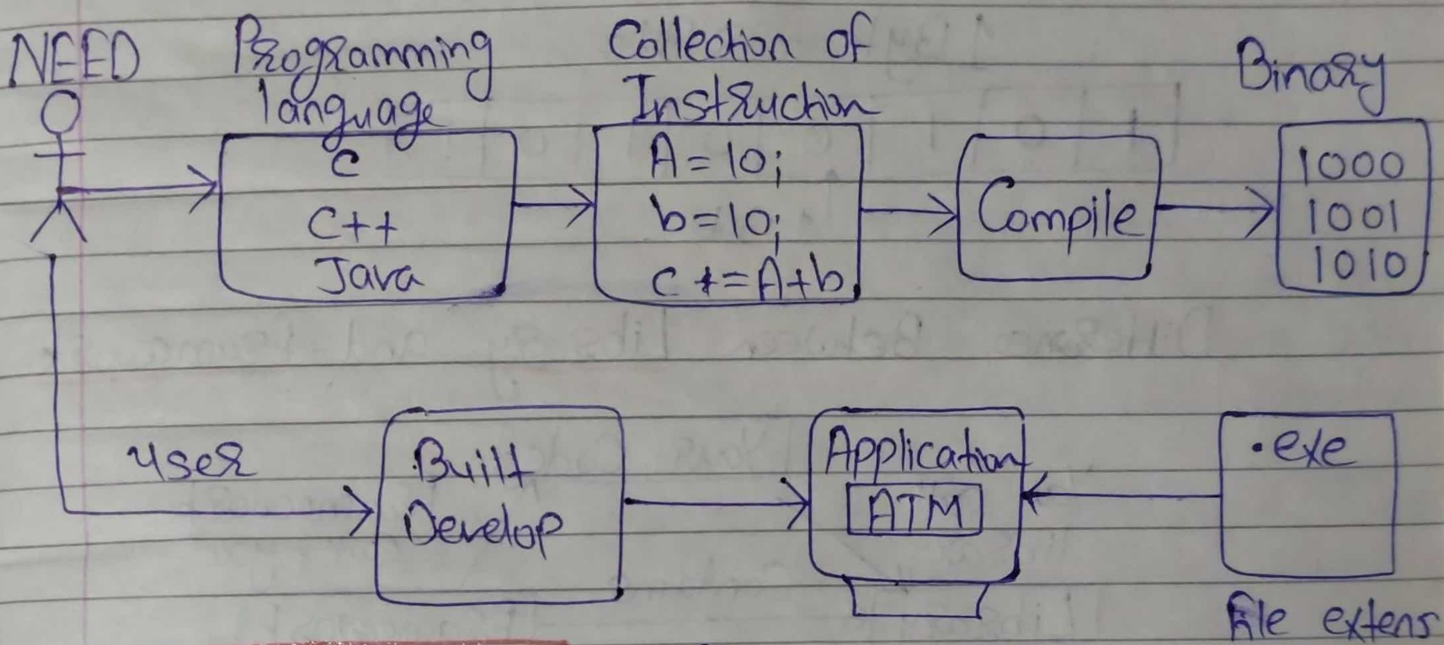
① what is Computer or Computer system?

→ A Combination of memory, cPU, Peripheral devices that are Connected to it, and os (operating system).

→ Developed by "charles Babbage"

→ A device for working with information.



Computer Language - A formal language used to Communicate with a Computer.

- Standalone - Request mandatory Installation
  - VLc
  - depend upon System Depation path
  - c, c++, Java, Python, .Net
- Web application:- No need for Installation depend on operating system.

1 byte = 8 bits
1 kilobyte = 1024 bytes
1 megabyte = 1024 kilobytes
1 Gigabyte = 1024 megabyte
1 terabyte = 1024 Gigabyte

Computer Bit
● - ON   O - off

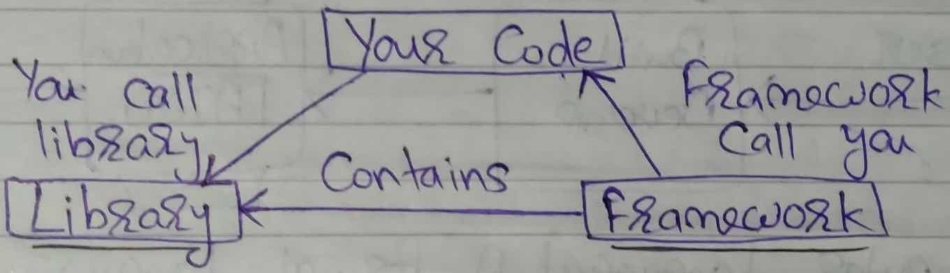
Computer Byte
O O ● ● O ● O ●
0 0 1 1 0 1 0 1

1 Byte = 8 Bits

1	0	1	0	0	1	0	1

↑ 1 Bit

Difference Between Library and framework.



Library - It Consist of set of functions.
→ Collection of reusable functions used by Computer Program.
→ ex:- React js

Framework - It Consist of set of library
→ Set of rules
→ ex:- Angular js.
→ A framework calls on your Code. Your Code calls on a library.

Stdio.h
Printf()
Scanf()

Conio.h
clr()
getch()

String
strlen()
strrev()

math.h
Pow()
rand()

doc.h
getdata()
gettime()

Graphics
setcolor()
setbkcolor()

Frontend	Backend
→ Markup & web lang. Such as HTML, CSS.	→ Programming and Scriptin such as Python, Ruby
→ Apperence for layout (fast changing techno)	→ Database & data administra (Data transformation)

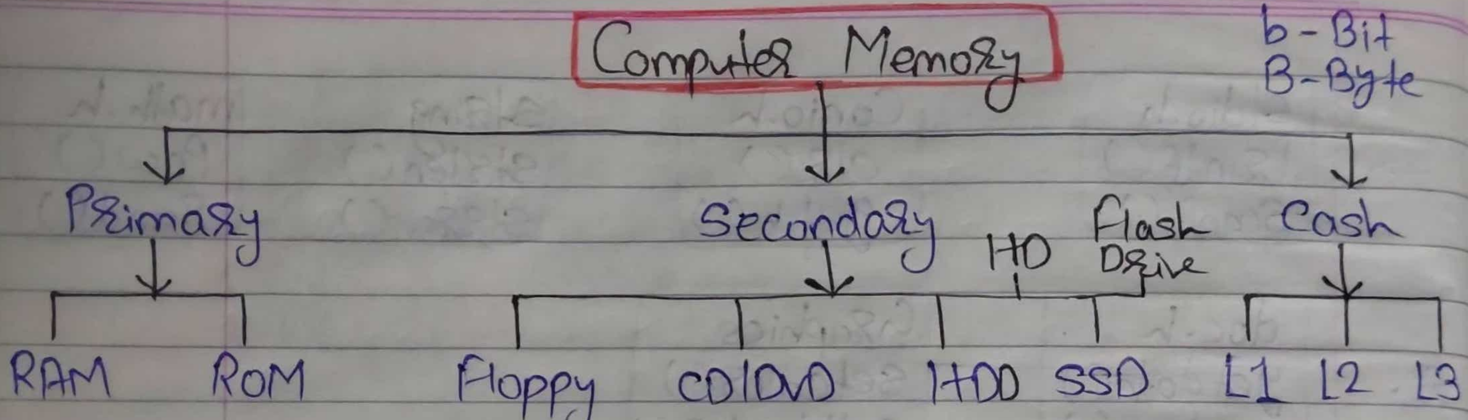
U ~~UI~~:- Design from user's Point of view (user expectatio
→ Refers to the screens, buttons, toggles, icons and other visual elements.

UX:- Design from user's Point of view
-(user expectation) - Interaction designer
→ Refers to the entire interaction with a Product.

UI:- User Interface
→ Refers to the screens, buttons, toggles, icons. and other visual elements.

Thread:- A subset of Process.

Process:- Sequence of Instruction.



(Place where the data is stored) **Memory** :- Process of tacking in info from the World around ~~us~~, Processing it, storing and recalling info. 03 - Bit

→ Measured in terms of Bit/Byte

Computer Memory :- Used to store data or Program on temporary or Permanent basis use in electronic digital Computer.

RAM	ROM
- Random Access memory	Read only memory
- Volatile	- Non-volatile
- Temporary storage	- Permanent storage
① static ② Dynamic	① PROM ② EPROM ③ EEPROM

(Diskette) **Floppy Disk** :- A removable magnetic storage medium allow recording of Data

→ Distribute software, Transfer Data, create backups.

CD :- Compact Disk.

→ A Portable storage medium can be used to record, store and Play audio, video

DVD :- Digital Versatile Disk / Digital video disc.

→ Provided a storage Capacity

→ Allow storage of large amount of data using digital technology

HDD :- Hard Disk Drive

→ A Data storage device that lives inside the Comp.

→ is a Non-volatile data storage device.

SSD :- Solid - state Drive

→ A new generation of Storage device used in Computers.

→ Store data using flash - based memory, which is much faster than the HD. (more immediate data transfer)

Cache :- (Some like Buffer)

→ The temporary memory officially termed "CPU

→ L1 :- (Level 1) / Primary cache

- Stored Include CPU inside
- Extremely fast but relatively small.

→ L2 :-(Level 2) / Secondary cache.

- May be stored Inside or outside the CPU.
- Single core and shared core

→ L3 :-(Level 3) / Specialized memory

- stored outside CPU
- shared core.

Stack memory :- Used as temporary data storage that behaves as a first-In-First-out.

- Stack always in RAM.

Static memory:- Allocation of memory Performs at the Compile time.

→ A form of flip-flop holds each bit of memory

Dynamic Memory:- Allocation is done at the execution or run time.

ALU:- Arithmetic logic Unit
CPU:- Central processing Unit
DRAM:- Dynamic Random Accesr Memory
SRAM:- Static Random Accesr Memory
RAM :- Random Accesr Memory
PROM:- Programmable Read-only memory
EPROM :- Erasable Programmable Read-only memory
EEPROM :- Electrically Erasable programmable ROM

Variables:- Contain for storing data values
- Variable is name of memory locations

Function:- Block of Code that Performs Particular task

Datatype:- Specific size & type of value that can be stored in variable.

① Primitive(Primary)	② Derived	③ user defined (Non-Prim)
int	Array	Class
Float	String	structure
char	Pointer	Union
void	Function	enum
Boolean	Reference	Typedef

int
- Short
  - Signed } 2 Byte
  - unsigned
- int
  - Signed } 2 or 4 Byte
  - unsigned
- long
  - Signed } 4 Byte
  - unsigned

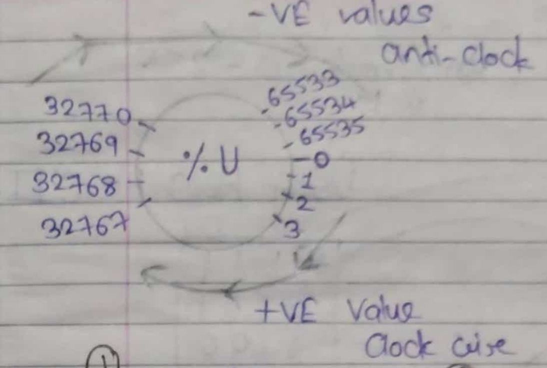
Signed:- +Ve / -Ve
Unsigned:- +Ve

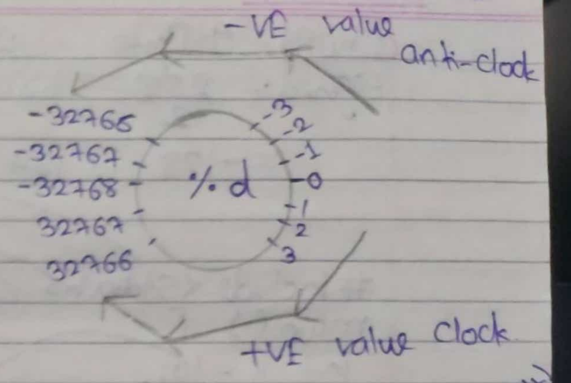
Tinyint	-	~~8~~ 1 Byte
Smallint	-	2 Byte
int	-	4 Byte
bigint	-	8 Byte

Char
- Signed } 1 Bit
- unsignd

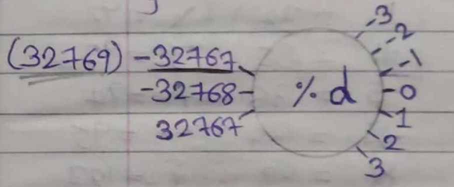
Float
- Float
- Double
- long Double

Data:- Collection of related facts & figure is called Data
→ Item of Information / Collection of information.

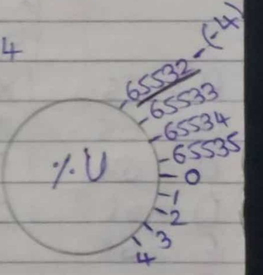




ex:- ①
```
Main()
{ short x=32769;
  clrscr();
  printf("%d",x);
}
```



② unsigned short x=-4
printf("%u",x);
O/P:- 65532



③ Unsigned short
x=65538

O/P:- 2

Second Method:- 65538
- 65535
0000③ (0,1,2)

④ Unsigned short x=65538
printf("%d",x);
O/P:- 2

⑤ short x=32768
printf("%d",x);
O/P:- -32768

ASCII :- American Standard Code for Information Interchange

Char :- 1 Byte $\quad 2^8 = 256$

Signed :- $\frac{256}{2}$ $\quad$ Unsigned :- 0 - 255

= -128 to 127

Using ANS ASCII

```
#include <stdio.h>
main () {
 int x = 10;
 Printf ("%d", x);
```

```
1111
0101
1100
1010
```

A = 65
B = 66
C = 67
D = 68
E = 69
F = 70
G = 71
H = 72
I = 73
J = 74
K = 75
L = 76
M = 77
N = 78
O = 79
P = 80
Q = 81
R = 82
S = 83
T = 84
U = 85
V = 86

~~X~~W = 87
X = 88
Y = 89
Z = 90

a = 97
b = 98
c = 99
d = 100
e = 101
f = 102
g = 103
h = 104
i = 105
j = 106
k = 107
l = 108
m = 109
n = 110
o = 111
p = 112
q = 113

r = 114
s = 115
t = 116
u = 117
v = 118
w = 119
x = 120
y = 121
z = 122

0 = 48
1 = 49
2 = 50
3 = 51
4 = 52
5 = 53
6 = 54
7 = 55
8 = 56
9 = 57

Space = 32
! = 33
" = 34
= 35
$ = 36
% = 37
& = 38
' = 39
(= 40
) = 41
* = 42
+ = 43
, = 44
- = 45
. = 46
/ = 47
: = 58
; = 59
< = 60
= = 61
> = 62

? = 63
@ = 64
[= 91
\ = 92
] = 93
^ = 94
_ = 95
{ = 123
| = 124
} = 125
~ = 126

26 (upper) + 26 (lower) + 10 (Number) + 150 (symbol)
= 26+26+10+150 < 256 → 2^8

ex:-
```
#include <stdio.h>
#include <conio.h>
main () {
 int char c;
 Printf ("enter the ");
 Scanf ("%c", &c);
 Printf ("%d", c);
```

O/P :- C $\boxed{\#}$
35

Operators - Required to Perform an operation on varia.

① Unary (modification) :-
++ (Increment) $\qquad$ -- (Decrement)

② Binary :-
- Arithmetic :- +, -, *, /, %
- logical :- &&, ||, !
- Bitwise :- &, |, <<, >>, ~, ^
- Assignment :- =, +=, -=, *=, /=, %=

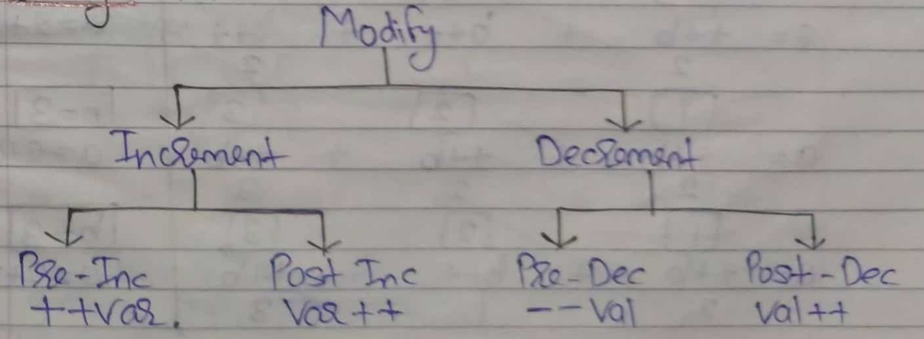
③ Ternary (conditional)
- ?:

Prev
Inc/Dec
User val

Post
Inc/Dec

Post
User val
Inc/Dec

① **Unary** - ++ --



①

a=0
b=0

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}}$ | $\boxed{b=2}$ $a=0+1=\underline{1}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}}$ | $\boxed{b=2}$ $a=2+2=\underline{4}$

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}}$ | $\boxed{b=2}$ $a=0+2=\underline{2}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}}$ | $\boxed{b=2}$ $a=2+1=\underline{3}$

②

a=0
b=0

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} +- \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{3}}{\underset{3}{{+}{+}b}}$ | $\boxed{b=3}$ $a=1+3=\underline{4}$

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}} + \overset{\boxed{3}}{\underset{2}{b{+}{+}}}$ | $\boxed{b=3}$ $a=2+2=\underline{4}$

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}} + \overset{\boxed{3}}{\underset{3}{{+}{+}b}}$ | $\boxed{b=3}$ $a=2+3=\underline{5}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{3}}{\underset{2}{b{+}{+}}}$ | $\boxed{b=3}$ $a=2+1+2=\underline{5}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}} + \overset{\boxed{3}}{\underset{3}{{+}{+}b}}$ | $\boxed{b=3}$ $a=2+2+3=\underline{7}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{3}}{\underset{3}{{+}{+}b}}$ | $\boxed{b=3}$ $a=2+1+3=\underline{5}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{2}{{+}{+}b}} + \overset{\boxed{3}}{\underset{2}{b{+}{+}}}$ | $\boxed{b=3}$ $a=2+2+2=\underline{6}$

$a = \overset{\boxed{1}}{\underset{0}{b{+}{+}}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{3}}{\underset{2}{b{+}{+}}}$ | $\boxed{b=3}$ $a=1+2=\underline{3}$

a $a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{1}}{\underset{1}{{-}{-}b}}$ | $\boxed{b=1}$ $a=\underline{4}$

$a = \overset{\boxed{1}}{\underset{2}{{+}{+}b}} + \overset{\boxed{2}}{\underset{1}{b{+}{+}}} + \overset{\boxed{1}}{\underset{2}{b{-}{-}}}$ | $\boxed{b=1}$ $a=2+1+2$ $=\underline{5}$

Ex:- ③
```
main () {
  int x=10, y;
  y = ++x;
  Print ("%d", "%d", %u)
```
O/P:- y = ++x
= $\boxed{11}$ 11
= $\boxed{x=11}$ $\boxed{y=11}$

④ → y = x++
x = $\boxed{10}$
y = $\boxed{11}$

⑤ → $y = \overset{\boxed{11}}{\underset{10}{x{+}{+}}} + \overset{\boxed{12}}{\underset{12}{{+}{+}x}}$

$\boxed{X=12}$
$\boxed{Y=22}$

Logical Operators:-

① && (AND)
② || (OR)
③ ! (NOT)

Ex:-
```
int i=5;
if (i==5 && i6=20 && i!=6 && i>4)
{
  Printf ("welcome")
}
```
O/P:- True (welcome)

Ex:- if (i==6) → True But if i(i==6) → false

Conditional operator (?:)

- Also known as Ternary opⁿ. The Conditional statement are The Decision-making statements which depends upon the output of the expression

ex:-
```
char result;
int marks;
if (marks >35)
{
  result = 'p';
}
else
{
 result = 'f';
}
result = (marks > 35) ? result = 'p' : result = 'f';
  result = 'p';
  result = 'f';
```

Bitwise operator-

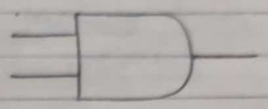
≫ , ≪ , XOR (exclusive OR).

ex:-
```
int main ()
{
   char var=3
   Printf ("%d", var << );
   var =3
   var = <<1  //Power

    ← <<
   0000 0011 → 3       3x2^1
   0000 0110 → 6     = [6]
               ↓
             empty
```

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
0	1	0	1	0	1	0	1

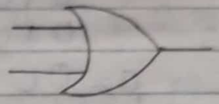
AND:-



$A.B$

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

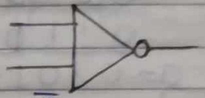
OR



$A+B$

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

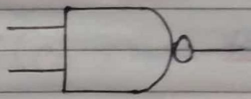
NOT



$\overline{A}$

A	$\overline{A}$
0	1
1	0

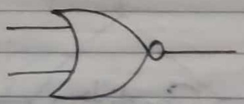
NAND



$\overline{A.B}$

A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

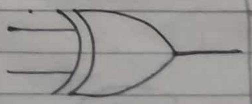
NOR



$\overline{A+B}$

A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

XOR



$A \oplus B$

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

0	0	0
0	1	1
1	0	1
1	1	0

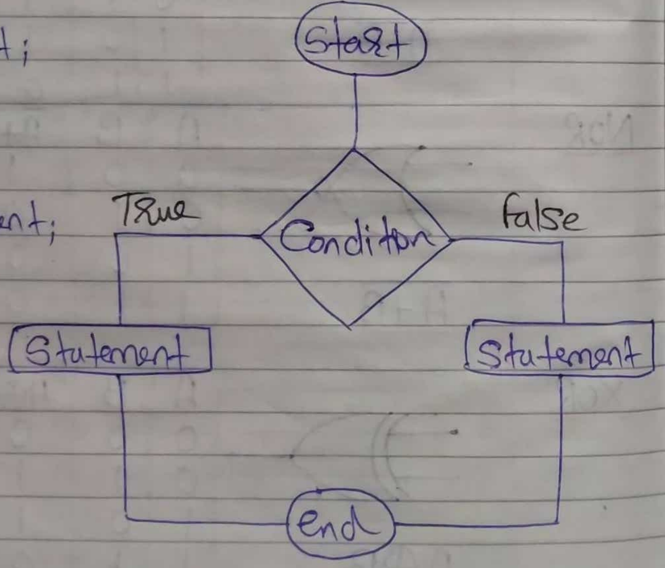
```
① a = 7^4
    0111 -> 7
    0100 -> 4
    ---------
    0011 -> 3

②
a=4   a= a^b          b= a^b          a= a^b
b=3   0100 -> a4      0111 -> a7      0111 -> 7 a
      0011 -> b3      0011 -> b3      0100 -> 4 b
    a=0111 -> 7     b=0100 -> 4     a= 0011 -> 3
                    [b=4]   [a=3]

③
a=7   0111 -> 7 a     0010 -> a2      0010 -> a 2
b=5   0101 -> 5 b     0101 -> b5      0110 -> b7
    a=0010 -> 2       0111 -> 7     a= 0101 -> 5
                    [b=7]   [a=5]
```

Control Structure - if .else

```
Syntax:- if (condition)
         {
           True statement;
         }
         else
         {
           false statement;
         }
```



Ex:-
```
# include <stdio.h>
int main ()
{
  int n=0;
  Printf ("Enter a number:");
  Scanf (" %d" ,&n);
  if (n% 2 ==0)
  {
    Printf ("%d is even number", number);
  }
}
```

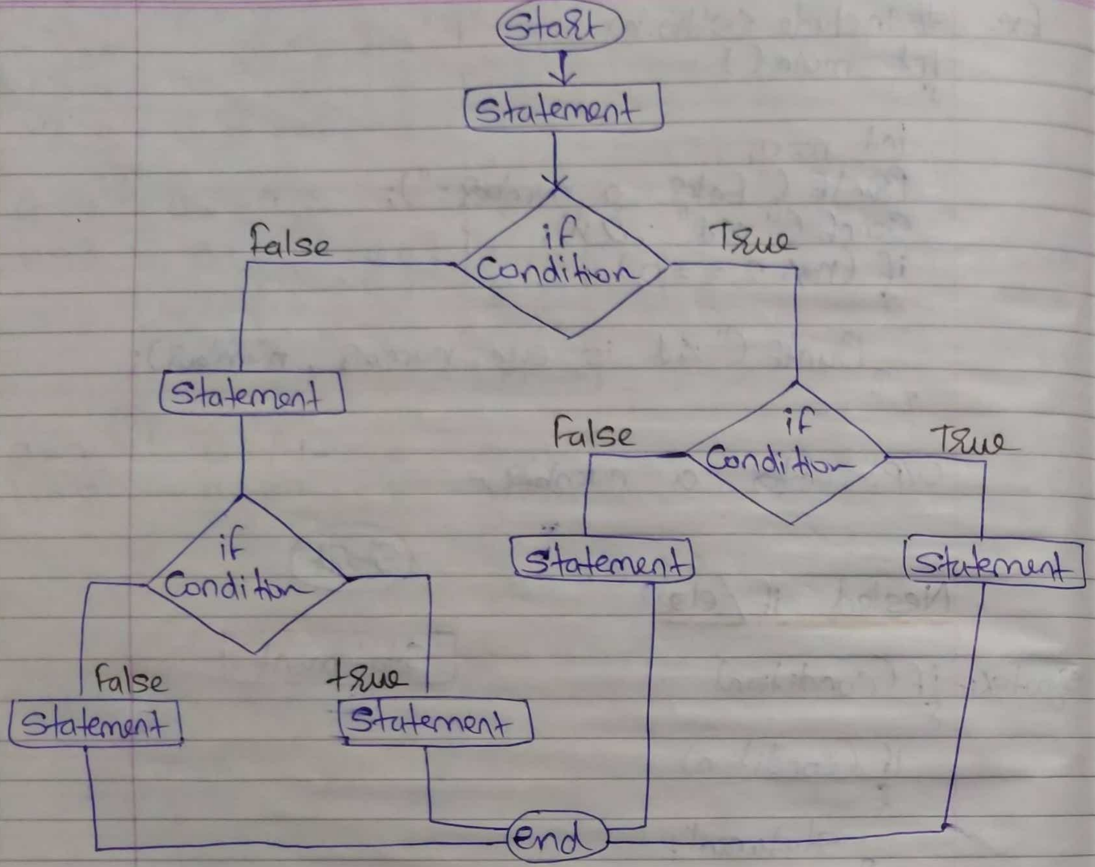
O/P:- enter a number:-

Start

~~Statement~~

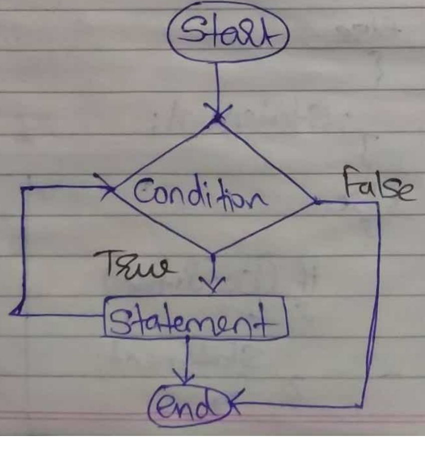
Nested if/else

```
Syntax:- if (Condition)
         {
           if (Condition)
           {
             Statement;
           }
           else
           {
             Statement;
           }
           else
           {
             if (Condition)          | else
             {                       | {
               Statement;            |   Statement;
             }                       | }
                                     | }
                                     }
```



While loop :-

```
Syntax:- while (condition)
         {
           Statement;
         }
```



ex:-

```
main ()
{
    int i=1;
    while (i<=10)
    {
      Printf ("%d", i++)
  }  }

O/P:- 1,2,3,...,10
```

```
while (i<=10)
{
  Printf ("i=%d", ++i);
}
O/P:- 2,3,4,...,10
```

ex:- a=7 b=7 c=10 (=, <, >, !=, >=, <=)

```
Printf ("%d  %d", a==b); True
Printf ("%d  %d", a<b);   true
Printf ("%d  %d", a>b);   true
Printf ("%d  %d", a==c);  false
Printf ("%d  %d", a<c);   true
Printf ("%d  %d", a>c);   true
Printf ("%d  %d", a!=b);  false
Printf ("%d  %d", a>=b);  True
Printf ("%d  %d", a<=b);  false
Printf ("%d  %d", a!=c);  True
```

```
int       [6]                      [7]      a=6
a=5   b=a++            -          b--       b=-3
b=8.      5            -          8
          [5]                     [-2]
      a = a--          +          ++b       a=4
          6                       -2        b=-2
          [5]                     [-3]
      b = ++a          +          --b       a=5
          5                       -3        b=2
```

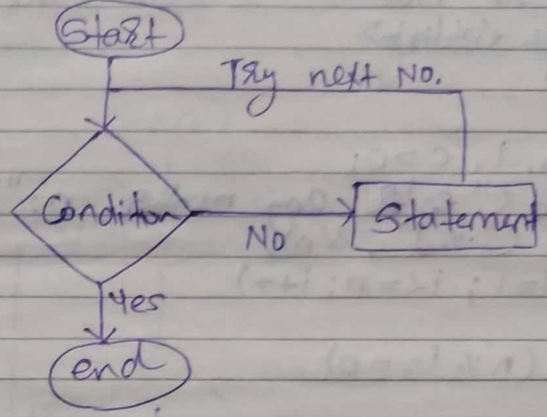
for loop :-

Syntax:-

```
     ①              ②                ⑤
for (Initialization ; Condition ; modify)
{          ③                 ④
    Statement ;
}
```

```
main ( )
{
  int n; i=sum=0;
  Printf ("enter n");
  Scanf ("%d", &n);
  for (i=1 ; i<=n; i++)
  {
     sum = sum + i ;
  }
  Printf ("sum of %d" %d", n, sum);
}
```

```
i=1   sum= 0+1 =1
i=2   sum= 1+2 =3
i=3   sum= 3+3 =6
i=4   sum= 6+4 =10
i=5   sum= 10+5 =15
```



Ex:- 1 to 10 (first 10 Natural) using for loop

```
#include <stdio.h>
void main ()
{
   int i;
   Printf ("The first 10 natural number are: \n");
   for (i=1 ; i<=10; i++)
   {
      Printf ("%d", i);
   }
} Printf ("\n");
```

O/P:- The first 10 natural number are:
1 2 3 4 5 6 7 8 9 10

Prime or Not:-

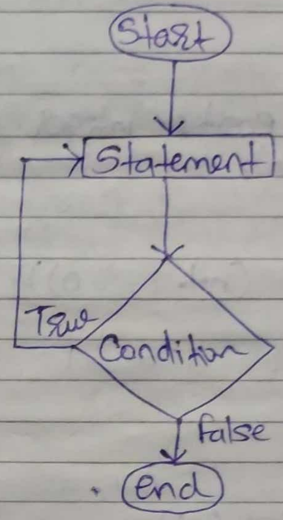
```
#include <stdio.h>
main ()
{
  int n, i, c=0;
  Printf ("enter any number n:");
  Scanf ("%d", &n);
  for (i=1; i<=n; i++)
  {
     if (n% i==0)
     {
       c++;
     }
  }
  if (c==2)
  {
    Printf ("n is a Prime number");
  }
  else
  {
    Printf ("n is not a Prime number");
  }
  return 0;
}
```

O/P:- enter any number n: 7

Do-while loop:-

Syntax:-

```
do
{
  Statement;
}
While (Condition)
```



ex:- Reversed Number:- (using while).

```
#include <stdio.h>
int main ()
{
  int n, reverse=0, rem;
  Printf ("enter a number:");
  Scanf ("%d", &n);
  While (n!=0)
  {
     rem = n %10;
     reverse = reverse * 10 + rem;
     n /=10;
  }
  Printf ("Reversed Number: %d", reverse);
  return 0;
}
```

O/P:- Enter a number:
1234
Reversed a number:
4321

Prime or Not (do while):-

```
#include <stdio.h>
Void main()
{
   int n, i, Flag=0;
   Printf ("enter a positive integer value: \n");
   Scanf (" %d", &n);
   do
   {
      if ((n!=2) && (n%i==0))
      {
         Flag=1;
         break;
      }
      i++;
   } while (i<=sqrt (n));
     if (Flag==0)
   Printf ("%d is prime number.", n);
     else
   Printf (" %d is Not Prime number.", n);
   getch();
}
```

O/P- enter number: 44   enter Number: 13
44 is Not Prime Number   13 is Prime Number

User with enter Pin 1234 (while loop):-

```
#include <stdio.h>
int main()
{
   int Pass, X=10;
   While (X!=0)
   {
      Printf (" \n Input the Password: ");
      Scanf (" %d", & Pass);
      if (Pass== 1234)
      {
         Printf ("Correct Password");
         X=0;
      }
      else
      {
         Printf ("Wrong Password, try another");
      }
      Printf ("\n");
   }
}
```

O/P:- Input the number:
1234
Correct Password