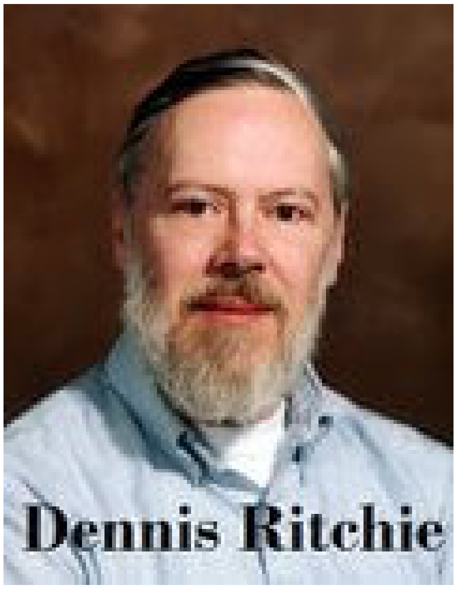
C Language

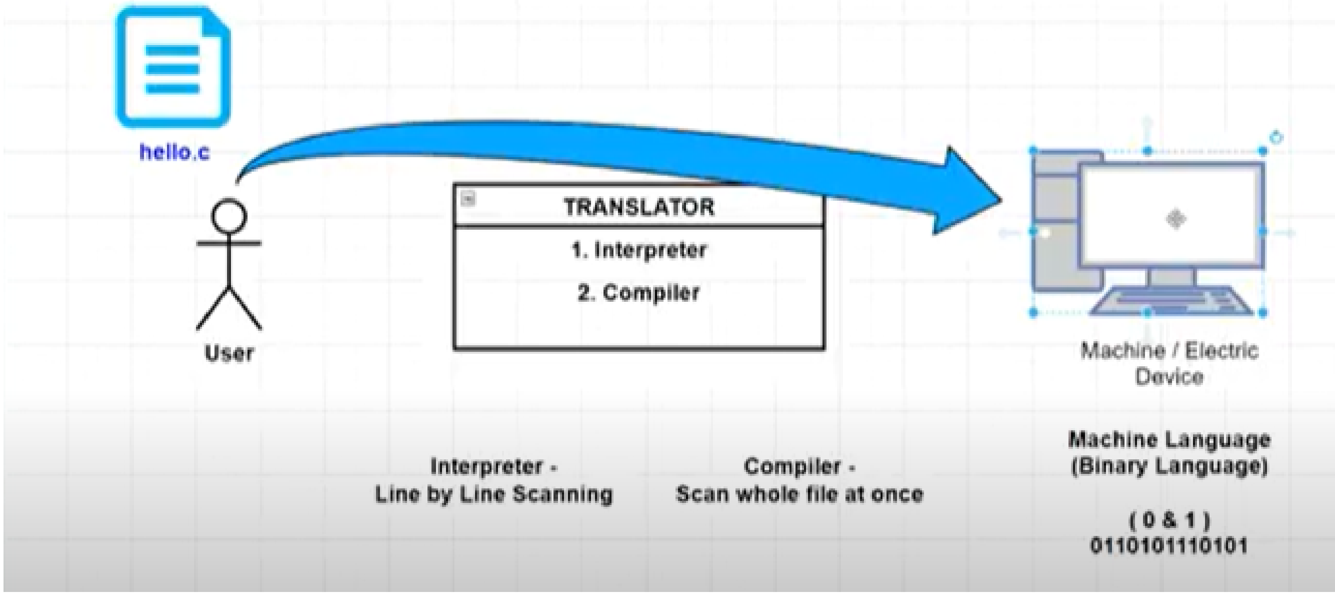


History of C Language

- **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**.
- **Dennis Ritchie** is known as the **Father of the c language**.
- **C language** is known as the **Mother of all programming Languages**.
- It was developed to overcome the problems of previous languages such as B, BCPL,Algol 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



Interpreter - Line by Line Scanning

Compiler - Scan whole file at once

Machine Language (Binary Language)

(0 & 1) 0110101110101

Program :

```
{
    Space For Codding
}
```

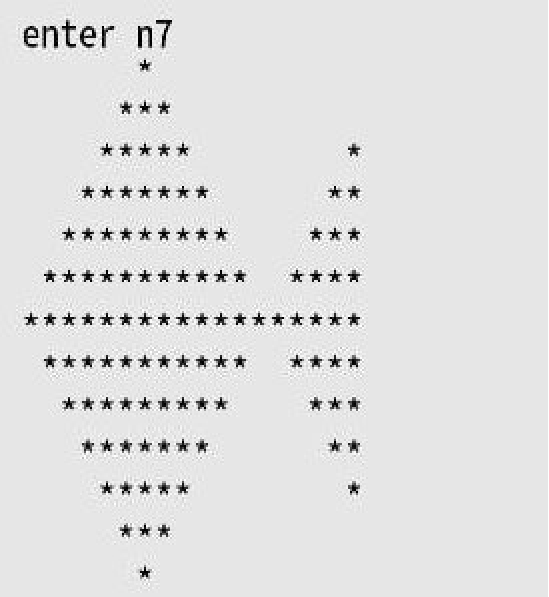
Space name : Main()

Main use for space name

() ⇒ Funcation / Parentheses

Print Massage :

Printf(“Hello World…!”);



```
enter n7
      *
     ***
    *****       *
   *******     **
  *********   ***
 *********** ****
*****************
 *********** ****
  *********   ***
   *******     **
    *****       *
     ***
      *
```

Dictionary :

stdio.h ⇒ standard Input Output header file

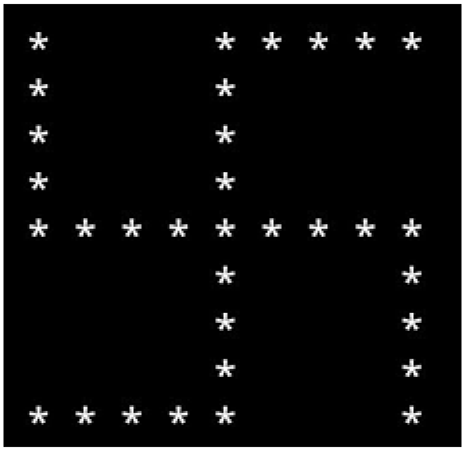
#include<stdio.h>

# ⇒ Preprocessor

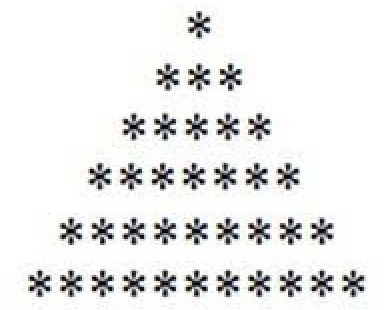
Escape Sequence Charaters :

1. /n ⇒ New Line
2. /t ⇒ Teb (4 Whitespace)

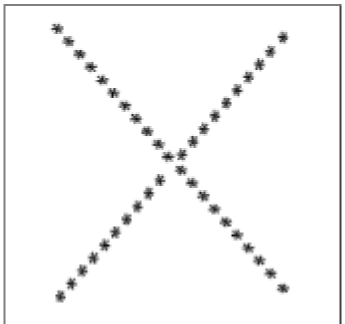
Work :



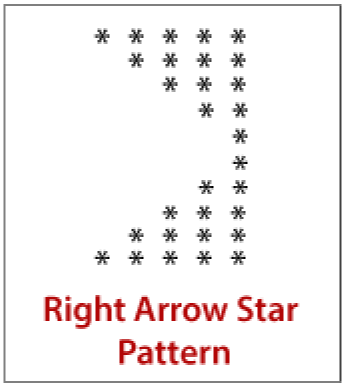
```
*       * * * * *
*       *
*       *
*       *
* * * * * * * * *
        *       *
        *       *
        *       *
* * * * *       *
```



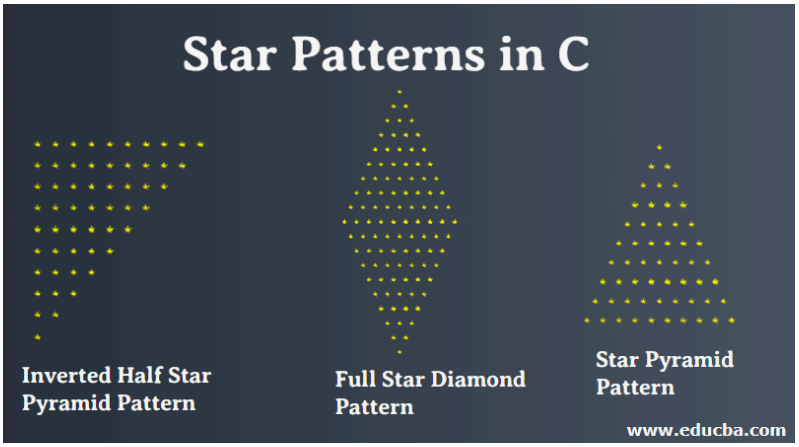
```
     *
    ***
   *****
  *******
 *********
***********
```

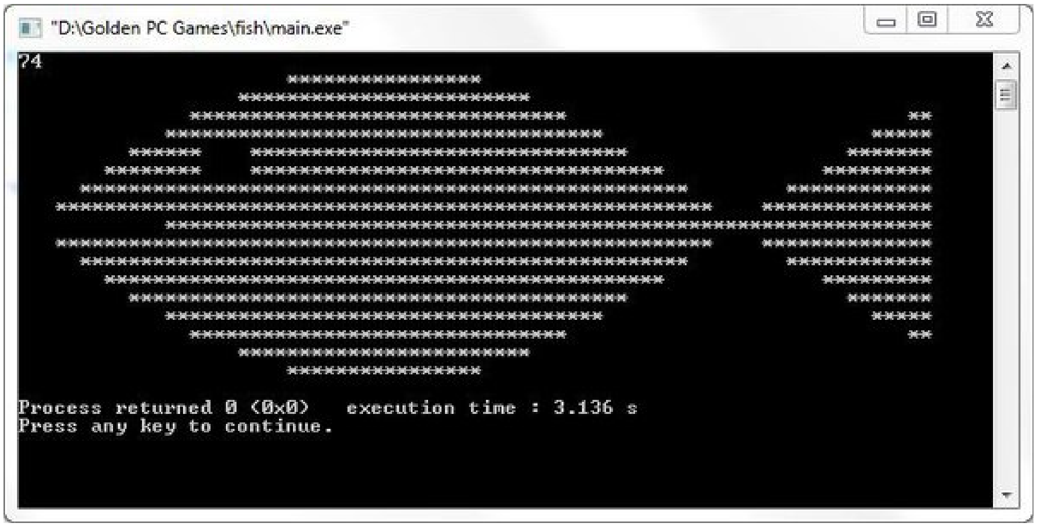


X Star Pattern



Right Arrow Star Pattern



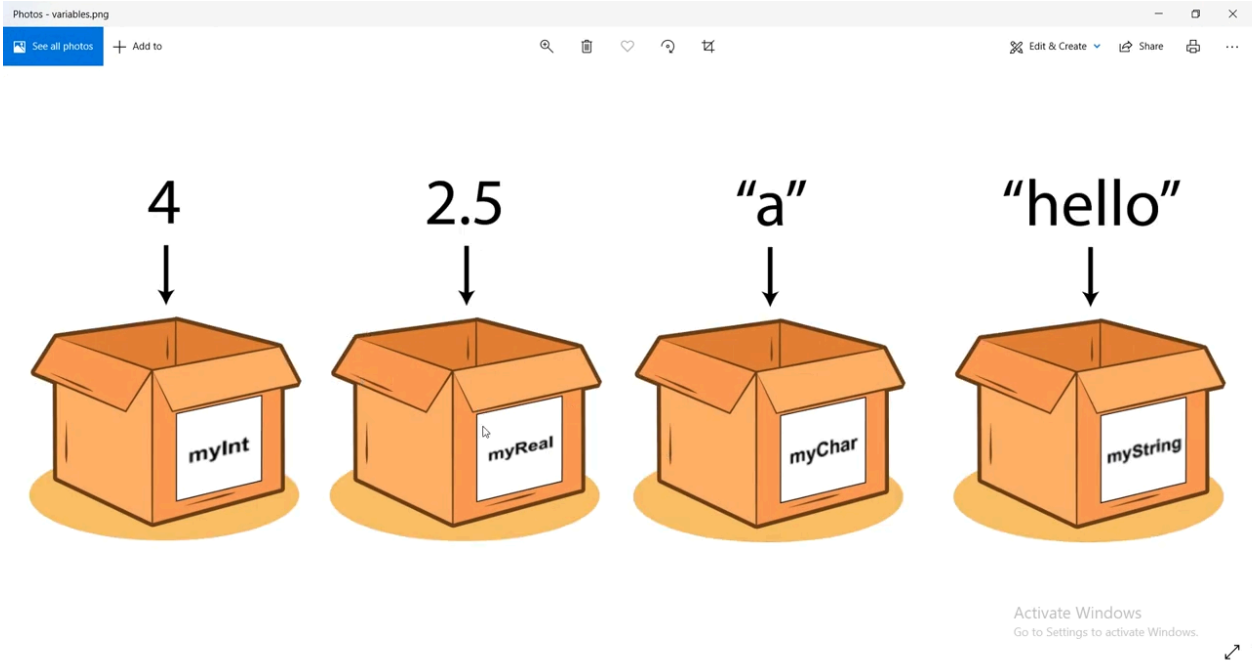


```
74
                      ****************
                  ************************
              *******************************                            **
            ************************************                      *****
         ******    *******************************                  *******
       ********    **********************************             *********
     **************************************************        ************
   ******************************************************    **************
            ***************************************************************
   ******************************************************    **************
     **************************************************        ************
       **********************************************             *********
         *****************************************                  *******
            ************************************                      *****
              *******************************                            **
                  ************************
                      ****************

Process returned 0 (0x0)   execution time : 3.136 s
Press any key to continue.
```

- Print Your Name Using Printf

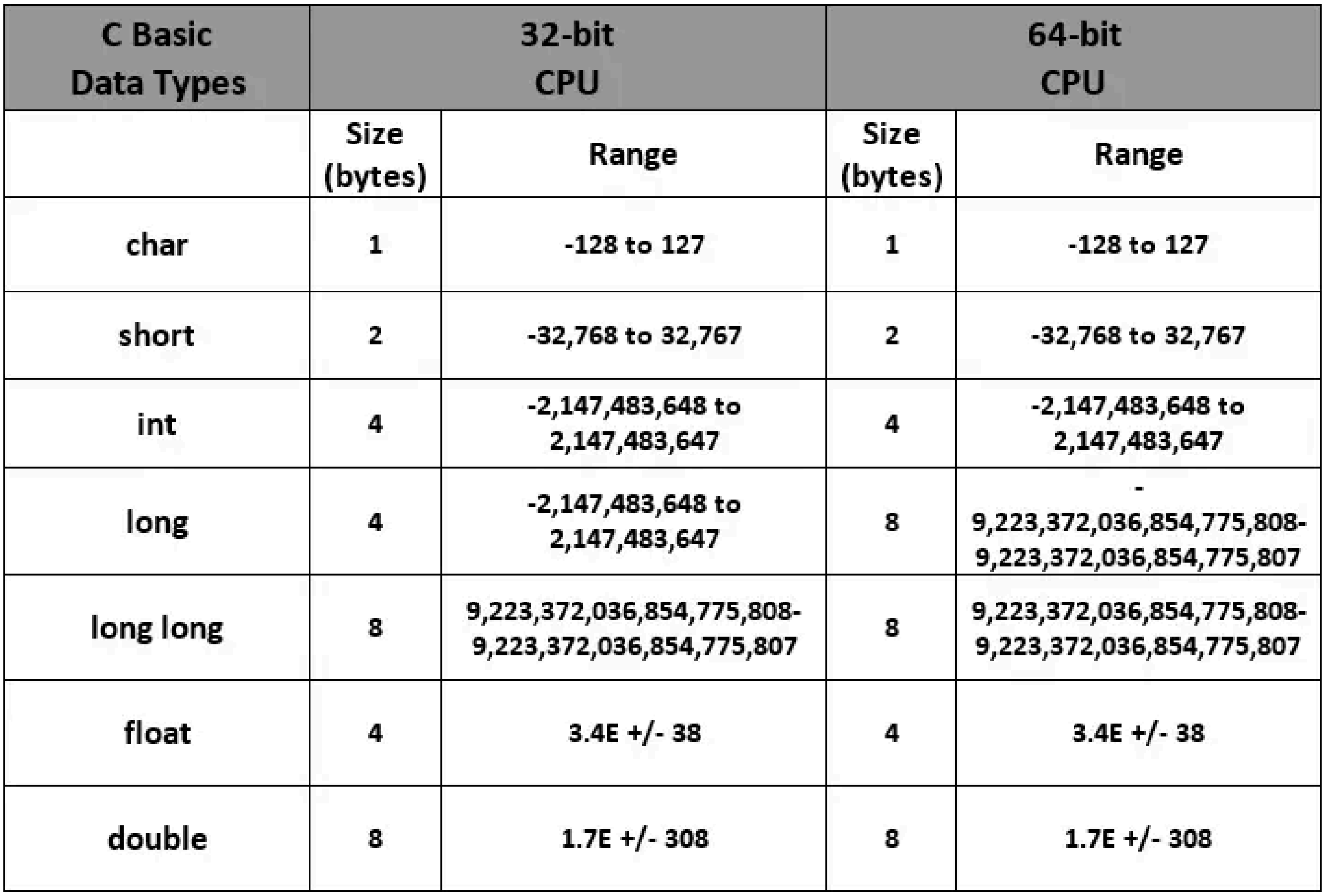
Lecture 2:

1. Comments :
    - // - Single Line Comment
    - /* */ - Multi Line Comment
2. Datatype, Variables & Format Specifiers
    - Variable ⇒ One type of Container which store our data



- 1 to .... ⇒ Integer data = int
- 0.1 to .... ⇒ Float data = float
- A = 20; b = 30; ⇒ = means Assignment / Initialization
- Variable Name : first / a / _first ⇒ Right
- 7first ⇒ Wrong
- Printf(“   ” ,);

- Data Type ⇒ Type of Specific Data
- Fomate Specifier ⇒
- int ⇒ %d
- Float ⇒ %f



C Basic Data Types	32-bit CPU		64-bit CPU	
	Size (bytes)	Range	Size (bytes)	Range
char	1	-128 to 127	1	-128 to 127
short	2	-32,768 to 32,767	2	-32,768 to 32,767
int	4	-2,147,483,648 to 2,147,483,647	4	-2,147,483,648 to 2,147,483,647
long	4	-2,147,483,648 to 2,147,483,647	8	-9,223,372,036,854,775,808- 9,223,372,036,854,775,807
long long	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807
float	4	3.4E +/- 38	4	3.4E +/- 38
double	8	1.7E +/- 308	8	1.7E +/- 308

Keyword?

- Keywords, also known as reserved words.
- there is a set of 32 keywords in C language

break	**case**	**char**	**const**	**Auto**	**short**	**struct**	**Switch**
double	**int**	**else**	**enum**	**float**	**continue**	**sizeof**	**Default**
extern	**for**	**do**	**goto**	**If**	**typedef**	**union**	**Void**
static	**signed**	**long**	**register**	**return**	**unsigned**	**volatile**	**while**

Operator :

Assignment Operator ⇒ =, +=, -=, *=, /=, %=

Arithmetic Operator ⇒ +, -, *, /, %

Relational / conditional Operator ⇒ <, >, <=, >=, !, !=

Logical Operators ⇒ &&, ||

Assignment Conditional Operator ⇒ ==(Comparision) / !=

Bitwise Operator ⇒ ~, ^

Constant Variable :

- A value / A variable which can not be Modified, after it is intialized

Tow method you are create a Constant variable

- “Const” Keyword ⇛ Const PI = 3.14
- Macro ⇛ #define a 10

The operators are types of symbols that inform a compiler for performing some specific logical or mathematical functions

1. Types of operators
2. Operator Precedence
3. Type Conversation

1. An operator in C can be defined as the symbol that helps us to perform some specific mathematical, relational, bitwise, conditional, or logical computations on values and variables.

   Assignment Operator ⇛ =, +=, -=, *=, /=, %=

   Arithmetic Operator ⇛ +, -, *, /, %, ++, --

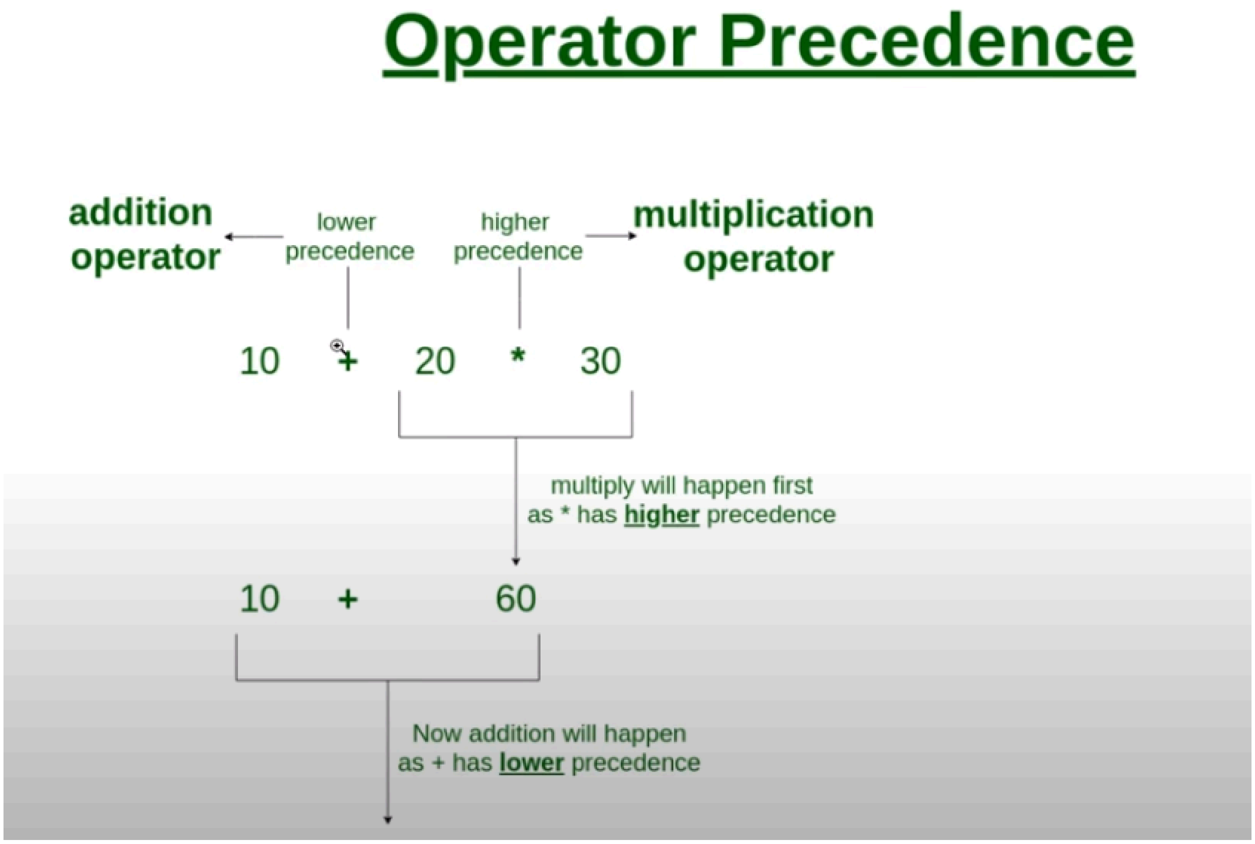
   Relational / conditional Operator ⇛ <, >, <=, >=,, !=

   Logical Operators ⇛ &&, ||

Assignment Conditional Operator ⇒ ==(Comparision) / !=

Bitwise Operator ⇒ ~, ^

2. Operator Precedence : - (/ , * , + , -)
    - () ⇒ prenthesis
    - / * ⇒ Left to right // Associativity
    - + - ⇒ Left to right // // Associativity
    - = ⇒ Last



3. Type Conversation :

Type conversion in C is the process of converting one data type to another.

- Printf(“ %f ” , (int)pi);
- C = (int)(a/b);
- C = (int)a / (int)b

❖ Control Structure : Decision Maker

➢ If Statement

➢ If ……. Else

➢ Leader if ……… else

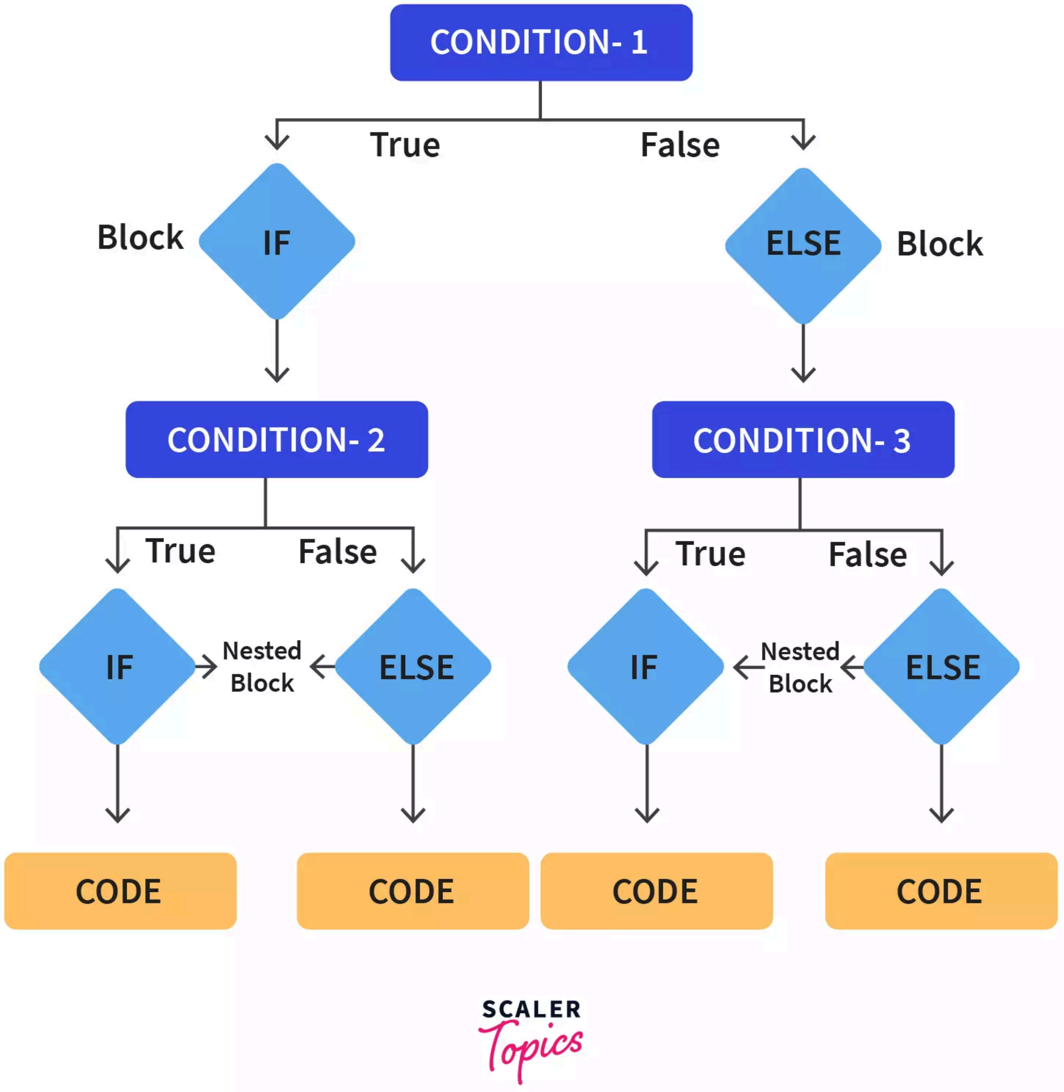
➢ Nested if ……… else

➢ Switch case

➢ Ternary operator

- If statement :

```
If (condition)

{
    Statement
}
A < b
15 < 10
```



Ternary Operator

(condition) ? (variable = Expression2) : (variable = Expression3);

? True

: False

Switch Case

```
switch(expression)

{

case value1: statement_1;

           break;

case value2: statement_2;

           break;

.

.

.

case value_n: statement_n;

            break;


default: default_statement;

}
```

Project :-

1. char s='a';

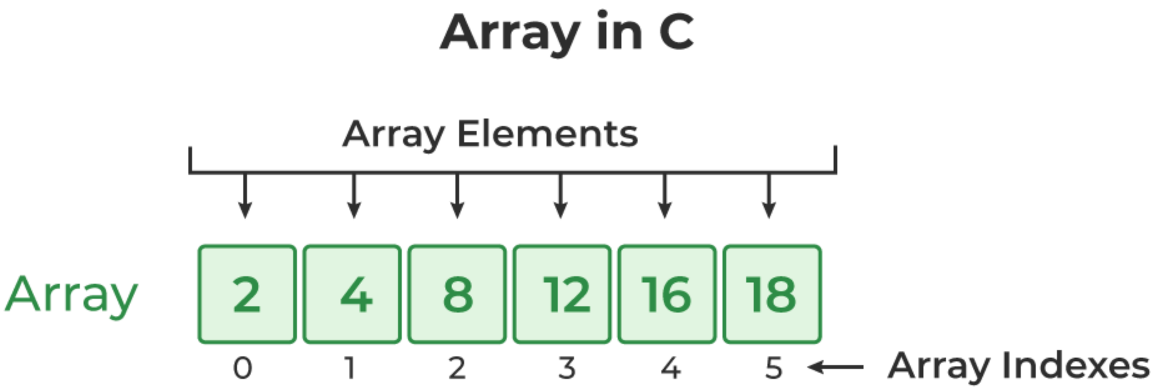
   ```
           do{
   ```

```
                         printf("%c,\t",s);

                         s+=4;

               }while(s<='z');

2.  int n, ans = 0;

         printf("Enter the number: ");

        scanf("%d", &n);

               if(n==0){

                         printf("Total number of digits: 1");

                 }

         else{

               while (n != 0) {


            n = n / 10;

            ans++;

         }

         printf("Total number of digits: %d\n", ans);

               }

3.  int a,b,first=0,last=0,result=0;

               printf("Enter any value of number: ");

               scanf("%d",&a);
```

```
                b=a;

                last=a%10;

        while(b>0)

        {

        first=b%10;

        b=b/10;

        }

result=first+last;

printf("Sum of %d and %d is %d\n",first,last,result);
```

What is Array in C?

An array in C is a fixed-size collection of similar data items stored in contiguous memory locations. It can be used to store the collection of primitive data types such as int, char, float, etc., and also derived and user-defined data types such as pointers, structures, etc



```
data_type array_name [size];



        int a;

                printf("Enter your value");

                scanf("%d",&a);

                int arr[a];

                int i;

                for(i=0; i<5; i++){

                        printf("arr[%d]",i);

                        scanf("%d", &arr[i]);

                }
```