Operator precedence and Associativity :-

(1) Unary operator

(II) Arithmetic multiply, Division and Modulus.

(III) Arithmetic Add and subtract

(IV) Relational operators.

(V) Equality Operators.

(VI) Logical AND

(VII) Logical OR

(8) Conditional operator.

(9) Assignment operator

(10) Comma operators

481

√|

}

Conditional operator :-

① Also Called Ternary operator that work on 3 operands.

② Represented by symbol ?:

③ Usually can be used as Conditional statement like if-else.

Syntax :- Cond_exp1 ? exp2 : exp3 ;

Here Cond-exp1 specifies some Condition if this Condition evaluate to true then exp 2 is executed otherwise exp3 will be executed.

Ex.

```
int n1, n2; x;
    n1 = 10, n2 = 5
    x = n1>n2 ? 1 : 0;
```

If $(n_1 > n_2)$ is true then $x = 1$ otherwise $x = 0$;

Type Converelon:-

(1) ~~Example~~ Type conversion is the process of converting data of one type to another type there are two types of Type conversion possible in c-program given as below

(1) Implicit Type Conversion (Automatic Type Conversion)

(1) It is Automatically done by the system Internally
* without programmer intervention.

(ii) Usually in a mixed operands exp. all the lower datatype are converted to the highest ~~of~~ data type in the expression.

(2) Explicit Type Conversion (Type casting)

(1) It is specifically written by the programmer in his code

Ex. of implicit Type Conversion.

```
int x = 10;
float y = 5.3;        5.3 is double
double z;                   type.
z = (x*5 + y*2.6);
```

(11) ex.

```
float x;
int y;
y = 10/3;
x = (float) 10/3
printf("y=%d", y); // y=3
printf("x=%f", x); //x=3.333
```

operator precedence :-

operator Associativity :-

Associativity	
left to right	* /
left to right	+ -

⇒ 2 + 3 * 4 / 5 > 5 + 3
= 2 + 12/5 > 5 + 3
= 2 + 2 > 5 + 3
= 4 > 8
= 0

Preprocessor Directives :-

↳ File Inclusion

include <stdio.h>

include "myheader.h"

userdefine

user can define and save in C library.

Decision Making / Conditional Constructes -

Decision making are used to -

① Make decision during the program execution based on certain Conditions. It helps in choseing ⓔ one of the path from among several avaible path.

② These paths are nothing but the sequence of programming statement and instructions.

③ Break the sequential flow of the execution of program.

④ Fallowing are the decision making Constructs provided in C-language.

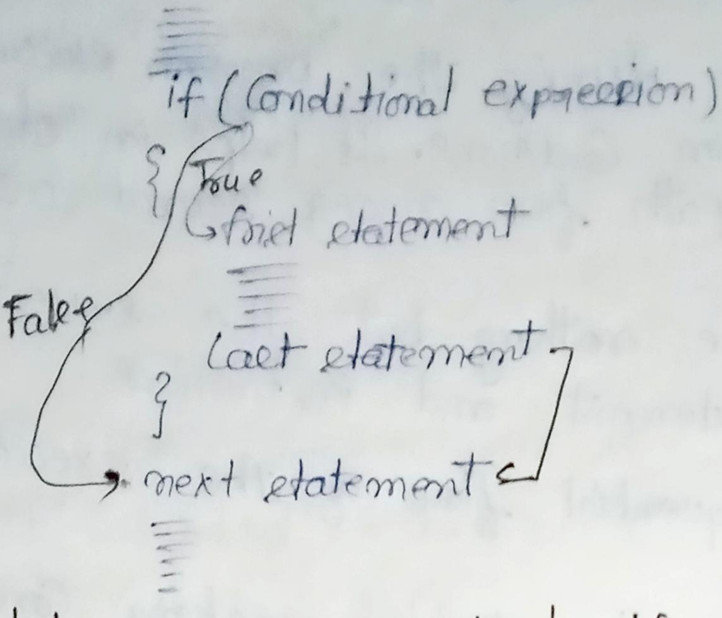
* 'if' and 'else' ~~structure~~ Constructe.
* 'switch' and 'case' Constructs.
* goto

① 'if' and 'else' Constructs:- i.e one of the decision making Construct in C-language.

- It chose from among two alternative Paths Corresponding to the result of the Condition; i.e either True or False.

- simple if
- if- else.
- if-else ladder

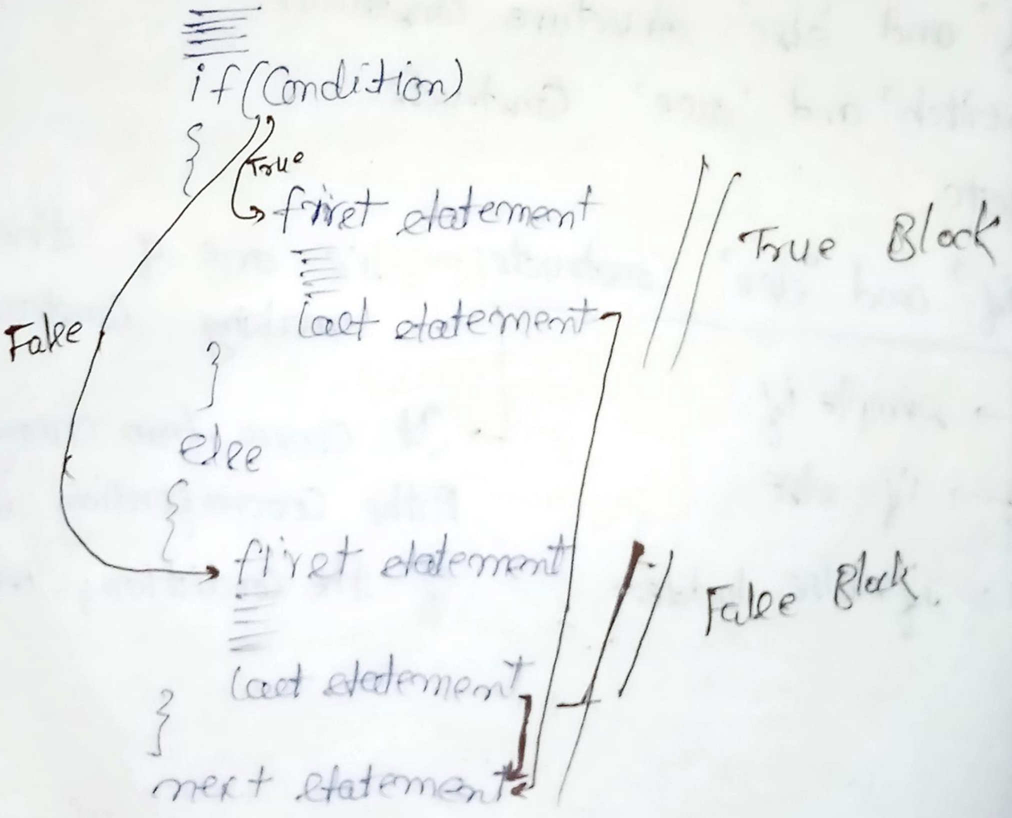
(a) Simple if



```
if (Conditional expression)
{
   first statement
   ...
   last statement
}
next statement
...
```

Control flow of ~~ey~~ simple-if

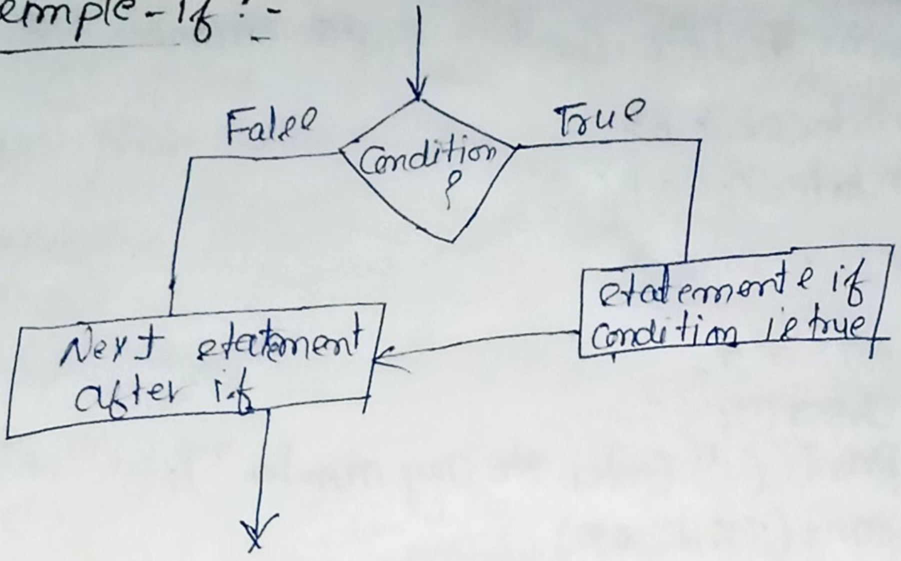
(b) ~~simple else~~ if-else:-



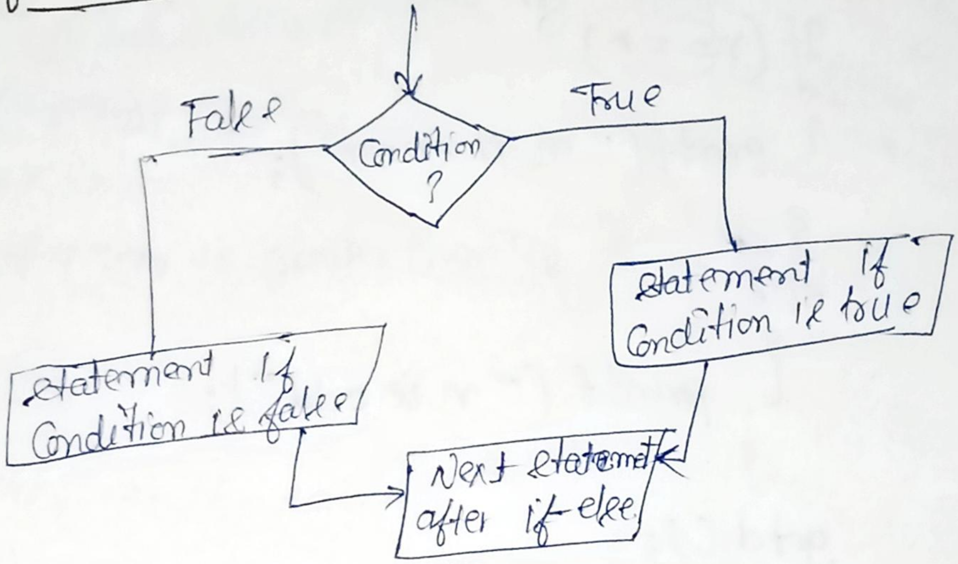
```
if (Condition)
{
   first statement
   ...
   last statement
}
else
{
   first statement
   ...
   last statement
}
next statement
```

Control flow of if-else:-

Flow chart of simple-if :-



Flow chart of if-else :-


False
True
Condition ?
Statement if Condition is true
statement if Condition is false
Next statement after if-else

(1) if

```
main()
{ int x;
  clrscr();
  printf("Enter the mark of student");
  scanf("%d", &x);
  if(x>33)
 { printf(" Student is pass");
 }
 if (x<=33)
 { printf (" student is fail");
  }
```

Nested if-else   Nested if-else ~~str~~ is a structure in which one if-else structure (or simple if) ~~it~~ contains another if else (or simple if) structure.

i.e:-
```
if ( )
{
    if( ) ...
    { ___ }
    else
    { ≡ }
}
```

Q. /* Program to find the largest number among three distinct numbers given as input */

```
# include <stdio.h>
# include <conio.h>
   void main()
{
   int n1, n2, n3;
   clrscr();
   printf("Enter three number");
   Scanf(" %d %d %d", &n1, &n2, &n3);
   if(n1>n2)
   {
       if (n1>n3)
       { printf(" n1 is largest");
       }
       else
       { printf("n3 is largest");
       }
   }
   ~~else~~ {
```

$n_1 > n_2$

n_3

switch Case :-

switch ~~such~~ case is a decision making Construct which provides multiway alternative. ~~Th~~ Based on the selective expression on of the alternative is selected to execute. It has following syntax.

```
switch (expr)
{
        case label1: // statements if 'expr' matches to 'label1'
                      break;
        Case label 2: // statement if 'expr' matches to label2
             |        break;
        case labeln: // statement if 'expr' matches to 'labeln'
                      break;
        defalt: // statement to be executed if 'expr' does
                 not match to any of the label's.
}
```

"expression is Combination of operands and operator."

Working of switch case :-

1. First of all 'expr' is Calculated.

step
2. The result of 'expr' is matched with labels one by one (label1, label2 up to labeln)

3. If the result of 'expr' is matches with 'label i' than the statements Corresponding to lebel i will be executed. if ~~operator~~ A break statement is encountered than flow of control ~~for~~ goes outside the switch block and Continues to execute next statement

after switch block.

Note:- If the value of "expr" does not matches to any of the lebel the ~~st~~ statement written under default section will be executed. Default is optional in switch case construct.

/* write a program to display a food menue to the user */

```
main()
{
    int ch;
    printf("Enter your choice");
     printf("\n ~~Press~~ 1 for tea");
     printf("\n Press 2 for samosa");
     printf("\n Press 3 for Cake");
     ~~Cake~~ scanf("%d", &ch);
     switch(ch)
     {
         Case 1: printf("ok, you will get Tea");
                 break;
         Case 2: printf("ok, you will get samosa");
                 break;
         Case 3: printf("ok, you will get cake");
                  break;
         default : printf("you entered wrong choice");
     }
    getch();
}
```

```
if (a%2==0)
   printf ("Number is ~~odd~~ even");
else
{ printf ("Number is odd");
}
break;
case 3 :
   printf (" Enter a number ");
    scanf ("%d", & a);
    for (b=1; b<=a; b++)
      printf ("%d", b);
    break;
case 4 : exit (0);
default :
      printf (" Invalide choice");
}
getch();
}
}
```

to check whether input number is even or odd*/