Branching and Looping

Branching Statements:

In sequential control, all the statements are executed in the order in which they are written sequentially in a program from top to bottom. However, it is necessary for the programmer to ***execute*** or ***skip*** certain set of statements *based on certain conditions* and is possible by using branching statements. These branching statements are also called as conditional or decision making statements.

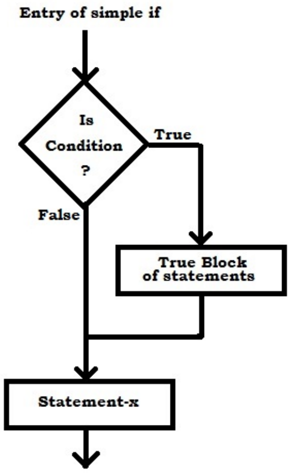
C language provides the following 5 types of branching statements for the programmers to take decisions.

1. Simple if statement (One way selection statement)
2. if … else statement (Two way selection statement)
3. nested if statement or nested if else statement (Multi way)
4. else if ladder or cascaded if statement (Multi way)
5. switch statement (Multi way)

Simple if statement: This is one-way selection statement available in C. It helps the programmer to ***execute*** or ***skip*** true block of statements *based on given condition*. The syntax, flowchart and examples are as follow.

Syntax:

```
        if (condition)
        {
        true block of statements;
        }
        Statements-X;
```



In the above syntax, the given condition will be evaluated for TRUE or FALSE by machine, if it is True, then the true block statements are executed; otherwise, no.

Example: C program to find largest of two numbers using simple if statement.

```
#include<stdio.h>
void main()
{
int n1,n2;
clrscr();
printf("Enter any two numbers");
scanf("%d%d",&n1,&n2);
        if (n2>n1)
        {
        printf("\n Largest number is %d", n2);
        }
        if (n1>n2)
        {
        printf("\n Largest number is %d", n1);
        }
getch();
}
```

```
#include<stdio.h>
void main()
{
int n1,n2,large;
clrscr();
printf("Enter any two numbers");
scanf("%d%d",&n1,&n2);
large=n1;
        if (n2>large)
        {
        large=n2;
        }
printf("\n Largest number is %d", large);
getch();
}
```

More Examples:

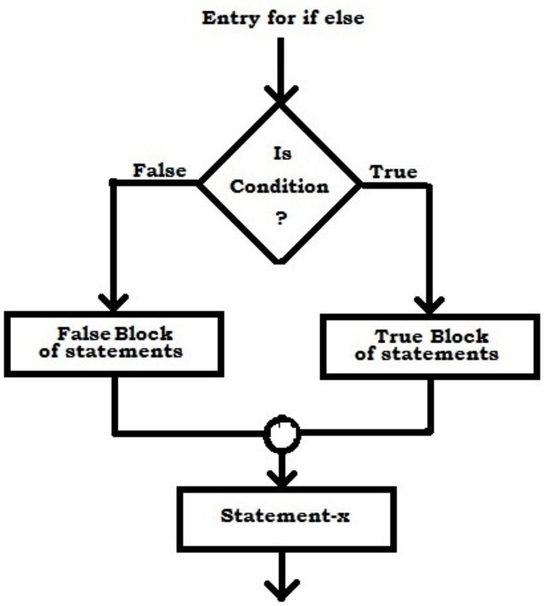
To find largest of three numbers.
large=a;
if(b>large)
large=b;
if(c>large)
large=c;
printf("\n Largest number is %d", large);
	To check for positive or negative number.
	
	if(n>0)
	printf("\n %d is positive number",n);
	if(n<0)
	printf("\n %d is negative number",n);

To check for EVEN or ODD number. if (n%2= =0) printf("\n %d is even number", n); if (n%2! =0) printf("\n %d is odd number", n);	To check eligibility for vote. if (age<18) printf("You are not eligible for vote"); if (age>=18) printf("You are eligible for vote");

2) if … else statement: It is two way selection statement. It executes either true block of statements or false block of statements based on a given condition. It is used by programmers to select one from two alternatives.

Syntax:

```
        if (condition)
        {
        True block of statements;
        }
        else
        {
        False block of statements;
        }
        Statement-X;
```



In the above syntax, the given **condition** will be evaluated for TRUE or FALSE by machine, if it is True, then the True Block of Statements are executed; otherwise, False Block of Statements are executed.

Example1:
C program to find largest of two numbers using simple if … else statement.

```
#include<stdio.h>
void main()
{
int n1,n2;
clrscr();
printf("Enter any two numbers");
scanf("%d%d",&n1,&n2);
        if (n1>n2)
        {
        printf("\n Largest number is %d", n1);
        }
        else
        {
        printf("\n Largest number is %d", n2);
        }

getch();
}
```

Example 2:

C program to check for leap year

```
#include<stdio.h>

void main()
{
int year;
clrscr();
printf(“Enter year to be check”);
scanf(“%d”,&year);
        if   ( ( (year%4==0) && (year%100!=0) ) || (year%400==0) )
        printf(“\n %d is leap year”, year);
        else
        printf(“\n %d is leap year”, year);
getch();
}
```

More Examples:

To find largest of two numbers.
if (a>b)
{
large=a;
}
else
{
large=b;
}
printf(“\n Largest number is %d”, large);
	To find largest of two numbers.
	if (a>b)
	{
	printf(“\n Largest number is %d”, a);
	}
	else
	{
	printf(“\n Largest number is %d”, b);
	}

To check for positive or negative number. if(n>0) printf(“\n %d is positive number”,n); else printf(“\n %d is negative number”,n);	To check for EVEN or ODD number. if (n%2= =0) printf(“\n %d is even number”, n); else printf(“\n %d is odd number”, n);

To check eligibility for vote.
if (age<18)
printf(“You are not eligible for vote”);
else
printf(“You are eligible for vote”);
	To check for word length of a host machine.
	int a;
	if (sizeof(a)==2)
	printf(“16 bits machine”);
	else
	printf(“Not 16 bits machine”);

More Examples on if ... else statement :

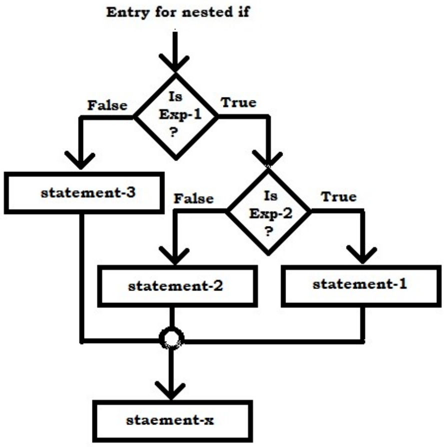
```
/*C program to find area of circle if the user name is your First Name and password is 999*/
#include<stdio.h>

#include<string.h>
void main()
{
char name[25];                    /* string variable or character array*/
int pwd;
float r,area;
clrscr();
printf("\n Enter User Name:");
scanf("%s",&name);
printf("\n Enter Password:");
scanf("%d", &pwd);
        if  (( strcmp(name,"john") ==0) && pwd= =999 )
        {
        printf("\n Login Successful ");
        printf("\n Enter radius of circle:");
        scanf("%f",&r);
        area=3.142*r*r;
        printf("\n Area of circle is %f",area);
        }
        else
        {
        printf("\n Invalid user name or password!");
        }
getch();
}
```

```
/*C program to check for upper case letters*/
#include<stdio.h>

void main()
{
char ch;
clrscr();
printf("\n Entera character:");
scanf("%c",&ch);
        if  (ch>='A' && ch<='Z')
        printf("\n %c is uppercase letter",ch);
        else
        printf("\n %c is not uppercase letter",ch);

getch();
}
```

3. Nested if statement: It is a multi-way selection statement. In this type, the programmer **uses *simple if* or *if … else* statement within another if statement**. Hence, it is called nested if statement. It helps to select one from many alternatives based on given conditions. The syntax, flowchart and examples are as follow.

Syntax:

```
if (exprn-1)
{
        if (exprn-2)
        {
        Statement1;
        }
        else
        {
        Statement2;
        }
}
else
        {
        Statement3;
        }
Statement-X;
```



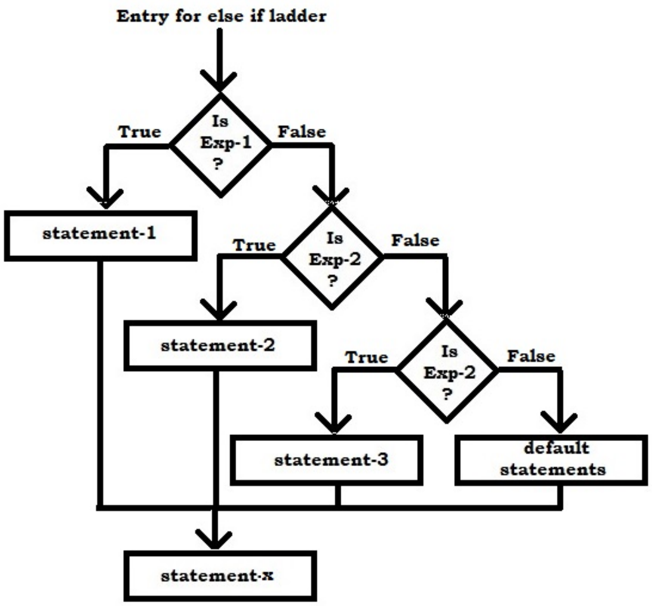
Example: To find largest of three numbers by using *nested if statement*

```
# include <stdio.h>
void main( )
{
int a, b, c, large;
clrscr();
printf("Enter three numbers");
scanf("%d%d%d", &a, &b, &c);
        if (a>b)
        {
                if (a>c)
                large=a;
                else
                large=c;
        }
        else
        {
                if (b>c)
                large=b;
                else
                large=c;
        }
printf("\n %d is large number", large);
getch();
}
```

4. The *else .. if ladder* or *cascaded if* statement: It is another multi way selection statement available in C to select one from many alternative. It helps the programmer to use another *if else* statement only within the *else* part of the previous *if else* statement. So, it is called else if ladder or cascaded if statement. The syntax, flowchart and examples are as follow.

Syntax:

```
if (expression-1)
        statement1;
else if (expression -2)
        statement2;
else if (expression -3)
        statement3;
else
        default statements;
statement-X;
```



In the above syntax, the expression-1 will be evaluated by machine for True or False, if it is True then Statement-1 executes followed by statement-x. Otherwise; other expressions will be evaluated for True/False to execute concerned block for the True condition. If, all the expressions are False then default statement will be executed by machine.

Example: C program to display the grades obtained by student based on the average marks scored.

Average Marks	Grade
80 to 100	Honors
60 to 79	First Division
50 to 59	Second Division
40 to 49	Third Division
0 to 39	Fail

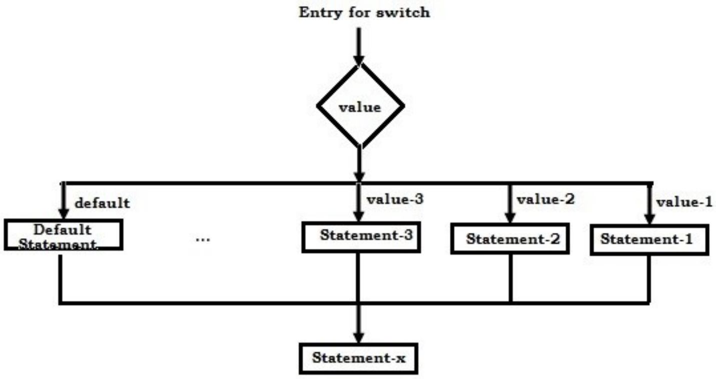
```
# include <stdio.h>
void main( )
{
float avg;
clrscr();
printf(“\n Enter average marks scored by student”);
scanf(“%f”, &avg);
if (avg>=80)
        printf(“\n Honors!!”);
else if (avg>=60)
        printf(“\n First Division”);
else if (avg>=50)
        printf(“\n Second Division”);
else if (avg>=40)
        printf(“\n Third Division”);
else
        printf(“\n Fail”);
getch();
}
```

```
/* To find roots of a quadratic equation*/
#include<stdio.h>
#include<math.h>
void main()
{
float a, b, c, r1, r2,d;
clrscr();
printf("\n Enter the coefficients of a Quadratic equation:");
scanf(" %f %f %f ", &a, &b, &c);
 if (a!=0)
{
        d=(b*b)-(4*a*c);
        if(d>0)
        {
        printf("\n Roots are real and distinct");
        r1=(-b + sqrt(d)) / (2*a);
        r2=(-b - sqrt(d)) / (2*a);
        printf("\n Root1=%f  \t Root2= %f ", r1, r2);
        }
        else if (d = =0)
        {
        printf("\n Roots are real and equal");
        r1=-b / (2*a);
        r2=r1;
        printf(" \n Root1=%f  \t Root2=%f ",r1,r2);
        }
        else
        {
        printf("\n Roots are imaginary");
        r1=-b / (2*a);
        r2=sqrt (abs(d)) / (2*a);
        printf("\n Root1= %f  + i %f   \t   Root2= %f  –i  %f ",r1, r2, r1, r2);
        }
 }
 else
        printf("Coefficients are not non zero");
getch();
}
```

5. **The Switch Statement**: It is another multi way statement. It helps the programmer to select one from many alternatives. It is mainly used by programmers to provide ***menu options*** for the end users of a program *to select one from displayed menu options*.

Syntax:

```
Switch (value)
{
case value1:
        statements block1;
        break;
case value2:
        statements block2;
        break;
case value3:
        statements block3;
        break;
…
…
…
default:
        default statement;
        break;
}
Statement-X;
```



In the above syntax, the passed value will be compared against all the case values one after another from top to bottom, wherever it matches that associated block of statements executes ; otherwise, default statement executes.

Limitations of using switch statement:

1. The value whichever, we pass to the body of switch for the comparison with values must be integer or character.

Rules for using switch statement:

1. The switch value must be an integer type. (or character also)
2. Case labels must be unique and constants.
3. Case labels must end with the colon(:).
4. The *break* and *default* statements are optional.

Example:

```
/*Simple C program to illustrate switch statement */
#include<stdio.h>
void main()
{
int ch;
clrscr();
printf("\n Enter your choice [1 2 3]?:");
scanf("%d",&ch);
        switch(ch)
        {
        case 1:
                printf("\n You have selected first choice");
                break;
        case 2:
                printf("\n You have selected second choice");
                break;
        case 3:
                printf("\n You have selected third choice");
                break;
        default:
                printf("\a Invalid choice!");
                break;
        }
getch();
}
```

Output 1:

```
Enter your choice [1 2 3]?: 2
You have selected second choice
```

Output 2:

```
Enter your choice [1 2 3]?: 5
Invalid choice!
```

More examples on Switch :

Write a c program to display the color names like Red, Green, Blue by reading first character of color as input.

```
/* To display color names */
#include<stdio.h>
void main()
{
char ch;
clrscr();
printf("\n Enter first character of color [R/G/B]? :");
ch=getchar();
        switch(ch)
        {
        case 'R':
        case 'r':
                printf("\n RED");
                break;
        case 'G':
        case 'g':
                printf("\n GREEN");
                break;
        case 'B':
        case 'b':
                printf("\n BLUE");
                break;
        default:
                printf("\a Invalid choice!");
                break;
        }
getch();
}
```

Output:

```
1.
Enter first character of color [R/G/B]? : b
BLUE
2.
Enter first character of color [R/G/B]? : G
GREEN
3.
Enter first character of color [R/G/B]? : i
Invalid choice
```

Write a c program to simulate simple calculator to perform arithmetic operations like addition subtraction, multiplication and division operations by reading ***operand1 operator operand2*** in form of an expression. Note: Check for *division by zero error* for division operation.[June/July2015 10M]

```
/*Simulation of Simple Calculator */
#include<stdio.h>
void main()
{
float a, b, ans;
char op;
clrscr();
printf("\n Enter arithmetic expression [operand1 operator operand2] :");
scanf("%f %c %f",&a, &op, &b);
        switch(op)
        {
        case '+':
                ans=a+b;
                printf("\n Addition is %f",ans);
                break;
        case '-':
                ans=a-b;
                printf("\n Subtract is %f",ans);
                break;
        case '*':
                ans=a*b;
                printf("\n Multiplication is %f",ans);
                break;
        case '/':
                if(b!=0)
                {
                ans=a/b;
                printf("\n Division is %f",ans);
                }
                else
                {
                printf("\n Division by zero error!");
                }
                break;
        default:
                printf("\a Invalid operator!");
                break;
        }
getch();
}
```

Write a c program to print a word for given digit (0- 9).

```
/*To print a word for a digit */
#include<stdio.h>
void main()
{
int digit;
clrscr();
printf("\n Enter a digit (0-9):");
scanf("%d",&digit);
        switch(digit)
        {
        case 0:
                printf("Zero");
                break;
        case 1:
                printf("One");
                break;
        case 3:
                printf("Three");
                break;
        /* Here write the code for remaining cases 4, 5, 6,7,8*/
        case 9:
                printf("Nine");
                break;
        default:
                printf("\a Invalid input!");
                break;
        }
getch();
}
```

Write a c program to check whether the character is vowel or not vowel.

```
/*To check character for vowel*/
#include<stdio.h>
void main()
{
char ch;
clrscr();
printf("\n Enter a character :");
scanf("%c",&ch);
        switch(ch)
        {
        case 'a':
        case 'A':
        case 'e':
        case 'E':
        case 'i':
        case 'I':
        case 'o':
        case 'O':
        case 'u':
        case 'U':
                printf("Vowel");
                break;
        default:
                printf("\a Not Vowel");
                break;
        }
getch();
}
```

Write a c program to find area of circle, area of triangle and area of rectangle by using switch statement to display menu options for the users to select any one to work accordingly.

```
*To find area of geometrical figures*/
#include<stdio.h>

void main()
{
int ch;
float r,l,b,breadth,height,area;
clrscr();
printf("\n 1. Area of Circle");
printf("\n 2. Area of Triangle");
printf("\n 3. Area of Rectangle");
printf("\n Enter your choice[1 2 3]?:");
scanf("%d",&ch);
        switch(ch)
        {
        case 1:
                printf("Enter radius of circle");
                scanf("%f",&r);
                area=3.142*r*r;
                printf("\n Area of circle is %5.2f",area);
                break;
        case 2:
                printf("Enter breadth and height of triangle");
                scanf("%f%f",&breadth,&height);
                area=0.5*breadth*height;
                printf("\n Area of triangle is %5.2f",area);
                break;
        case 3:
                printf("Enter length and breadth of rectangle");
                scanf("%f%f",&l,&b);
                area=l*b;
                printf("\n Area of rectangle is %5.2f",area);
                break;
        default:
                printf("\a Invalid choice!");
                break;
        }
getch();
}
```

Assignments:

1. Write a program to display week day (Sunday, Monday,…, Saturday) by reading a weekday number (1-7) by using switch statement. (from previous year QPs)
2. Write a c program to solve the following equations by using *else if ladder statement* and *switch statement*. (previous QPs for 12m)

   $y=1+x$ (if $x = 1$)

   $y=1+x^n$ (if $x = 2$)

   $y= n\ (1+x^3)$ Otherwise

3. **Explain all the five branching statements supported by C with syntax, flowchart and example. (important question for examination)**
4. Dear Student, you need to read recommended text books, reference books and notes with patience (slowly with consciousness) to understand the topics, re-write the key points of each studied topic without any reference. Revise the studied topics frequently to get success in your life. This is the secret behind success.
5. Know it: This notes acts as a reference to understand the topics. If, you want to score more marks in examination then you need to read recommended books and add some more key points to this notes.

Introduction to Loop Control Statements

As we know, the statements from written computer program executes sequentially from top to bottom. However, it is necessary for the programmer ***to execute certain block of statements repeatedly till certain condition satisfies*** while solving complex problems by using a machine and is possible by loop control statements. The loop statements are also called as iterative statements.

C provides the following three types of loop statements for the C programmers to repeat certain block of statements repeatedly till certain condition satisfies.

1. The **while** loop	(entry controlled or pre-test loop)
2. The **do … while** loop	(exit controlled or post test loop)
3. The **for** loop	(entry controlled or pre-test loop)

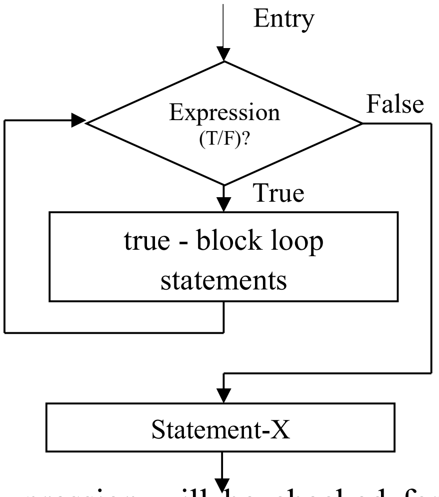
1. The **while** loop: It is one of the entry controlled loop statement. It repeats the given true-block of statements repeatedly till the given condition is *true*. Whenever, the condition becomes *false* then loop terminates.

The while loop is also called as pre-test loop or event controlled loop.

The syntax, flowchart and examples are as follow.

Syntax:

```
while  (expression)
{
true - block loop statements;
}
Statements-X;
```



In the above syntax, the given expression will be checked for True or False. If, it is True, then loop statements are executed repeatedly till the condition becomes False.

Examples:

To display the “Hello World!” message for 10 times.

```
/* Example for while loop */
#include<stdio.h>

void main()
{
int i;
clrscr();
i=1;
        while (i<=10)
        {
        printf(“\n Hello World!”);
        i=i+1;
        }
getch();
}
```

```
O/p:
Hello World!
Hello World!
…
…
Hello World!
```

To display 1 to 10 numbers.

```
/* Example for while loop */
#include<stdio.h>
void main()
{
int i;
clrscr();
i=1;
        while (i<=10)
        {
        printf(“\n %d”,i);
        i=i+1;
        }
getch();
}
```

```
O/p:
1
2
3
.
.
10
```

```
O/p:
Enter the value of n: 100
1
2
3
.
.
100
```

To display 1 to n numbers.

```
/* To display 1 to n numbers */
#include<stdio.h>

void main()
{
int i,n;
clrscr();
printf(“Enter the value of n: ”);
scanf(“%d”,&n);
i=1;
        while (i<=n)
        {
        printf(“\n %d”,i);
        i=i+1;
        }
getch();
}
```

To find sum of 1 to n numbers.

```
/* To find sum of 1 to n numbers */
#include<stdio.h>
void main()
{
int i, n, sum=0;
clrscr();
printf(“Enter the value of n: ”);
scanf(“%d”,&n);
i=1;
        while (i<=n)
        {
        sum=sum+i
        i=i+1;
        }
printf(“\n Sum is %d”,sum);
getch();
}
```

O/p:

```
Enter the value of n: 5
Sum is 15
```

```
/*To find sum and average of 1 to n numbers*/
#include<stdio.h>
void main()
{
int n,i,sum=0;
float avg;
clrscr();
printf("Enter n:");
scanf("%d",&n);
i=1;
        while (i<=n)
        {
        sum=sum+i;
        i++;
        }
avg=(float)sum/n;
printf("\n Sum=%d Average=%f",sum, avg);
getch();
}
```

```
O/p:
Enter the value of n: 5
Sum is 15 Average=3.00
```

Write a C program to find sum of odd numbers, even numbers and average of all numbers between 1 to n.

```
/*To find sum of even, odd and all nos with average of 1 to n numbers*/
#include<stdio.h>

void main()
{
int n,i,esum=0,osum=0,sum=0;
float avg;
clrscr();
printf("Enter n:");
scanf("%d",&n);
i=1;
        while (i<=n)
        {
                if (i%2==0)
                esum=esum+i;
                else
                osum=osum+i;
        sum=sum+i;
        i++;
        }
avg=(float)sum/n;
printf("\n Even Sum=%d Odd Sum=%d Sum=%d Average=%f", esum, osum, sum, avg);
getch();
}
```

```
O/p:
Enter the value of n: 5
Even Sum =6 Odd Sum=9 Sum=15 Average=3.00
```

Write a c program to find factorial of n.

```
/* To find factorial of n. */
#include<stdio.h>

void main()
{
int i, n, prod=1;
clrscr();
printf("Enter the value of n: ");
scanf("%d", &n);
i=1;
        while (i<=n)
        {
        prod=prod*i;
        i=i+1;
        }
printf("\n Factorial of %d is %d", n, prod);
getch();
}
```

```
O/p:
Enter the value of n: 5
Factorial of 5 is 120
```

Write a c program to reverse a number.

```
/*To reverse a number*/
#include<stdio.h>

void main()
{
int num, rev=0, rem;
clrscr();
printf("Enter n:");
scanf("%d", &num);
        while (num!=0)
        {
        rem=num%10;
        rev=(rev*10)+rem;
        num=num/10;
        }
printf("\n The reversed number is %d", rev);
getch();
}
```

```
O/p:
Enter n: 123
The reversed number is 321
```

CPL Lab. Program: Write a c program to check whether the number ‘n’ is palindrome or not palindrome.

```
/*To check for palindrome or not*/
#include<stdio.h>

void main()
{
int num, rev=0, rem, temp;
clrscr();
printf("Enter n:");
scanf("%d", &num);
temp=num;
        while (num!=0)
        {
        rem=num%10;
        rev=(rev*10)+rem;
        num=num/10;
        }
printf("\n The reverse of %d is %d", temp, rev);
if (temp==rev)
printf("\n %d is Palindrome", temp);
else
printf("\n %d is not Palindrome", temp);
getch();
}
```

```
O/p:
Enter n: 123
The reverse of 123 is 321
123 is not Palindrome
```

```
/*To repeat the application to find area of circle*/
#include<stdio.h>

void main()
{
float r,area;
char ch='y';
clrscr();
while(ch=='y'||ch=='Y')
{
printf("\n Enter radius of circle:");
scanf("%f",&r);
area=3.142*r*r;
printf("\n Area of circle is %f",area);
printf("\n Would you like to continue?[y/n]:");
ch=getch();
}
printf("\n Thank You");
getch();
}
```

Write a c program to display first ‘n’ Fibonacci numbers. [Exam Question]

```
/*To generate first n fibonacci numbers*/
#include<stdio.h>

void main()
{
int i=2,n,fib1=0,fib2=1,fib3;
clrscr();
printf("Enter n:");
scanf("%d",&n);
printf("\n Fibonacci Series\n");
printf("\n%d\n%d",fib1,fib2);
while(i<n)
{
fib3=fib1+fib2;
printf("\n%d",fib3);
fib1=fib2;
fib2=fib3;
i++;
}
getch();
}
```

```
O/p:
Enter n: 7
Fibonacci Series
0
1
1
2
3
5
8
```

```
/* To find GCD and LCM of two numbers by Euclid's algorithm*/
#include<stdio.h>

void main()
{
int m,n,temp1,temp2,rem,gcd,lcm;
clrscr();
printf("\n Enter any two numbers m and n:");
scanf("%d%d",&m,&n);
temp1=m,temp2=n;
while(n!=0)
{
rem=m%n;
m=n;
n=rem;
}
gcd=m;
lcm=(temp1*temp2)/gcd;
printf("\n GCD=%d and LCM=%d",gcd,lcm);
getch();
}
```

```
O/p:
Enter any two numbers m and n:
12 24
GCD=12 LCM=24
```

2) The do…while loop: It is exit controlled loop statement available in C. It helps the programmer to repeat certain block of true statements repeatedly till the given condition is True. Whenever, the given condition becomes False then the loop terminates.

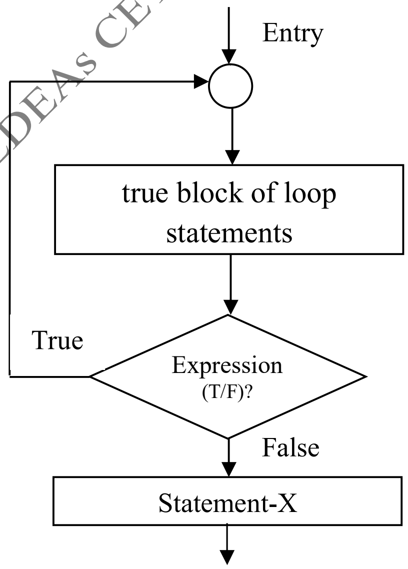
The major difference between *while* and *do…while* is that, if the given condition is False for the first time, then the body of the do while works at least once, whereas no execution of while loop.

The do while is also called as post test or event controlled loop.

The syntax, flowchart and examples are as follow.

Syntax

```
do
{
true block of loop statements;
} while (expression);
statement-X;
```



In the above syntax, the statements from true block of loop statements are executed repeatedly till the given expression is true.

To display 1 to n numbers.

```
/* To display 1 to n numbers */
#include<stdio.h>

void main()
{
int i,n;
clrscr();
printf("Enter the value of n: ");
scanf("%d",&n);
i=1;
        do
        {
        printf("\n %d",i);
        i=i+1;
        } while (i<=n);
getch();
}
```

```
O/p:
Enter the value of n: 100
1
2
3
.
.
100
```

```
/*To find sum and average of 1 to n numbers*/
#include<stdio.h>

void main()
{
int n,i,sum=0;
float avg;
clrscr();
printf("Enter n:");
scanf("%d",&n);
i=1;
        do
        {
        sum=sum+i;
        i++;
        } while (i<=n);
avg=(float)sum/n;
printf("\n Sum=%d Average=%f",sum, avg);
getch();
}
```

```
O/p:
Enter the value of n: 5
Sum is 15 Average=3.00
```

Write a C program to find sum of odd numbers, even numbers and average of all numbers between 1 to n.

```
/*To find sum of even, odd and all nos with average of 1 to n numbers*/
#include<stdio.h>

void main()
{
int n,i,esum=0,osum=0,sum=0;
float avg;
clrscr();
printf("Enter n:");
scanf("%d",&n);
i=1;
        do
        {
                if (i%2==0)
                esum=esum+i;
                else
                osum=osum+i;
        sum=sum+i;
        i++;
        } while (i<=n);
avg=(float)sum/n;
printf("\n Even Sum=%d Odd Sum=%d Sum=%d Average=%f", esum, osum, sum, avg);
getch();
}
```

```
O/p:
Enter the value of n: 5
Even Sum =6 Odd Sum=9 Sum=15 Average=3.00
```

Write a c program to find factorial of n.

```
/* To find factorial of n. */
#include<stdio.h>

void main()
{
int i, n, prod=1;
clrscr();
printf(“Enter the value of n: ”);
scanf(“%d”, &n);
i=1;
        do
        {
        prod=prod*i;
        i=i+1;
        }while (i<=n);
printf(“\n Factorial of %d is %d”, n, prod);
getch();
}
```

```
O/p:
Enter the value of n: 5
Factorial of 5 is 120
```

Write a c program to reverse a number.

```
/*To reverse a number*/
#include<stdio.h>

void main()
{
int num, rev=0, rem;
clrscr();
printf("Enter n:");
scanf("%d", &num);
        do
        {
        rem=num%10;
        rev=(rev*10)+rem;
        num=num/10;
        } while (num!=0);
printf("\n The reversed number is %d", rev);
getch();
}
```

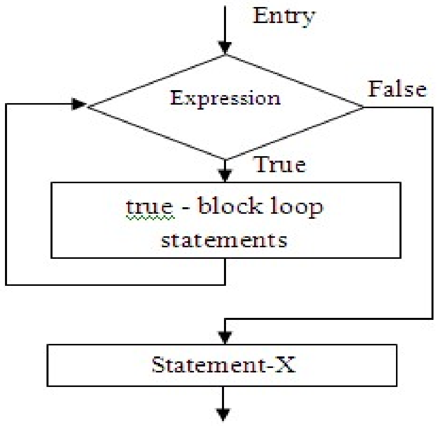
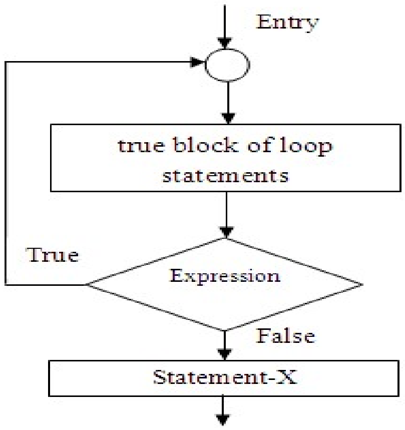
```
O/p:
Enter n: 123
The reversed number is 321
```

```
/*To repeat the application to find area of circle*/
#include<stdio.h>

void main()
{
float r,area;
char ch;
clrscr();
do
{
printf("\n Enter radius of circle:");
scanf("%f",&r);
area=3.142*r*r;
printf("\n Area of circle is %f",area);
printf("\n Would you like to continue?[y/n]:");
ch=getch();
} while(ch=='y'||ch=='Y');
printf("\n Thank You");
getch();
}
```

Differences between while and do while loop

Sl. No.	While loop	do … while loop
1	It is entry controlled loop statement. It repeats the given true block of loop statements repeatedly till the given condition is True. When the condition becomes False then the loop terminates.	It is exit controlled loop statement. It repeats the given true block of loop statements repeatedly till the given condition is True. When the condition becomes False then the loop terminates.
2	If the given condition is False for the first time, then no execution of while loop body.	If the given condition is False for the first time, then body of do while works at least once.

3	Syntax of while loop
	while (condition)
	{
	loop statements;
	}
	statement-x;
		Syntax of do … while loop
		do
		{
		loop statements;
		} while (condition);
		statement-x;

4	Flowchart  Entry Expression False True true - block loop statements Statement-X	Flowchart  Entry true block of loop statements True Expression False Statement-X

5	Example:
	program to find factorial of n
	i=1,prod=1;
	clrscr();
	printf(" Enter n:");
	scanf("%d",&n);
	while(i<=n)
	{
	prod=prod*i;
	i++;
	}
	printf("\n factorial is %d", prod);
		Example:
		program to find factorial of n
		i=1,prod=1;
		clrscr();
		printf(" Enter n:");
		scanf("%d",&n);
		do
		{
		prod=prod*i;
		i++;
		}while(i<=n);
		printf("\n factorial is %d", prod);

6	It is also called as pre test or event controlled loop.	It is also called as post test or event controlled loop.

3) The **for** loop: It is counter controlled loop statement available in C. It helps the programmer to repeat given true bock of loop statements repeatedly till the given expression is True. Whenever, the given condition becomes False then loop gets terminated. If the number of iterations well known by programmers then they prefer the use of for loop statement.

Syntax:

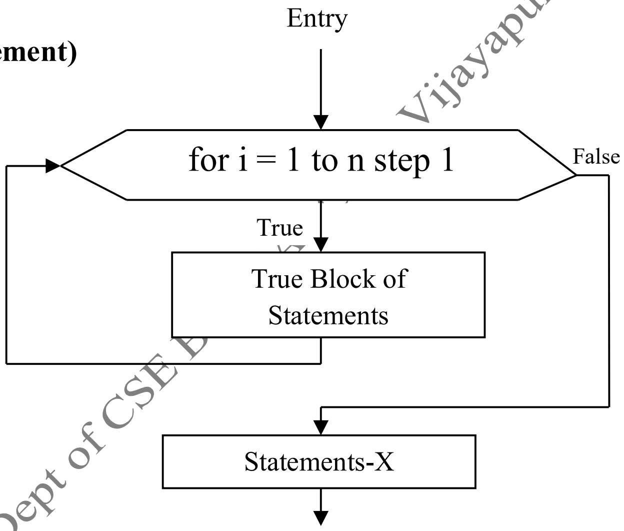
for (initialization; expression; increment/decrement)

{

True block of loop statements;

}

Statement-x;



Where,

Initialization → Counter will be assigned with initial value. e.g. i=1,i=0,etc.

Expression → The conditional expression on which number of iterations depends. E.g. i<n, i<=n, etc.

increment/decrement → increment or decrement expression for the counter. E.g. i++, j++, i--, j--,etc.

To display 1 to 10 numbers.

```
#include<stdio.h>

void main()
{
int i;
clrscr();
i=1;
        for(i=1;i<=10;i++)
        {
        printf("\n %d",i);
        }
getch();
}
```

```
O/p:
1
2
3
.
.
10
```

To display 1 to n numbers.

```
/* To display 1 to n numbers */
#include<stdio.h>

void main()
{
int i,n;
clrscr();
printf(“Enter the value of n: ”);
scanf(“%d”,&n);
i=1;
        for(i=1;i<=n;i++)
        {
        printf(“\n %d”,i);
        }
getch();
}
```

```
Enter the value of n: 100
1
2
3
.
.
100
```

Write a c program to find factorial of n.

```
/* To find factorial of n. */
#include<stdio.h>

void main()
{
int i, n, prod=1;
clrscr();
printf(“Enter the value of n: ”);
scanf(“%d”, &n);
i=1;
        for(i=1;i<=n;i++)
        {
        prod=prod*i;
        }
printf(“\n Factorial of %d is %d”, n, prod);
getch();
}
```

```
O/p:
Enter the value of n: 5
Factorial of 5 is 120
```

```
/*To display multiple table of n*/
#include<stdio.h>

void main()
{
int i,n;
clrscr();
printf("Enter n:");
scanf("%d",&n);
printf("\n Multiple Table of %d \n",n);
for(i=1;i<=10;i++)
{
printf("\n %d X %d = %d",n,i,n*i);
}
getch();
}
```

```
O/p:
Enter n: 5
Multiple Table of 5
5  X  1  = 5
5  X  2  =10
5  X 3  = 15
…
…
…
5  X 10  = 10
```

Nested loops: The programmer can use one loop statement within another loop statement and is called nested loop. It informs the machine to do repetitive work repeatedly. It is used by programmer to solve complex problems that include complex repetitive steps. The ANSI C supports 32 times of nesting of loops.

Examples

```
i=1;
while (i<=5)            /* outer loop works for 5 times */
{
        j=1;
                while (j<=10)  /*every time inner  loop works for 10 times*/
                {
                printf("\t VTU, Belagavi");
                j++;
                }
        i++;
        printf("\n");
}
```

Output: The given message "VTU, Belagavi" will be displayed for 5 X 10 i.e. 50 times.

To read elements of matrix A[m x n].

```
printf("\n Enter elements of matrix A");
for(i=0;i<m;i++)                /* outer loop works for m times [to generate row index] */
{
        for(j=0;j<n;j++)        /*every time inner  loop works for n times [column index] */
        {
        scanf("%d", &a[i][j]);
        }
}
```

To display elements of matrix A[m x n] in matrix format.

```
printf("\n Elements of matrix A \n");
for(i=0;i<m;i++)                /* outer loop works for m times [to generate row index] */
{
        for(j=0;j<n;j++)        /*every time inner  loop works for n times [column index] */
        {
        printf("%d", a[i][j]);
        }
        printf("\n");
}
```

```
/*To display the number pattern*/
#include<stdio.h>

void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
        for(j=1;j<=i;j++)
        {
        printf("\t %d",j);
        }
printf("\n");
}
getch();
}
```

```
O/p:
1
1       2
1       2       3
1       2       3       4
1       2       3       4       5
```

```
/*To display the number pattern*/
#include<stdio.h>

void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
        for(j=1;j<=i;j++)
        {
        printf("\t *");
        }
printf("\n");
}
getch();
}
```

```
O/p:
*
*       *
*       *       *
*       *       *       *
*       *       *       *       *
```

```
/*To display the pattern*/
#include<stdio.h>

void main()
{
int i,j,k;
clrscr();
for(i=1;i<=5;i++)
{
        for(j=5;j>=i;j--)
        {
        printf(" ",j);
        }
        for(k=1;k<=i;k++)
        {
        printf(" *");
        }
printf("\n");
}
getch();
}
```

```
O/p:
                *
             *     *
          *     *     *
       *     *     *     *
    *     *     *     *     *
```

Know it :

```
for ( i=1,j=10; (i<=10 && j>=1);i++,j--;)
{
printf("\n %d  \t  %d", i, j);
}
o/p:    1       10
        2       9
        3       8
        …
        …
        10      1
```

```
for ( ; ;)
{
printf ("Hello World!");
}
o/p: infinite loop
```

Introduction to Break and Continue Statements

As we know, the loop control statements like while, do while and for are used by programmers to repeat certain block of statements repeatedly till the given condition is True. However, it is necessary for the programmers ***to exit*** from loop body before the condition satisfies or ***to skip*** certain statements from loop body based on certain conditions. This is possible by making the use of ***break*** and ***continue*** statements available in C. These are also called as loop interrupt statements.

The **break** statement:

This is one of the loop interruption statements. It is used by C programmers **to exit** from the body of *while*, *do-while* and *for* loop statements based on certain condition. i.e loop terminates before the *expression* of loop becomes False. It is usually used in association with the *if* statement. A break is also used to exit from particular case of *switch* statement .

Syntax:

```
while (expression)           do                           for(exp1;exp2;exp3)
{                            {                            {
statements;                  statements;                  statements;
        if (condition)               if (condition)               if (condition)
        break;                       break;                       break;
statements;                  statements;                  statements;
}                            } while (expression);        }
statement-x;                 statement-x;                 statement-x;
```

In the given syntax, whenever the given *if (condition)* becomes True then loop terminates followed by statement-x.

Examples:

```
i=1;                  O/p:          for(i=1; i<=100 ; i++)    O/p:
while(i<=100)         1             {                         1
{                     2                     if ( i = = 5)     2
        if ( i = = 5) 3                     break;            3
        break;        4                                       4
printf("\n %d", i);                 printf("\n %d", i);
i++;
}                                   }
```

Write a c program to check whether the unknown number ‘n’ is prime or not prime.

```
/* To check for prime number*/
#include<stdio.h>

void main()
{
int n,i,flag=1;
clrscr();
printf("\n Enter n:");
scanf("%d",&n);
for(i=2;i<=n/2;i++)
{
        if(n%i==0)
        {
        flag=0;
        break;
        }
}
if(flag = =1)
printf("\n %d is prime number",n);
else
printf("\n %d is not prime number",n);
getch();
}
```

O/p:

```
Enter n: 7
7 is prime number


Enter n: 20
20 is not prime number
```

The **continue** statement: It is another loop interrupt statement available in C. It helps the programmers ***to skip*** certain statements from the body of loop statements like *while, do while* and *for* based on certain conditions.

Syntax:

```
while (expression)
{
statements;
        if (condition)
        continue;
statements block2;     /* skip */
}
statement-x;
```

```
for(exp1;exp2;exp3)
{
statements;
        if (condition)
        continue;
statements block2;       /* skip */
}
statement-x;
```

In the above syntax, whenever the given *if (condition)* becomes True then statements bock-2 are not going to execute.

Examples:

```
i=1;                      O/p:            for(i=1; i<=5 ; i++)     O/p:
while(i<=5)               1               {                        1
{                         2                   if ( i = = 3)        2
    if ( i = = 3)         Infinite            continue;            4
    continue;             loop            printf("\n %d", i);      5
printf("\n %d", i);
i++;                                      }
}
```

To find sum of even numbers between 1 to n by using *continue* inside loop.

```
/* Example for continue statement*/
#include<stdio.h>

void main()
{
int n,i,sum=0;
clrscr();
printf("\n Enter n:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
    if (i%2!=0)
    continue;
sum=sum+i;
}
printf("\n Sum of even numbers =%d",sum);
getch();
}
```

O/p:

```
Enter n: 6
Sum of even numbers 12
```

Differences between ***break*** and ***continue*** statements

Sl.No.	The ***break*** statement	The ***continue*** statement
1	It is loop interruption statement. It terminates (**<u>breaks</u>**) the repetitive process of loop statements (*while*, *do while* and *for* loop) based on certain conditions. i.e loop terminates before the loop expression becomes False.	It is also loop interruption statement. It **<u>skips</u>** certain statements from the loop statements based on certain conditions.

2	Syntax:
	while (expression)
	{
	statements;
	        if (condition)
	        break;
	statements;
	}
	statement-x;
		Syntax:
		while (expression)
		{
		statements;
		        if (condition)
		        continue;
		statements;
		}
		statement-x;

3	Example:
	for(i=1;i<=5;i++)
	{
	if (i= = 3)
	break;
	printf("\t%d",i);
	}
	o/p: 1 2
		Example:
		for(i=1;i<=5;i++)
		{
		if (i= = 3)
		continue;
		printf("\t%d",i);
		}
		o/p: 1 2 4 5

4	It can be used in the body of *switch* statement to terminate the cases.	It is not permitted in the body of *switch* statement.

The *goto* statement (unconditional statement):

The *goto* is unconditional statement available in C. It transfers the program execution flow from one point of the program to another point. It is usually used by programmers to exit from deeply nested loops. Unfortunately, the programmers do not use goto statement in structured programming because the program debugging and modification is difficult.

Syntax:

```
Statements;
label:
statements;
        if (expr-n)
        goto label;
statements;
```

```
Statements;
if (expr-n)
        goto label;
statements;
label:
statements;
```

```
/* Example for goto statement*/
#include<stdio.h>

void main()
{
int n;
clrscr();
top:
printf("\n Enter n:");
scanf("%d",&n);

        if (n<0)
        {
        printf("\n Negative number! \n");
        goto top;
        }

        if (n%2==0)
        printf("\n %d is even number",n);
        else
        printf("\n %d is odd number",n);
getch();
}
```

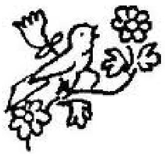
```
O/p:
Enter n:
-7
Negative number!
Enter n: 8
8 is even number
```

```
/* To find sum of all numbers till user enters 999*/
#include<stdio.h>

void main()
{
int n,sum=0;
clrscr();
top:
printf("\n Warning!!! Press 999 to exit.");
printf("\n\n Enter numbers to be sum:");
for(;;)
{
scanf("%d",&n);
if (n==999)
goto bottom;
sum=sum+n;
}
bottom:
printf("\n Sum of all numbers=%d",sum);
getch();
}
```

```
O/p:
Warning!!! Press 999 to exit.
Enter numbers to be sum:
1 2 3 4 5 999
Sum of all numbers =15
```

Important Questions on Module-2 for Examination

1. Explain all the FIVE branching statements with syntax, flowchart and minimum two examples.
2. Explain all the THREE loop statements with syntax, flowchart and minimum two examples.
3. Explain *break*, *continue* and *goto* statement with syntax and examples.
4. Write the difference between while and do while loop statements.
5. Write the differences between break and continue statements.
6. Practice the following important programs on rough paper for examination purpose.
   - a. To find largest of 3 numbers
   - b. The programs on switch statements.
   - c. To find factorial of n by using while loop
   - d. To find factorial of n by using do … while loop
   - e. To find factorial of n by using for loop
   - f. To reverse a number
   - g. To check for palindrome
   - h. To find GCD and LCM of two numbers by using Euclid’s algorithm.
   - i. To print first N Fibonacci numbers.
   - j. To find sum of even numbers, sum of odd numbers, sum of all numbers and average of 1 to n numbers.
   - k. To print multiple table of ‘n’.
   - l. To check for prime number.



WIT and WISDOM

Don’t compare yourself with anyone in this world, if you do so, you are insulting yourself. –Alen Strike

Don’t compare yourselves with others, instead compare yourself with your last performance- Bill Gate

Attitude is a little thing that makes a big difference. – Winston Churchill

There is no use of running fast, when you are on the wrong road. So, first choose the correct way in your life. - German Proverb

The causes of failure in the examination [By- Dr. C. R. Chandrashekar]

Subject / Course is difficult for the student or he/she has no interest in it. It was forced on them by others.

Wrong study habits like (continuous reading without understanding, no review, no recall exercises, poor in representation, last hour preparation, poor time management, poor self confidence, and negative attitudes.)

The future depends on what we do in the present – Mahatma Gandhi

“Success is the sum of small efforts, repeated day in and day out.” ~ Robert Collier

“Learn from mistakes”- Thomas Edison tried two thousand different materials in search of a filament for the light bulb. When none worked satisfactorily, his assistant complained, “All our work is in vain. We have learned nothing.”Edison replied very confidently, “Oh, we have come a long way and we have learned a lot. We know that there are two thousand elements which we cannot use to make a good light bulb.”

When you are not practicing, then remember that, someone somewhere is practicing for the same and when you will meet him/her, then he/she will be the winner. – By Ed Macauley

Striving for success without hard work is like trying to harvest where you have not planted. – By David Bly

The roots of education are bitter, but the fruits are sweat- Aristotle.

“If you try and lose then it isn't your fault. But if you don't try and we lose, then it's all your fault.” — Orson Scott Card, *Ender's Game*

“Hard work is much more important than talent”— Carlo Rotella

“Yesterday is gone. Tomorrow has not yet come. We have only today. Let us begin.” — Mother Teresa

“The future depends on what you do today.”— Mahatma Gandhi

“The future belongs to those who believe in the beauty of their dreams.” — Eleanor Roosevelt

Dear Student,

You are talented student, only you need to do is that read more to understand unknown things.

```
while (Not Understood)
{
read (recommended text books || notes || both);
}
if ( you have understood )
        Success in examination or life;
else
        First Attempt In Learning;
```

Read → Read → Read → Understand → Re Write → Re call → Revise frequently → Success (Ur dreams will be converted to True)

…