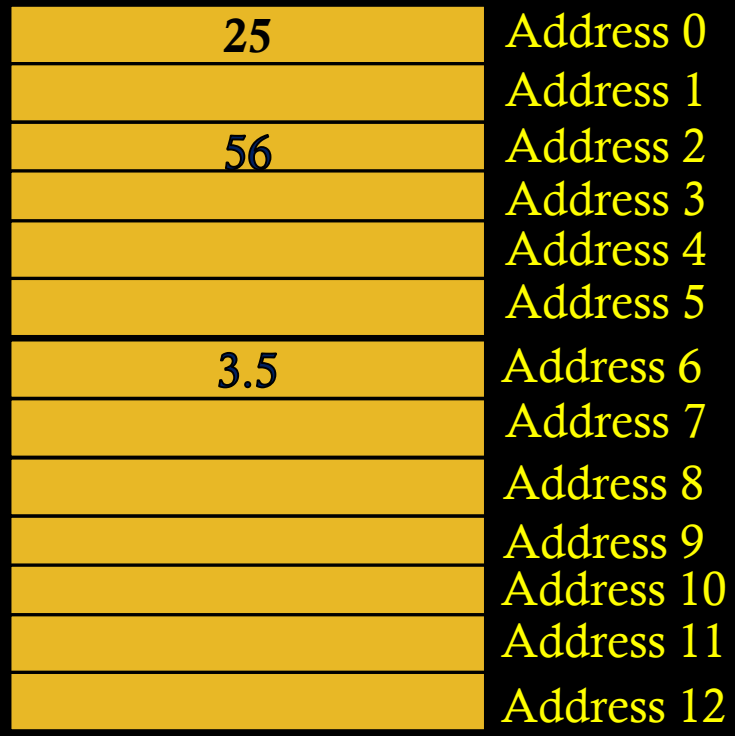
High language program → compile → assigns distinct location to each variable

sum = 25

x = 56

y = 3.5

| variable | address |
|---|---|
| sum | Address 0 |
| x | Address 2 |
| y | Address 6 |



X = y  x=5

X equals y  y=5

= assignment

X=y  X ⟵ y

X is assigned y

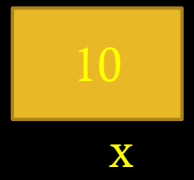
X is getting the value of y

Memory location with some distinct address

X = y  X ⟵ y

X=10

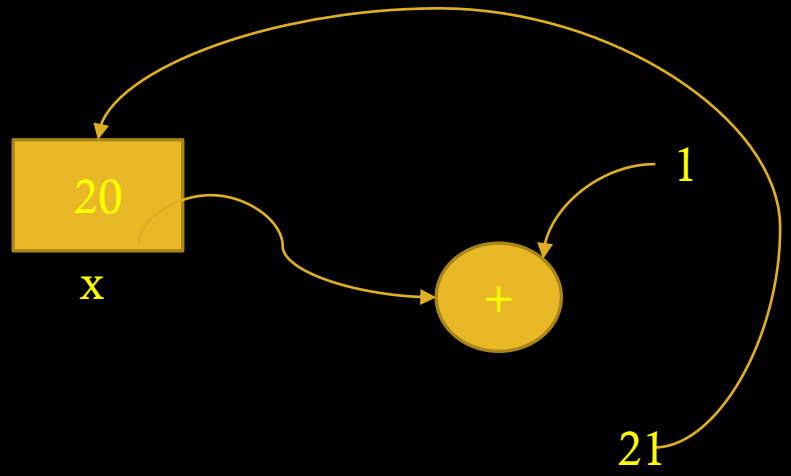
X is assigned the value 10



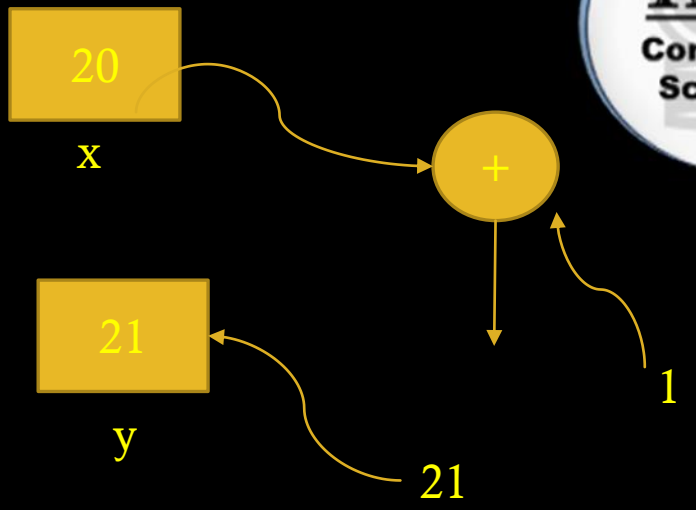
X = x+1

X ⟵ x+1

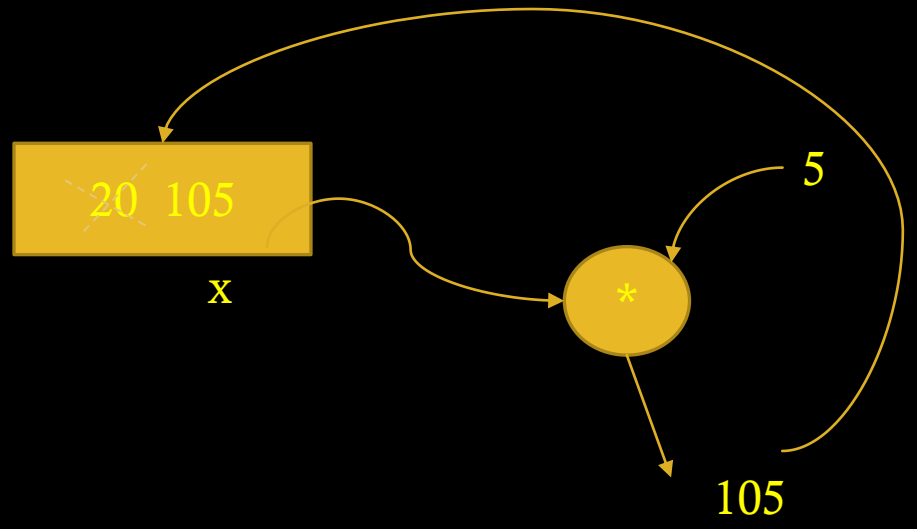
value after computing of the expression

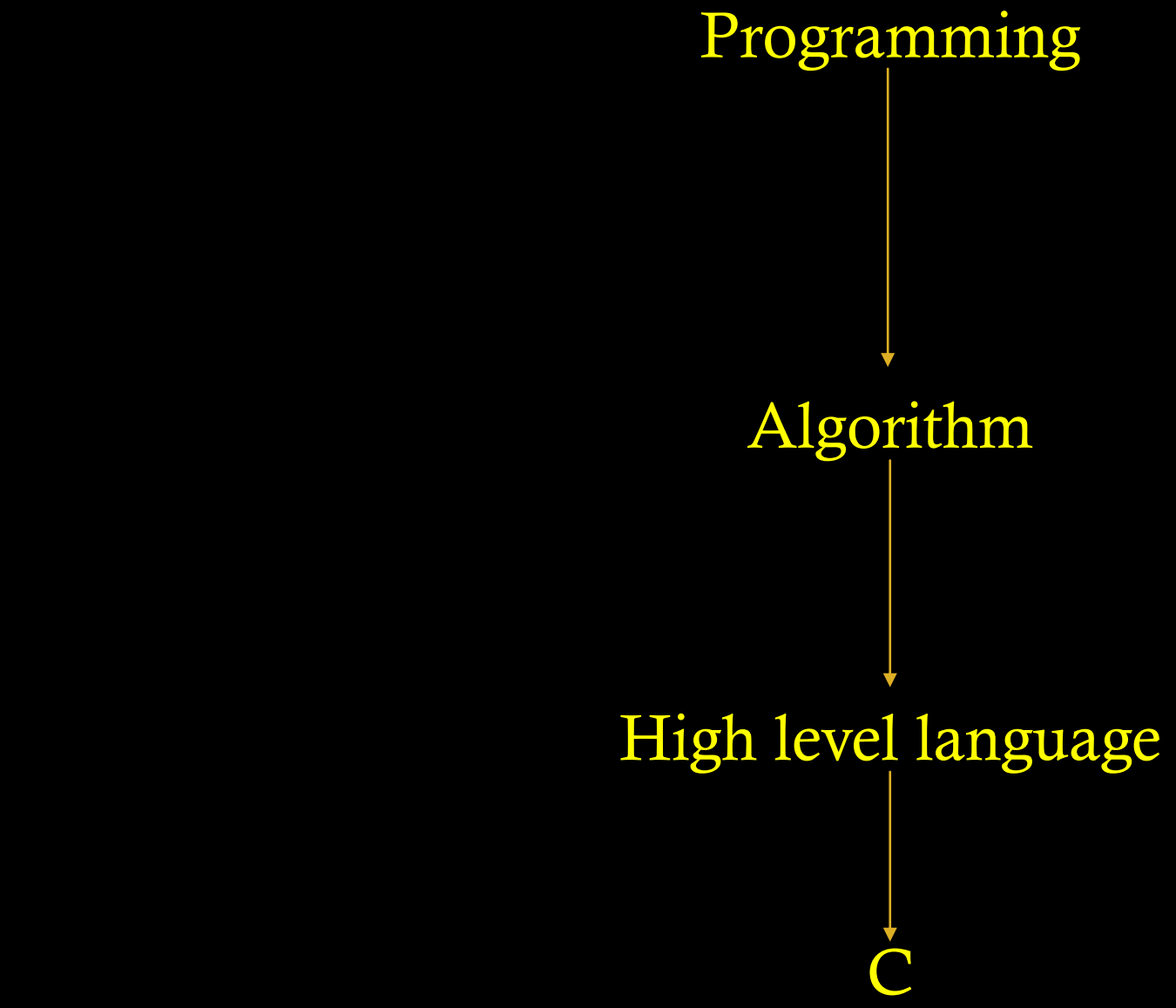



y = x+1




x = x*5




Programming
Algorithm
High level language
C

# C language

# Start

# -: Introduction to ‘C’ language :-


## Definition :-

- It is a basic , general – purpose programming language.
- It is the base of all high level programming language so it is called basic programming language.
- It is developed by dennis Ritchie.
- It is developed at AT and T’s bell laboratory (USA)
- It was developed in 1972
- Basically it was developed to implement UNIX operating system

# History of c :-

- ALGOL → 1960 → INTERNATIONAL GROUP
- BCPL → 1960 → MARTIN RICHARDS
- B → 1870 → KEN THOMPSON
- C → 1972 → DENNIS RITCHIE
- ANSI C → 1989 → ANSI COMMITTEE
- ANSI / ISO C → 1990 → ISO COMMITTEE

# Application :-

- No body can learn directly C++ , java
- Skill of c helps to OOP (object oriented programming
- The OS like window , linux , android written in C
- C programe directly intract with hardware (h/w)
- Compilers
- Database system
- Network drivers
- Interpreters
- Editors
- Graphics packages etc..

# Features :-

- Simple
- Powerful
- Portable
- Machine independent
- Structure oriented
- Middle level programming language
- High speed
- High efficiency
- flexible

- C is a precedural programming language
- C is a case sensitive language

Program :- set of instruction / statement

Instruction/statement :- set of tokens

Tokens :- set of character set

Token :-

keywords
Identifier / variable
Constant
Operator
String
Special symbol

# Keyword :-

- The word has a predefined meaning is called keywords
- It's functionality is also predefined
- It can not be used as an identifier
- There are 32 keywords are used in c

## Keywords in C Language

| Keywords in C Language | | | |
|---|---|---|---|
| auto | double | int | struct |
| break | else | long | switch |
| case | enum | register | typedef |
| char | extern | return | union |
| continue | for | signed | void |
| do | if | static | while |
| default | goto | sizeof | volatile |
| const | float | short | unsigned |

# variable


Definition :-

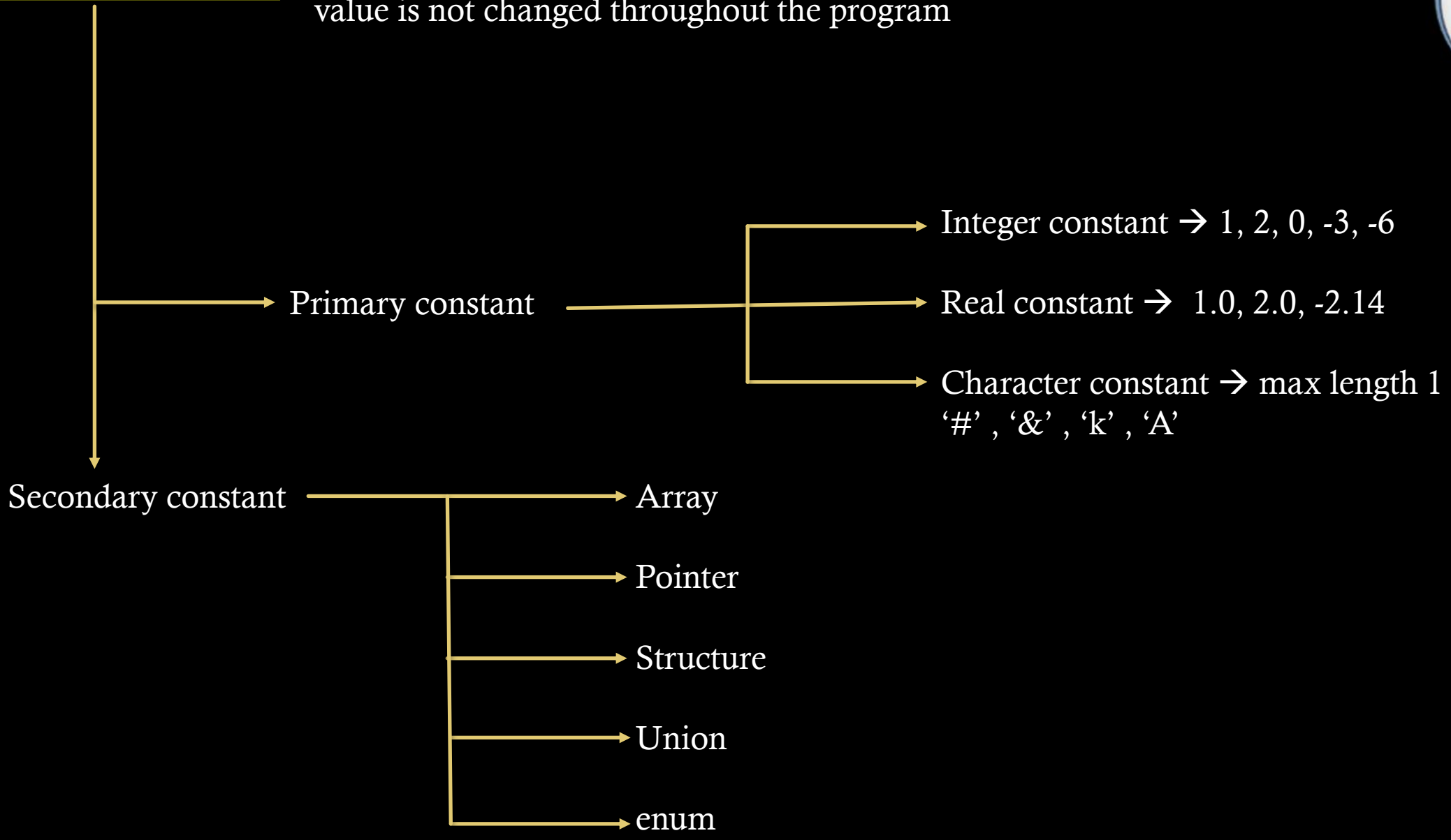
it is a name of storage space which is used to store data
Its value is changeable
It always contains last value stored to it
It is always declared with data type

## Rules to declare a variable :-

The first latter of a variable should be alphabet underscore(_)
The first latter of variable should not be digit
After first character it may be combination of alphabets and digit.
Blank space are not allowed in variable name .
Variable name should not be a keyword

# Constant :-

Its value fix throughout the program that mean constant are those variable which value is not changed throughout the program



# Constant :-


definition:-
An element of program whose value can not be changed at the time of execution of program is called constant
It is also called literals
It may be int ,float and character data type

Use of constants in program

There are two way of using constants in the c program
Using const
Using #define


## Rules for constructing integer constant :-

It must have at least one digit
It must not have a decimal point
It may be positive or negative
The range of integer constant is between -32768 to +32768
No comma or blank space are allowed in integer  point constant

## Rules for construting floating point constant :-

It must have atleast one digit
It must have a decimal point
It may be positive or negative
No comma or blanck space are allowed in floating constant

## Rules for constructing character constent :-

It is a single alphabet ,digit or special symbol
The length of character constant is 1 character
Character is enclosed within single quotes (example char c='A';)

# Data type :-


## Definition

1. Data type determine the type of data a variable will hold
2. It is also tell about the space that a variable stores
3. 1.It is a type of data which is used in the program.
4. 2.There are many predefined data types in c library like int, char, float etc.

## Data type

### Predefine data type

- Int
- Float
- Char
- Double
- Long

### Drived data type

- Array
- String
- pointer

### User defined data type

- Struct
- Union
- enum

# Character Type

| Data Type | Size in bytes | Range |
|---|---|---|
| char | 1 | -128 to +127 |
| signed char | 1 | -128 to +127 |
| unsigned char | 1 | 0 to 255 |

# Float Type

| Data Type | Size in bytes | Range |
|---|---|---|
| float | 4 | 3.4E-38 to 3.4E+38 |
| double | 8 | 1.7E-308 to 1.7E+308 |
| long double | 10 | 3.4E-4932 to 1.1E+4932 |

# Integer Type

| Data Type | Size in bytes | Range |
|---|---|---|
| short | 2 | -32768 to +32767 |
| int | 2 | -32768 to +32767 |
| unsigned int | 2 | 0 to 65536 |
| long | 4 | -2147483648 to +2147483647 |
| unsigned long int | 4 | 0 to 4,294,967,295 |

| Variable type | Keyword | Byte required | Format specifier |
|---|---|---|---|
| integer (signed) | int | 4 | %d |
| integer (unsigned) | unsigned int | 4 | %u |
| character (signed) | char | 1 | %c |
| character (unsigned) | usigned char | 1 | %c |
| float (signed) | float | 4 | %f |
| double | double | 8 | %lf |
| long double | long double | 10 | %lf |
| | | | |
| | | | |

# Variable declaration:-

```
int rollno;
float marks;
char grade;
```

Here rollno is a variable of type int ,marks is a variable of type float and grade is a variable of type char

# Variable initialization:-

```
int rollno=201;
float marks=85.6;
char grade ='a';
```

Here 201 is the value of rollno,85.6 is the value of marks and
A is the value of grade.
Character value is always written in single quotes

# Other method of variable initialization:-

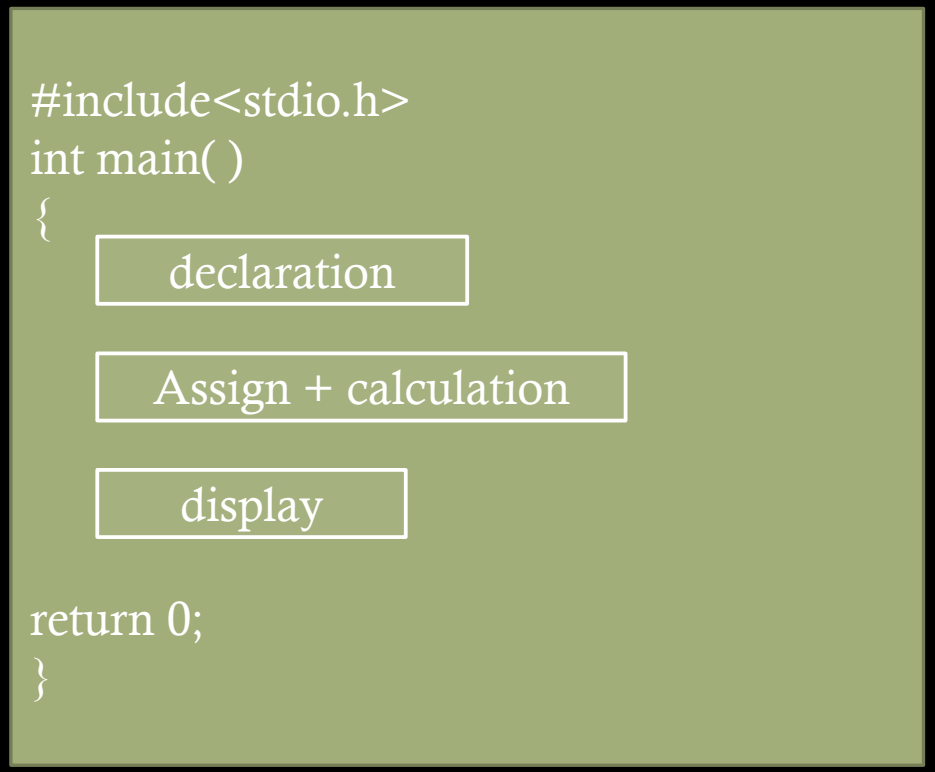
```
int rollno;
float marks ;
char grade ;

rollno =201;
marks=85.6;
grade='a';
```

## Example

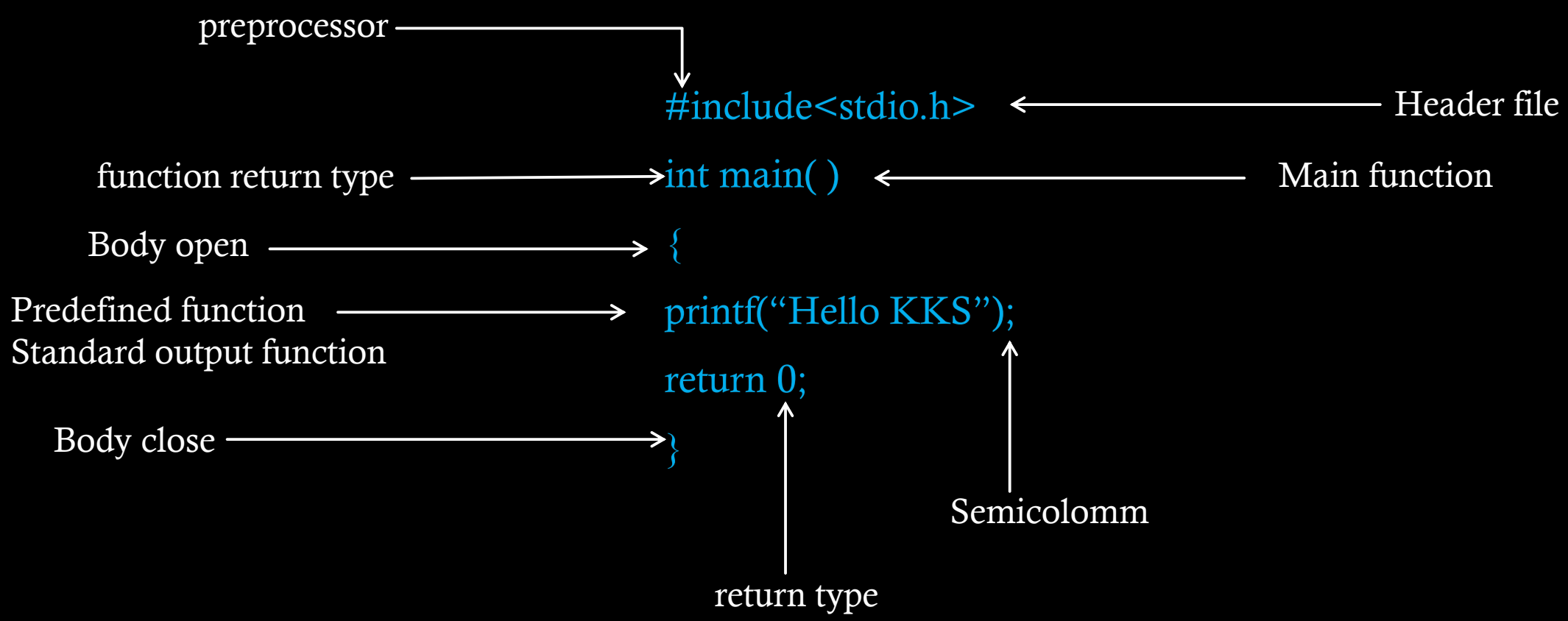
```
int rollno = 201;
float marks = 85.6;
char grade = 'a';
```

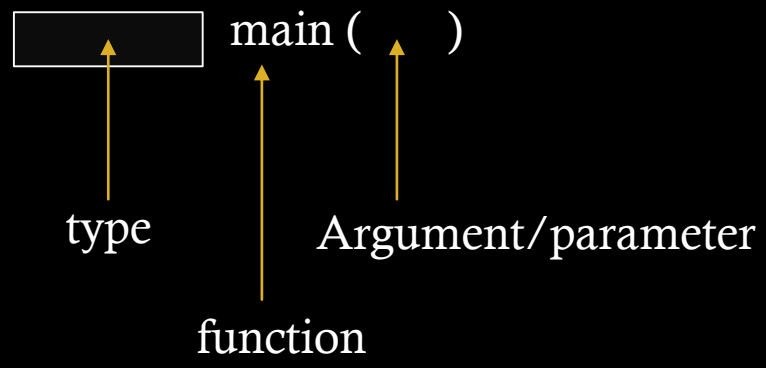
Here 201 is integer constant , 85.6 is float constant and A is charater constant

# Structure of c :-



# Hello KKS


preprocessor
#include<stdio.h>
Header file
function return type
int main( )
Main function
Body open
{
Predefined function
Standard output function
printf("Hello KKS");
return 0;
Body close
}
Semicolomm
return type


main ( )
type
function
Argument/parameter

## stdio

1.It stands for standard input output.
2.It is a collection of predefined functions/methods.
3.It is also called library of c.

## include

To include the header file into the program.

## #

1.It is called preprocessor.
2.It includes the library of c into the program before the execution of program.

## main

1.It is the function which is called the entry point of any program.
2.The execution of any prgram starts from the main function.
3.If in a program there is only one function then it should be main function.

## printf

1.It is a predefined function which is use used to print information or data on to the output screen.
2.It is defined in the stdio.h header file.

## void

1.It is a keyword.
2.It indicates that no one value is being returned by the function.
3.If we use any other keyword like int,float,char etc in place of void then we will use return keyword

# Syntax :-

```
printf(“  “);
```

## Example

Print Hello world

**Comment line :-** any text ignored by the compiler

# Escape sequence :-

| Escape Sequence | Meaning |
|---|---|
| \n | New Line |
| \t | Horizontal Tab |
| \b | BackSpace |
| \r | Carriage Return |
| \a | Audible bell |
| \' | Printing single quotation |
| \" | printing double quotation |
| \? | Question Mark Sequence |
| \\ | Back Slash |
| \f | Form Feed |
| \v | Vertical Tab |
| \0 | Null Value |
| \nnn | Print octal value |
| \xhh | Print Hexadecimal value |

printf(“ kuchh bhi ? “);

# Examples on basics like print value, comment, format specifier, escape sequence, etc


Q. Print

```
* * * * *
*       *
*       *
*       *
* * * * *
```

```
* * * * *
*
*
*
* * * * *
        *
        *
        *
* * * * *
```

**Q. Print**

1) %
2) \
3) “ ”
4) “Hii”
5) “Hii Hello”_%_ “ ”_\

Go to code editor software

**Format specifier :-** used to take input and print the output of a type


| Format Specifier | Type |
|---|---|
| %c | Used to print a character |
| %d | Used to print the signed integer |
| %f | Used to print the float values |
| %i | Used to print the unsigned integer |
| %l | Used to print the long integer |
| %lf | Used to print the double values |
| %lu | Used to print the unsigned integer or unsigned long integer |
| %s | Used to print the string |
| %u | Used to print the unsigned integer |

# Standard input output function


| | | |
|---|---|---|
| printf ( ) | gets( ) | getchar( ) |
| scanf( ) | puts( ) | putchar( ) |

# printf and scanf in c language :-


- There are many predefined function in the library of c
- Printf and scanf are the predefined function
- They are commonly used for input and output
- Printf and scanf both are predefined in **stdio.h** header file


# Printf function :-

- It is a predefined function
- It is used to print information or data on to the output screen
- It is predefined in the stdio.h header file
- Printf is case sensitive means Printf is wrong so it must be in lower case

# Printf syntax :-

Printf (“ format_specifier ”, variable_name);

# scanf() function in c :-

- It is a predefined function
- It is used to take user input at the time of execution of program
- It is also predefined in the stdio.h header file
- scanf is case sensitive means Scanf is wrong so it must be in lower case

## Syntax :-

scanf ( " format specifier ", address of variable);

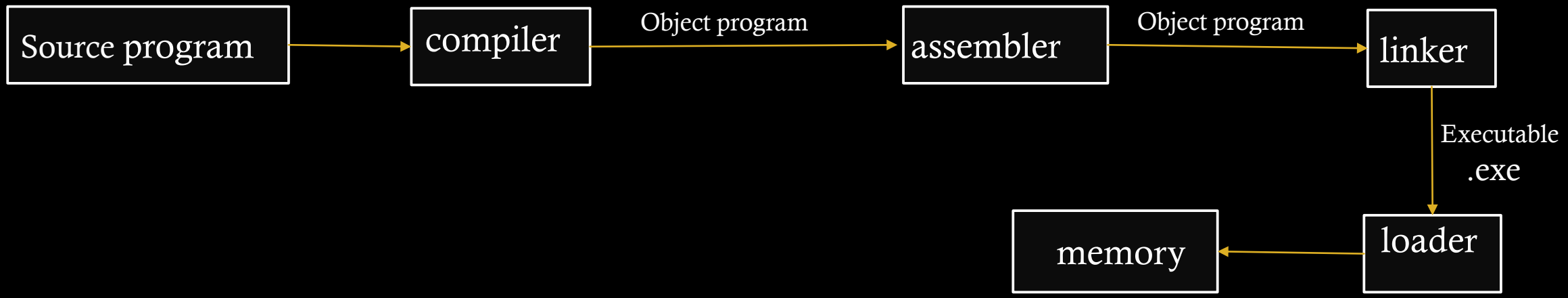
# Examples :-

printf and scanf with integer

printf and scanf with float

Printf and scanf with character

# Execution of program



# Type conversion in c language

- The type conversion process in c is basically converting one type of data type to other to perform some operation . The conversion is done only between those datatype where in the conversion is possible.

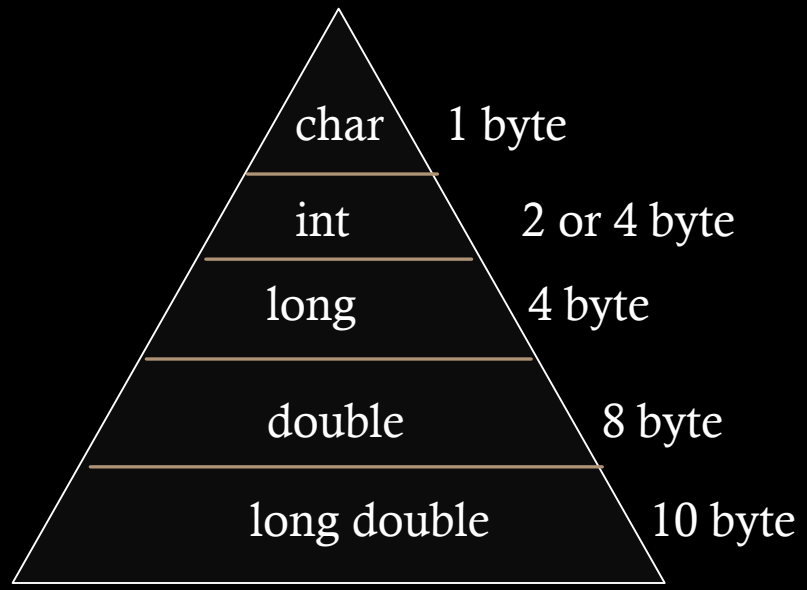
example - char to int and vice versa.

- There are two types of type conversion .

1. Implicit type conversion
2. Explicit type conversion

# Implicit type conversion

- This type of conversion is usually performed by the compiler when necessary any commands by the user thus it is called “automatic type conversion”.
- All the data of the variable are upgraded to the data type of the variable with largest data type




eg .

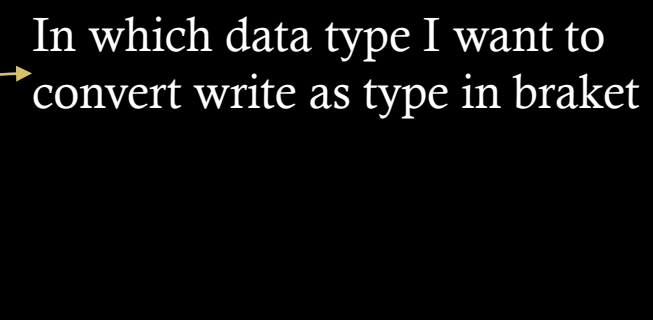
```
#include<stdio.h>
int main()
{
int x=10;
char y=‘a’;
x=x+y;
float z=x+1.0;
printf(“x=%d,z=%f”,x,z);
return 0;
}
```

# Explicit type conversion

- The type conversion performed by programmer by posing the data type of the expression of specific type is known as explicit type conversion
- The explicit type conversion is also know as type casting

eg .

```
#include<stdio.h>
int main()
{
double x= 1.2;
int sum= (int) x+1;
printf("sum=%d",sum);
return 0;
}
```



# ASCII

0 → 48
A → 65
a → 97

## ASCII Table

| Dec | Hex | Oct | Char | Dec | Hex | Oct | Char | Dec | Hex | Oct | Char | Dec | Hex | Oct | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 32 | 20 | 40 | [space] | 64 | 40 | 100 | @ | 96 | 60 | 140 | ` |
| 1 | 1 | 1 | | 33 | 21 | 41 | ! | 65 | 41 | 101 | A | 97 | 61 | 141 | a |
| 2 | 2 | 2 | | 34 | 22 | 42 | " | 66 | 42 | 102 | B | 98 | 62 | 142 | b |
| 3 | 3 | 3 | | 35 | 23 | 43 | # | 67 | 43 | 103 | C | 99 | 63 | 143 | c |
| 4 | 4 | 4 | | 36 | 24 | 44 | $ | 68 | 44 | 104 | D | 100 | 64 | 144 | d |
| 5 | 5 | 5 | | 37 | 25 | 45 | % | 69 | 45 | 105 | E | 101 | 65 | 145 | e |
| 6 | 6 | 6 | | 38 | 26 | 46 | & | 70 | 46 | 106 | F | 102 | 66 | 146 | f |
| 7 | 7 | 7 | | 39 | 27 | 47 | ' | 71 | 47 | 107 | G | 103 | 67 | 147 | g |
| 8 | 8 | 10 | | 40 | 28 | 50 | ( | 72 | 48 | 110 | H | 104 | 68 | 150 | h |
| 9 | 9 | 11 | | 41 | 29 | 51 | ) | 73 | 49 | 111 | I | 105 | 69 | 151 | i |
| 10 | A | 12 | | 42 | 2A | 52 | * | 74 | 4A | 112 | J | 106 | 6A | 152 | j |
| 11 | B | 13 | | 43 | 2B | 53 | + | 75 | 4B | 113 | K | 107 | 6B | 153 | k |
| 12 | C | 14 | | 44 | 2C | 54 | , | 76 | 4C | 114 | L | 108 | 6C | 154 | l |
| 13 | D | 15 | | 45 | 2D | 55 | - | 77 | 4D | 115 | M | 109 | 6D | 155 | m |
| 14 | E | 16 | | 46 | 2E | 56 | . | 78 | 4E | 116 | N | 110 | 6E | 156 | n |
| 15 | F | 17 | | 47 | 2F | 57 | / | 79 | 4F | 117 | O | 111 | 6F | 157 | o |
| 16 | 10 | 20 | | 48 | 30 | 60 | 0 | 80 | 50 | 120 | P | 112 | 70 | 160 | p |
| 17 | 11 | 21 | | 49 | 31 | 61 | 1 | 81 | 51 | 121 | Q | 113 | 71 | 161 | q |
| 18 | 12 | 22 | | 50 | 32 | 62 | 2 | 82 | 52 | 122 | R | 114 | 72 | 162 | r |
| 19 | 13 | 23 | | 51 | 33 | 63 | 3 | 83 | 53 | 123 | S | 115 | 73 | 163 | s |
| 20 | 14 | 24 | | 52 | 34 | 64 | 4 | 84 | 54 | 124 | T | 116 | 74 | 164 | t |
| 21 | 15 | 25 | | 53 | 35 | 65 | 5 | 85 | 55 | 125 | U | 117 | 75 | 165 | u |
| 22 | 16 | 26 | | 54 | 36 | 66 | 6 | 86 | 56 | 126 | V | 118 | 76 | 166 | v |
| 23 | 17 | 27 | | 55 | 37 | 67 | 7 | 87 | 57 | 127 | W | 119 | 77 | 167 | w |
| 24 | 18 | 30 | | 56 | 38 | 70 | 8 | 88 | 58 | 130 | X | 120 | 78 | 170 | x |
| 25 | 19 | 31 | | 57 | 39 | 71 | 9 | 89 | 59 | 131 | Y | 121 | 79 | 171 | y |
| 26 | 1A | 32 | | 58 | 3A | 72 | : | 90 | 5A | 132 | Z | 122 | 7A | 172 | z |
| 27 | 1B | 33 | | 59 | 3B | 73 | ; | 91 | 5B | 133 | [ | 123 | 7B | 173 | { |
| 28 | 1C | 34 | | 60 | 3C | 74 | < | 92 | 5C | 134 | \ | 124 | 7C | 174 | \| |
| 29 | 1D | 35 | | 61 | 3D | 75 | = | 93 | 5D | 135 | ] | 125 | 7D | 175 | } |
| 30 | 1E | 36 | | 62 | 3E | 76 | > | 94 | 5E | 136 | ^ | 126 | 7E | 176 | ~ |
| 31 | 1F | 37 | | 63 | 3F | 77 | ? | 95 | 5F | 137 | _ | 127 | 7F | 177 | |

# Storage class :-


- The storage class determine the part of the memory where the variable would be stored
- The storage class also determine the initial value of the variable
- And it used to define the scope and lifetime of variable
- There are two storage location in computer CPU register and memory

## There are four types of storage class in C

1. Auto
2. Extern
3. Static
4. Register

## Auto :-

| | |
|---|---|
| Keywords | : auto |
| Storage | : memory |
| Default initial value | : garbage value |
| Scope | : local to the block in which the variable is defined |
| Life | : till the control remains within the block in which the variable is defined |

## Static :-

Keywords : static
Storage : memory
Default initial value : 0
Scope : local to the block in which the variable is defined
Life : value of the variable persist between different function calls

## Extern :-

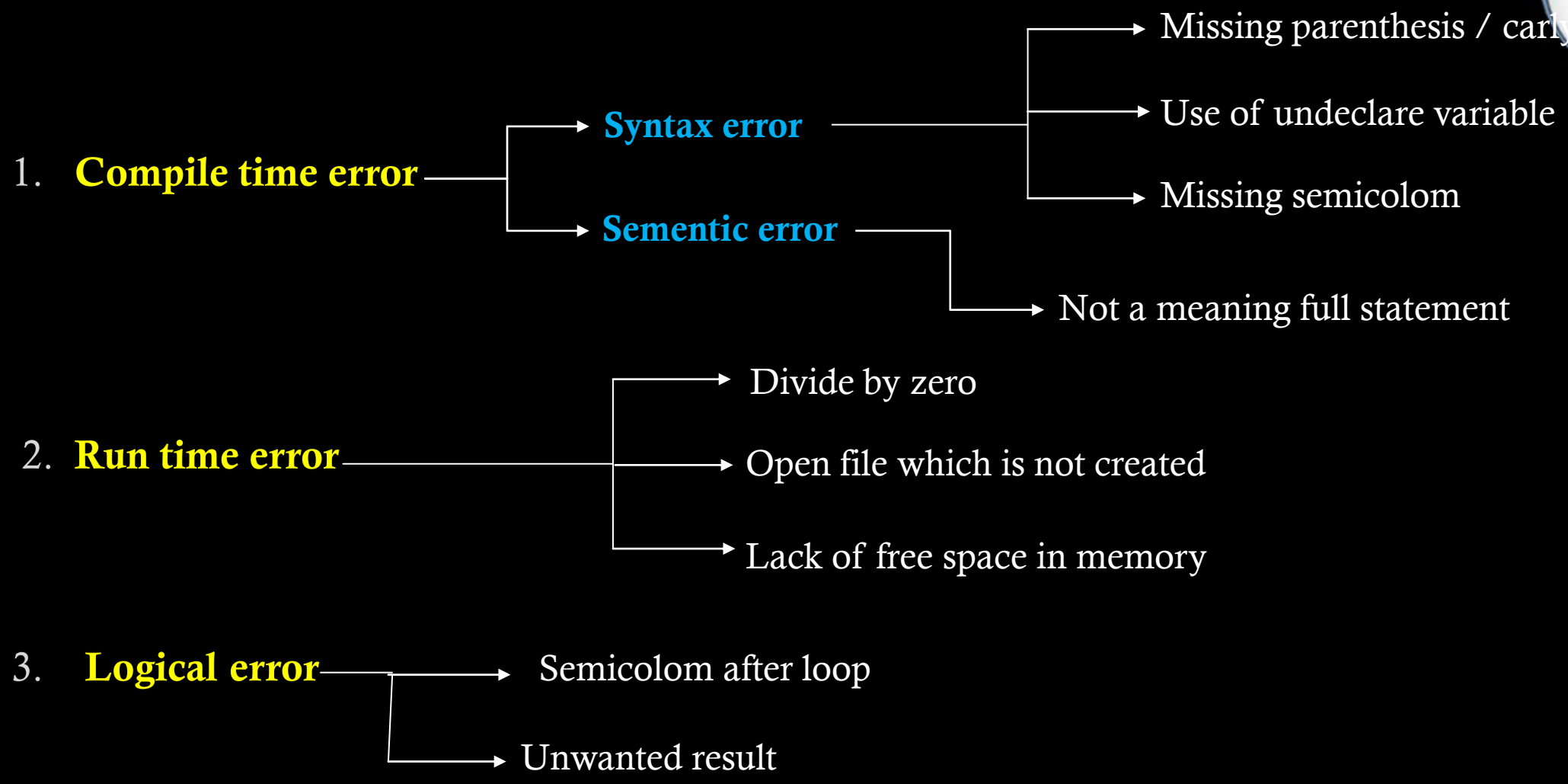
Keywords : extern
Storage : memory
Default initial value : 0
Scope : global
Life : as long as the program's doesn't come to an end

## Register :-

Keywords : register
Storage : CPU register
Default initial value : garbage value
Scope : local to the block in which the variable is defined
Life : till the control remains within the block in which the variable is defined


| Storage Specifier | Storage | Initial value | Scope | Life |
|---|---|---|---|---|
| Auto | Stack | Garbage | Within block | End of block |
| Extern | Data segment | Zero | Global multiple files | Till end of program |
| Static | Data segment | Zero | Within block | Till end of program |
| Register | CPU register | Garbage | Within block | End of block |

# Error in c program :-




# Thank You !!

# Dhanybad !!

# Shukriya !!