UNIT-II

DECISION CONTROL STRUCTURE

INTRODUCTION:-

* In a program all the instructions are executed sequentially by default, when no repetitio of some calculations are necessary. when in some situation we may have to change the execution order of statements based on condition or to repeat a set of statements until certain conditions are met.

* In such situation conditional and control statements are very useful.

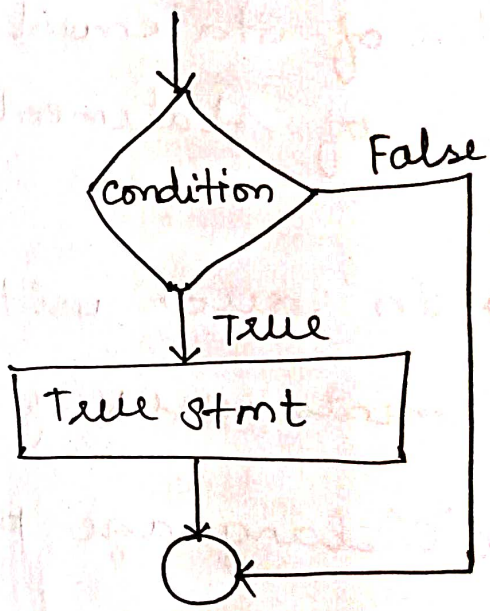
* 'C' language provides the following conditiona (decision making) statements

* if Statement
* if.... else Statement
* Switch statement
* Conditional operator
* goto statement.

THE IF STATEMENT:

* The if statement is a decision making statement. It is used to control the flow of execution of the statements and also used to test logically to find whether the condition is true or false.

* It is always used in conjunction with condition.



SYNTAX :-

```
if(condition is true)
  {
   true statement
  }
```

EXAMPLE :-

```
main()
{
 int i;
 printf("\n Enter the number...");
 scanf("%d", &i);
 if (i<=10)
```

```
    Printf("in the entered number is <10");
}
```

OUTPUT:-

Enter the number 5

The entered number is <10

* If you want to execute multiple statement in the if statement, that statements must be blocked with in braces.

EXAMPLE:-

```
main()
{
  int i;
  scanf("%d", &i);
  if (i<=10)
  {
   printf("%d", i);
   printf("in The entered number is less
                                than 10");
  }
}
```

OUTPUT:-

5

The entered number is less than 10

THE IF ELSE STATEMENT:

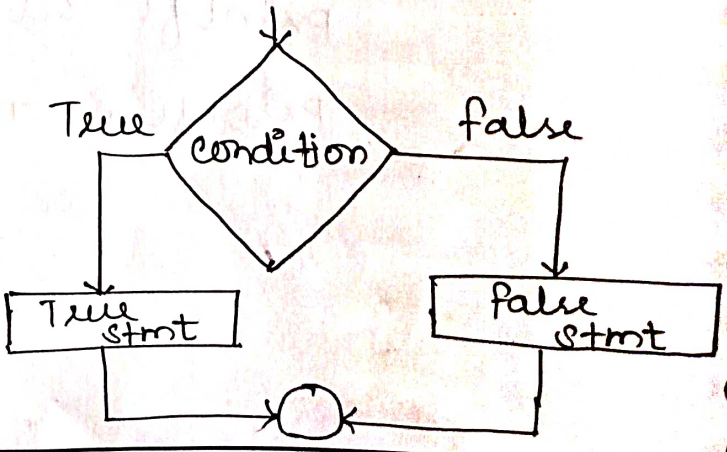
*It is basically two way decision making statement and always used in conjunction with condition.

*It is used to control the flow of execution and also used to carryout the logical test and then pickup one of the two possible actions depending on the logical test.

*It is used to execute some statements when the condition is true and execute some other statements, when the condition is false.

SYNTAX:-

```
if (condition)
{ True stmt; }
else
{ false stmt; }
```



EXAMPLE:-

```
main()
{
 int i;
 scanf("%d", &i);
 if(i<=10)
    printf("in the number is less than 10");
    else
    printf("in the number is greater than 10");
 }
```

OUTPUT:-

```
11
The number is greater than 10
```

* If there is only one statement in the if block (or) else block, then the braces are optional.

* But if there is more than one statement the braces are compulsary.

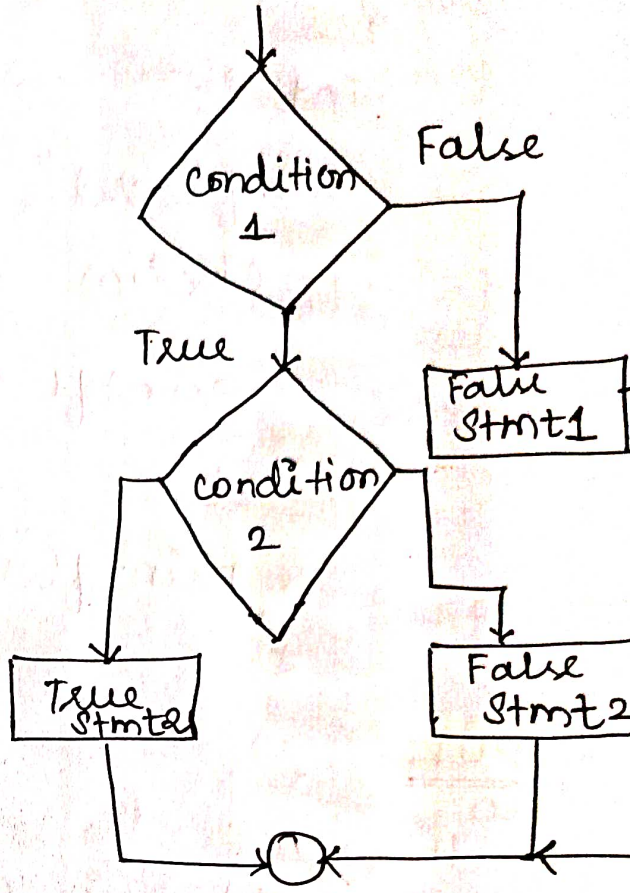
NESTED IF.... ELSE STATEMENT:

* When a series of if... else statements are occurred in a program, we can write an entire if.... else statement in another if... else statement

called nesting, and the statement is called nested if.

SYNTAX:

```
if (condition 1)
  {
   if (condition 2)
    {
      True statement 2;
    }
   else
    {
      false statement 2;
    }
   else
    {
      false statement 1;
    }
```



EXAMPLE :-

```
main ()
{
int n;
printf("in Enter a number");
Scanf("%d", &n);
if (n==15)
      printf("in play foot ball");
```

```
else
{
    if (n==10)
        printf(" play cricket);
        else
        printf(" don't play");
    }
}
```

OUTPUT:-

```
10
play cricket
```

* Note that the second if-else statement is nested in the first else statement. If the condition in the first if statement is false, then the condition in the second if statements is checked, if it is false the final else statement is executed.

* Similarly, we can construct any number of if ... else statements in nested fashion, that is called if.... else ladder.

* if you want to test more than one condition in if statement the logical operator are used as

```
else
 {
   if (n==10)
        printf(" play cricket);
        else
        printf(" don't play.");
   }
}
```

OUTPUT:-

10

play cricket

* Note that the second if-else statement is nested in the first else statement. If the condition in the first if statement is false, then the condition in the second if statements is checked, if it is false the final else statement is executed.

* Similarly, we can construct any number of if ... else statements in nested fashion, that is called if ... else ladder.

* if you want to test more than one condition in if statement the logical operator are used as

specified below. These are used to ~~combine~~ combine the results of two or more conditions.

operator	Meaning
&&	logical AND
||	logical OR
!	logical NOT

EXAMPLE :

```
main ()
{
 int i;
 printf("\n Enter the number");
 scanf(" %d", &i);
 if((i>10) && (i<20))
 printf("\n The Number is in b/w 10 & 20");
 else
  printf("\n The number is >20 or <10");
}
```

OUTPUT:- 15

The number is in b/w 10 & 20.

THE SWITCH STATEMENT:-

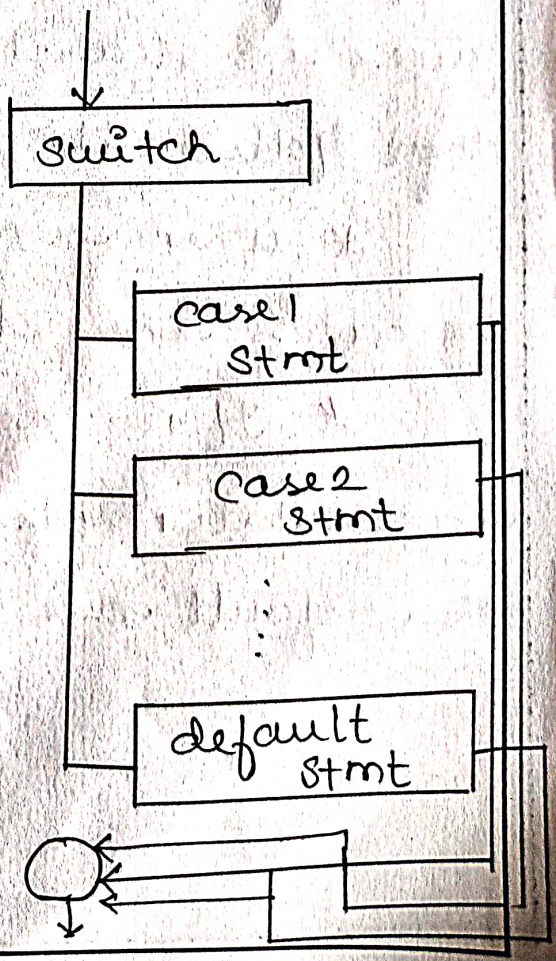
* The switch statement is used to pickup or execute a particular group of statements from several available group of statements.

* It allows us to make a decision from the number of choices.

* It is a multiway decision statement, it tests the value of given variable or expression against a list of case values and when a match is found, a block of statements associated with that case is executed.

SYNTAX:-

```
switch (expression)
{
 case constant 1:
    block 1;
    break;
 case constant 2:
    block 2;
    break;
 :
 default :
    default block
 break;
}
```



* The expression following the keyword switch is any 'c' expression that must yield an integer value.

* It must be an integer constant like 1, 2, 3 or an expression that evaluates to an integer.

* The ~~expression~~ keyword case is followed by an integer or a character constant. Each constant in each case must be different from all the other.

* First the integer expressions following the keyword switch is evaluated. The value it gives is searched against the constant values that follows the case statements.

* When a match is found, the program executes the statements following the case.

* If no match is found with any of the case statements, then the statements following the default are executed.

RULES FOR WRITING SWITCH() STATEMENT:-

* The expression in switch statement must be an integer value or a character constant.
* No real numbers are used in an expression.
* Each case block and default blocks must end with break statements.
* The default is optional and can be placed anywhere, but usually placed at the end.
* The case keyword must terminate with colon(:)
* No two case constants are identical.
* The case labels must be constants
* The switch can be nested.

EXAMPLE:-

*program to use the computer as calculator. The user input a code +, -, *, / and values that are to be computed.

```
#include <stdio.h>
main()
{
```

```
int a, b, c = 0;
char op;
clrscr();
printf("CALCULATION CODE");
printf("\n + ADD \n - SUB \n * MUL \n / DIV
                                    \n");
printf("Enter code ....");
scanf("%c", &op);
printf("Enter values.....");
scanf("%d %d", &a, &b);
switch(op)
{
   case '+':
    c = a+b;
    break;
   case '-':
      c = a-b;
      break;
   case '*':
     c = a*b;
        break;
```

```
case '/':
    c= a/b;
    break;
}
printf("\n Result is.... %d", c);
getch();
}
```

OUTPUT:-

calculation code

+ ADD

- SUB

* MUL

/ DIV

Enter code ..... +

Enter values 20 10

Result is 30