Programming in C and Data Structures (15PCD13/23)
Module 2
Branching and Looping

Topics:
Two way selection (if, if-else, nested if-else, cascaded if-else),switch statement, ternary operator? Go to, Loops (For, while-do,do-while) in C, break and continue, Programming examples andexercises.
Course Outcome:
Understand the basic principles of Programming in C language.

Selection statements are those in which the statements are executed depending upon the condition.
If the condition is true, a true block is executed, otherwise the false block is executed. A block is a set of statements enclosed within a pair of { }.
In selection statements, relational and logical operators are used.
Examples

Original expression	Simplified expression
! (x < y)	x > = y
! (x < = y)	x > y
! (x > y)	x < = y
! (x > = y)	x < y
! (x = = y)	x ! = y
! (x ! = y)	x = = y

Decision making statements

- Two way Selection
  - Simple if statement
  - if – else statement
  - if - else ladder
  - Nested if statement
- Multi way selection
  - switch statement

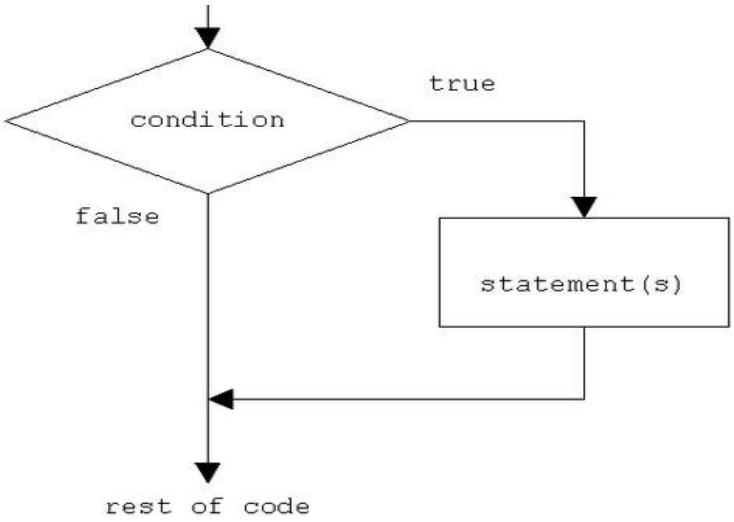
Simple if statement

Syntax :

```
if( (condition)
{
        true block
}
statement x;
```

Working :

If condition is **true**
True block
+
statement x
If condition is**false**
statement x



It is basically a two way decision statement and it is used in conjunction with an expression. It is used to execute a set of statements if the condition is true. If the condition is false it skips executing those set of statements.

Example

```
int a, b, c;
printf("Enter a & b\n");
scanf("%d %d", &a, &b);
if(a<=b)
{
        b+ = 3;
        c = a + b;
        printf("%d "%d", b, c);
}
printf("\nEnd of program");
```

Output

Run 1

```
Enter a & b
3 15
18  21
End of program
```

Run 2

```
Enter a & b
13 5
End of program
```

Example

```
int a, b, c;
printf("Enter a and b\n");
scanf("%f %f", &a, &b);
if(a>=b)
printf("a & b equal\n");
if(a>b+5)
printf("a grater than b\n");
printf("End");
```

Output

Run 1

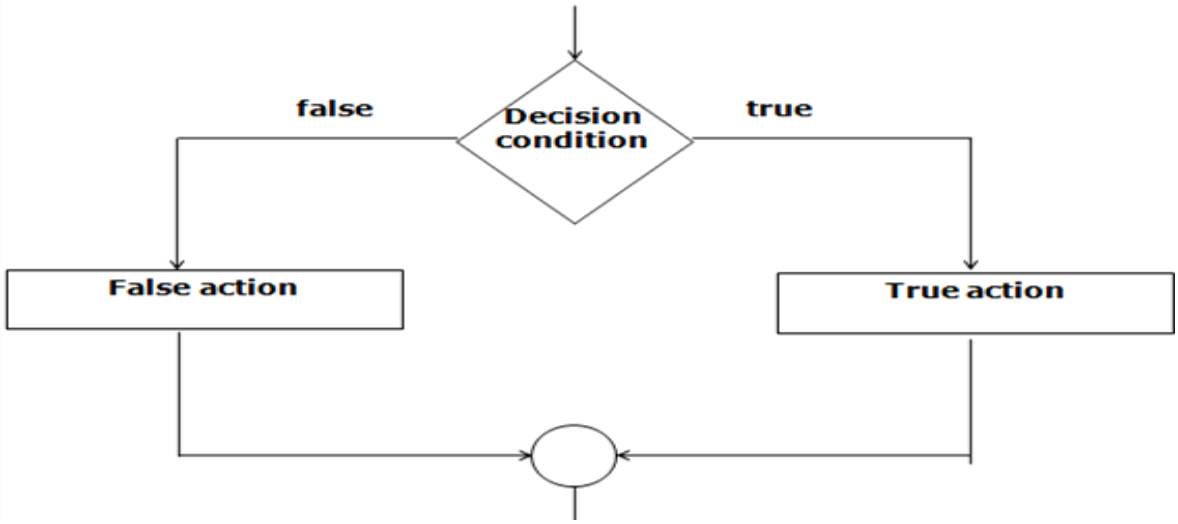
```
Enter a and b
5 5
a& b equal
End
```

Run 2

```
Enter a and b
10 3
a& b equal
a greater than b
End
```

if – else statement

It is an extension of if statement. It is used to execute one among the two set of statements at a time. If condition is true it executes the true block otherwise it executes the false block. The syntax and flow diagram is shown below



Syntax :

```
if( (condition)
{
        true block
}
else
{
        false block
}
statement x;
```

Working :

If condition is <u>true</u>

True block + statement x

If condition is <u>false</u>

False block + statement x

<u>Note :</u>

A pair of { } is not necessary when a true or false block has only one statement.

Example:

```
int a=56, b=89, c=12, d=44;
if(a>=b)
{
        c = c * 5;
}
else
{
        d = d%2 ;
}
printf("End of program");
```

Can be written as

```
int a=56, b=89, c=12, d=44;
if(a>=b)
c = c * 5;
else
d = d%2 ;
printf("End of program");
```

Statement x

Examples

1.
```
int a = 15, b = 10, c;
    if(a<=b)
    {
            b+ = 3;
            c = a + b;
            printf("%d "%d", b, c);
    }
    else
    {
            a++;
            b++;
            printf("%d %d",a,b);
    }
    printf("\nEnd of program");
```

Output
16 11
End of program

2.
```
float a, b, c;
    printf("Enter a and b\n");
    scanf("%f %f", &a, &b);
    if(b==0)
    printf("Invalid value of b\n");
    else
    {
            c = a/b;
            printf("c=%f",c);
    }
```

Output
Execution 1
Enter a and b
12
5
c=2.4
Execution 2
Enter a and b
12
0
Invalid value of b

3.
```
float a, b, c;
    printf("Enter a and b\n");
    scanf("%f %f", &a, &b);
    if(b==0)
    printf("Invalid value of b\n");
    else
    c = a/b;
    printf("c=%f",c);
```

Statement x

Output
Execution 1
Enter a and b
12
5
c=2.4
Execution 2
Enter a and b
12
0
Invalid value of b
c=garbage value

4. Input an integer and check whether it is odd or even

```
void main()
{
        int n;
        printf("Enter an integer");
        scanf("%d", &n);
        if(n%2 = = 0)
        printf("The number is even");
        else
        printf("The number is odd");
}
```

if – else ladder

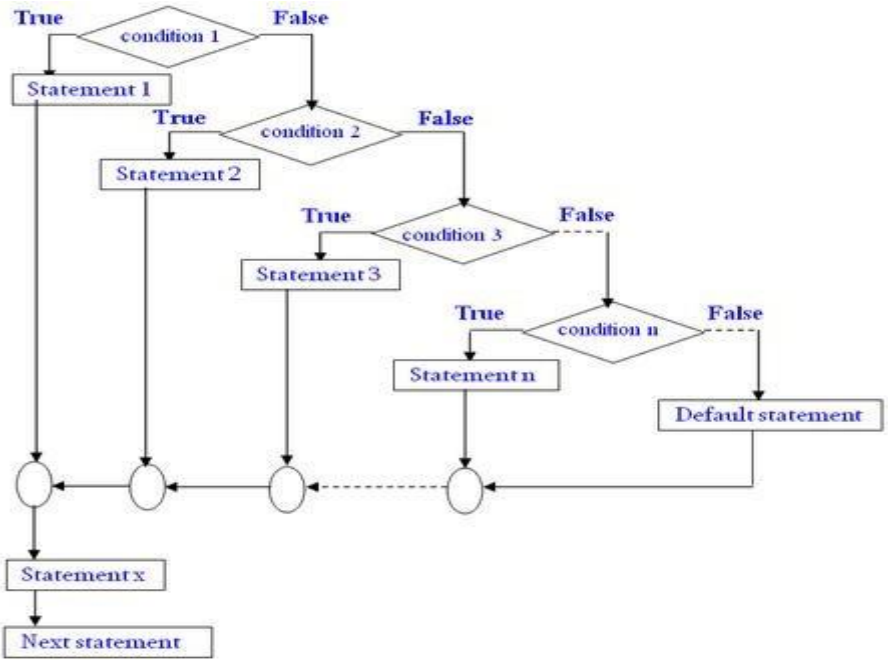
Syntax :

```
if( (condition 1)
{
        true block 1
}
else if(condition 2)
{
        true block 2
}
else if(condition 3)
{
        true block 3
}
 - - -
 - - -
else if(condition n)
{

        true block n
}
else
{
        false block
}
statement x;
```

Working :

```
if condition1 is true

        True block1 + statement x

else if condition2 is true

        True block2 + statement x

else if condition3 is true

        True block3 + statement x

- - -

- - -

else if condition n is true

        True block n + statement x

else

        false block + statement x
```

This is another way of putting ifs together when multipath decisions are involved. A multipath decision is a chain of ifs in which the statement associated with each **else** is an **if.**



Examples :

1. **Input an integer and check whether it is 0 or +ve or –ve**

```
void main()
{
        int n;
        printf("Enter an integer\n");
        scanf("%d", &n);
        if(n==0)
        printf("The number is 0");
        else if(n>0)
        printf("The number is +ve");
        elseprintf("The number is –ve");
}
```

2. **Input a character and print whether it is uppercase letter or lowercase letter or digit or any other character.**

```
void main()
{
        charch;
        printf("Enter a character\n");
        scanf("%c", &ch);
        if(ch>='a' &&ch<='z')
        printf("It is an upper case letter");
        else if(ch>='a' &&ch<='z')
        printf("It is a lower case letter");
        else if(ch>='0' &&ch<='7')
        printf("It is a digit");
        elseprintf("It is a special symbol");
}
```

3. **Input an integer m. If m is 10, square it and print it. If m is 9, read new value for m and print it. If m is 2 or 3, multiply m by 5 and print it. If m is any other value, increment m by 1 and print it.**

```
void main()
{
        int m;
        printf("Enter an integer\n");
        scanf("%d", &m);
        if(m = = 10)
        printf("square is %d", m*m);
        else if (m = = 9)
        {
                printf("Enter new value of m\n");
                scanf("%d", &m);
                printf("m= %d", m);
        }
        else if(m = =2 || m = = 3)
        printf("%d", m*5);
        elseprintf("%d", ++m);
}
```

4. **Input income and find the tax amount according to the following conditions.**

Income	**tax**
<=10000	2%
<=20000	5%
<=30000	7%
<=50000	10%
>50000	15%

```
void main()
{
        float income, tax;
        printf("Enter income\n");
        scanf("%f", &income);
        if(income<=10000)
        tax = 0.02 * income;
        else if(income<=20000)
        tax = 0.05 * income;
        else if(income <=30000)
        tax = 0.07 * income;
        else if (income<=50000)
        tax = 0.1 * income;
        else
        tax = 0.15 * income;
```

```
        printf("tax = %f", tax);
}
```

5. **Input marks and find the grade according to the following conditions**

marks	**grade**
75 – 100	A
60 – 74	B
50 – 59	C
40 – 49	D
<40	E

```
void main()
{
        int marks;
        char grade;
        printf("Enter marks\n");
        scanf("%d", &marks);
        if(marks>=75 && marks <=100)
        grade = 'A';
        else if(marks>=60 && marks <75)
        grade = 'B';
        else if(marks>=50 && marks <60)
        grade = 'C';
        else if(marks>=40 && marks <50)
        grade = 'D';
        else
        grade = 'E';
        printf("The grade is %c", grade);
}
```

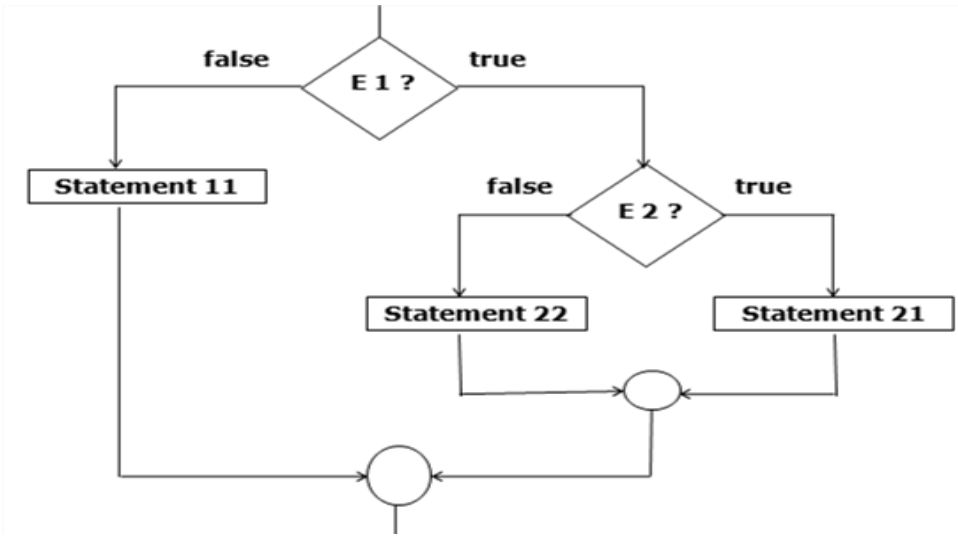
6. **Input 2 operands and an operator (+ , - , * , /) and find corresponding value.**

```
void main()
{
        float a, b, c;
        char op;
        printf("Enter an expression (operand operator operand - Ex : 7 * 6 )\n");
        scanf("%f%c%f", &a, &op,&b);
        if(op = = ' + ')
        {
                c = a + b;
                printf("Sum = %f", c);
        }
        else if(op = = ' - ')
```

```
        {
                c = a - b;
                printf("Difference = %f", c);
        }
        else if(op = = ' * ')
        {
                c = a * b;
                printf("Product = %f", c);
        }
        else if(op = = ' / ' && b!=0)
        {
                c = a / b;
                printf("Quotient = %f", c);
        }
        else
        printf("Invalid operator");
}
```

Nested if – else statement

When a series of decisions are involved we may have to use more than one if else statement in nested form. The nested if else statements are multi decision statements which consists of if else control statements within another if or else section



Use of one if within another if statement is called as nested if statement.

Syntax :

```
if (condition 1)
{
        if(condition 11)
        {
                true block 11
        }
        else
        {
                false block 11
        }
}
else
{
        if(condition 22)
        {
                true block 22
        }
        else
        {
                false block 22
        }
}
statement x ;
```

Working :

If condition 1 is **true**

Check condition 11

If condition 1 is **false**

Check condition 22

Condition 1 is true

If condition 11 is true

True block 11 + statement x

If condition 11 is false

False block 11 + statement x

Condition 1 is false

If condition 22 is true

True block 22 + statement x

If condition 22 is false

False block 22 + statement x

1. Input 3 numbers and find the smallest among them.

```
void main()
{       int a, b, c, s;
        printf("Enter 3 numbers\n");
        scanf("%d %d %d ", &a, &b, &c)
        if(a<b)
        {       if(a<c)
                s = a;
                else
                s = c;
        }
        else
        {
                if(b<c)
                s = b;
                else
                s = c;
        }
        printf("The smallest number is %d", s);
}
```

2. Input 3 numbers and find the largest among them.

```
void main()
{
        int a, b, c, large;
        printf("Enter 3 numbers\n");
        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("The largest number is %d", large);
}
```

Multi way selection - Switch statement

Syntax :

```
switch( expression)
{
        case constant 1 :
                statements
        break;

        case constant 2 :
                statements
        break;

        case constant 3 :
                statements
        break;

        - - -
        - - -
        default :
                false statements
        break;
```

Working :

A switch statement checks the value of an expression against many cases. If any of the case values is matching with the value of the switch expression, the block of Statements under case value are executed. Break is a keyword used along with each case to come out of the switch after the execution of the statements of the Corresponding case. If the switch expression value is not matching with any of the case values, the control is transferred to the default case and the statements under default case are executed.

}Statement x;

Examples

1.
```
int x = 1, y = 12, z =4;
switch(x)
{
    case 0 :
    x = 2;
    y = 3;
    break;
    case 1 :
    x = 4;
    y++;
    break;
    default :
    y = 3;
    z = 7;
}
printf("%d %d %d", x, y, z);
```

Working :
The value of x is compared with case values.
Since x = 1, case 1 is matching and the statements for that case are executed.
Hence x = 4 , y++ --- > y = y+1
x = 4 , y = 13

Output :
4 13 4

2. **Rewrite the following code using switch statement**

```
if (ch = = 'E')
count1++;
else
if(ch = = 'A')
count2 ++;
else
if(ch = = 'I')
count3 ++;
else
printf("Error");
```

Using switch

```
switch ( ch )
{
    case 'E' :
    count1 ++;
    break;
    case 'A' :
    count 2 ++;
    break;
    case 'I' :
    count3 ++;
    break;
    default :
    printf("Error");
}
```

3. **Rewrite the following code using if else statement**

```
int a = 1, b = 0;
switch (++a)
{
    case  1:
    b= a + 10;
    break;
    case  2 :
    b = a * 5;
    break;
    case 3 :
    b = a – 2;
```

Using if else

```
int a = 1, b =0;
++a;
if (a = = 1)
b= a + 10;
else if (a = = 2)
 b = a * 5;
else if (a = = 3)
 b = a – 2;
else
printf("Invalid value");
printf(%d %d", a, b);
```

```
        break;
        default :
        printf("Invalid value");
}
printf(%d %d", a, b);
```

Input an integer between 1 and 7 and print the corresponding day of the week. If it is not between 1 and 7, display suitable message

```
void main()
{       int a;
                printf("Enter an integer(1 - 7)\n");
                scanf("%d", &a);
                switch(a)
                {
                case 1 :
                        printf("Sunday");
                        break;
                case 2 :
                        printf("Monday");
                        break;
                case 3 :
                        printf("Tuesday");
                        break;
                case 4 :
                        printf("Wednesday");
                        break;
                case 5 :
                        printf("Thursday");
                        break;
                case 6 :
                        printf("Friday");
                        break;
                case 7 :
                        printf("Saturday");
                        break;
                default :
                        printf("Invalid number");
                }
}
```

Simulate a calculator using switch statement.

```
void main()
{
float a, b, c;
                        char op ;
                        printf("Enter an expression (a + b)\n");
                        scanf("%f %c %f" , &a, &op, &b);
                        switch(op)
                        {
                        case '+' :
```

```
                c = a + b;
                printf("sum = %f ", c);
                break;
        case '-' :
                c = a - b;
                printf("Difference = %f ", c);
                break;
        case '*' :
                c = a * b;
                printf("Product = %f ", c);
                break;
        case '/' :
                if(b==0)
                {
                        printf(" b cannot be 0");
                        exit(0);
                }
                c = a / b;
                printf("Quotient = %f ", c);
                break;
        default :
                printf("Invalid operator");
        }
}
```

Rules of switch statement

1. Expression that follows the keyword switch must be an integral type (int / char).
2. Constant values can be only int / char.
3. No two case labels have same values.
4. It cannot be used for range of values.
5. Values cannot be compared with relational or logical operators. It works only for = = operator.
6. default case is optional.

Examples of switch statement without break for case values

Give the output for the following

```
charch;
switch(ch)
{
```

Working :

If ch is 'A',

Case 'A' is matching and the statements for case 'A' are executed. Since case 'A' is not ended by break statement, the statements under the next case are also evaluated. Hence all the case values are executed till a break statement is encountered.

The **output** would be

Grade A

Grade B

If ch is 'H',

default case is evaluated.

The **output** would be

Grade F

If ch is 'C',

case 'C' and default case are evaluated.

The **output** would be

Grade C

Grade F

```
        ase 'A':
        printf("Grade A\n"); case 'B' :
        printf("Grade B\n"); break;
        case 'C':
        printf("Grade C\n"); default :
        printf("Grade 'F\n");
}
```

rogramming for Problem Solving - 18CPS13/23

Give the output for the following

```
int x = 0, y = 1, z = 1;
switch (x)
{
        case 0 :
        x = 2;
        y = 3;

        case 1 :
        x = 4;
        break;
        default :
        x = 1;
        y = 3;
}
printf("%d %d %d", x, y , z);
```

Working

x is 0 and matching with case 0.
Hence x = 2 , y =3
Since there is no break statement after case 0,
Case 1 statements are also executed.
Hence x = 4 and y remains as 3
So
X = 4 , y = 3 and z = 1

Output

4 3 1

Give the output for the following

```
int x = 1, y = 0, z = 1;
switch (x)
{       case 0 :
        x++ ;
        y++ ;
        break;
        case 1 :
        x+= 2;
        z++;
        case 2 :
        z * = 3 ; y--;
        break;
        default :
        x = x + 10;
        y * = 2;
```

Working

x is 1 and matching with case 1.
Hence
x+ = 2 ----------> x = x + 2 -----> x = 1 + 2 = 3
z++ ----------> z = z + 1 -----> = 1 + 1 = 2
Since there is no break statement after case 1,
Case 2 statements are also executed.
Hence
z = z * 3 ----> z = 2 * 3 = 6
y - - ----------> y = y – 1 -----> y = 0 – 1 = -1

So
x = 3 , y = -1 and z =6

Output

3 -1 6

```
}
printf("%d %d %d", x, y , z);
```

Rewrite the following code using switch statement

```
if (ch = = 'E' | | ch = = 'e') count1++;
else if(ch = = 'A' | | ch = = 'a') count2 ++;
else if(ch = = 'I' | | ch = = 'i') count3 ++;
else
printf("Error");
```

Using switch

```
switch ( ch )
{
case 'E' : case 'e' : count1 ++;
break;
case 'A' : case 'a': count 2 ++; break;
case 'I' : case 'i': count3 ++; break;
default :
printf("Error");
}
```

Differences between if else ladder and switch statement

S.N	If – else ladder	Switch statement
1	Each test condition is checked for true value. If true, statement block is executed.	The control is transferred to the particular case value. It does not check each and every case value.
2	Expression that follows the keyword ifcan be of any data type.	Expression that follows the keyword switch must be an integral type (int / char).
3	Values used for comparison can be of any data type.	Constant values given in **case**can be only int / char.
4	If else can be used for a range of values (using relational and logical operators) Ex : if(a>=7 && a<=10)	switch cannot be used for range of values.
5	Values can be compared with any type of operator.	Values cannot be compared with relational or logical operators. It works only for = = operator.

Repetition (Looping)

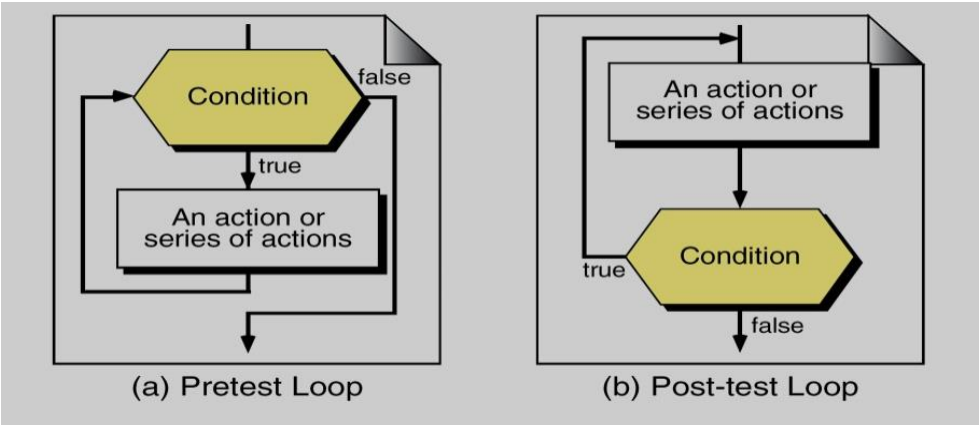
Definition

To repeat the instruction set many number of times, looping statements are used.

Types of loops

- Pretest loop (Entry controlled loop)
- Posttest loop (Exit controlled loop)

Pre test loop	Post test loop
In each iteration Control expression tested • If true, loop instructions are executed • If false, loop terminates.	**In each iteration** The loop instructions are executed, then the control expression is tested. • If true, new iteration • If false the loop terminates.
Minimum no. of iterations is **0**.	Minimum no. of iterations is **1**.
Pretest loops • while loop • for loop	**Posttest loop** • do – while loop
Also called as **entry controlled** loop.	Also called as **exit controlled** loop.



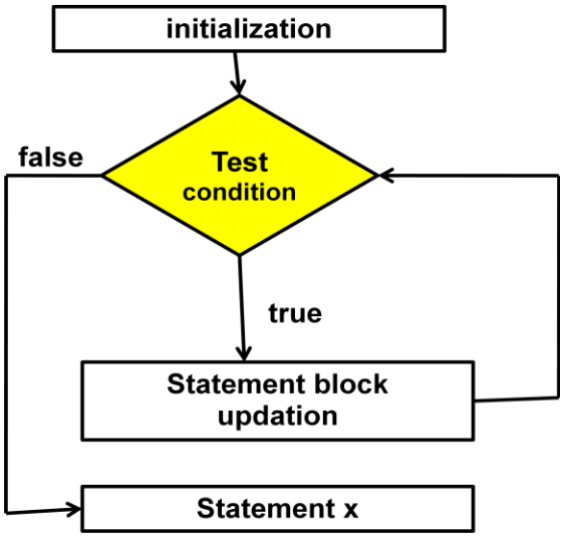
Loops in C

- while - pre test
- for - pre test
- do while - post test

While loop

Syntax :

Initialization
while(test condition)
{
statement block
updation
}
Statement x;



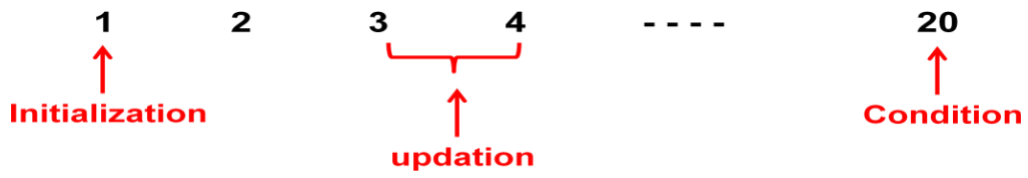
Working :

1. Initialization is done with one or more variables.
2. The test condition is checked.
   If **true**
   - Statement block and updations are executed
   - The control is transferred again to test condition, goto step 2

   If **false**
   Goto step 3
3. Statement x is executed

Examples

1. **Write a program to generate the numbers from 1 to 20**
   while has 3 parts
   - Initialization (start value)
   - Test condition (final value)
   - Updation (common difference)




```
void main()
{
        int a = 1;                      //  initialization
        while(a<=20)                    //  condition
```

```
        {
                printf("%d     ", a);
                a = a + 1;                      //  updation
        }
}
```

2. **Write a program to generate the following series**

   50 51 52 - - - 100

```
void main()
{
        int a = 50;
        while(a<=100)
        printf("%d     ", a++);
}
```

3. **Write a program to generate the following**

   2 4 6 8 - - - - 200

```
void main()
{
        int a = 2;
        while(a<=200)
        {
                printf("%d     ", a);
                a = a + 2;
        }
}
```

4. **Write a program to generate the following**

   100 99 98 97 - - - - 1

```
void main()
{
        int a = 100;
        while(a>=1)
        {
                printf("%d     ", a);
                a = a - 1;
        }
}
```

5. **Write a program to generate the following**

   10 20 30 40 - - - - 100

```
void main()
{
        int a = 10;
```

```
        while(a<=100)
        {
                printf("%d     ", a);
                a = a + 10;
        }
}
```

6. Write a program to find the sum of the following

Sum = 1 + 2 + 3 + - - - + 100

```
void main()
{
        int a = 1, sum = 0;
        while(a<=100)
        {
                sum = sum + a;
                a = a + 1;
        }
        printf("sum = %d     ", sum);
}
```

7. Find the output for the following code

```
void main()
{
        int a = 2, sum = 0;
        while(a<=6)
        {
                sum = sum + a;
                a = a + 1;
                printf(%d    %d\n", sum, a);
        }
        printf("sum = %d\n", sum);
        printf("a = %d", a);
}
```

Output

2	3
5	4
9	5
14	6
20	7

sum = 20
a = 7

8. Write a program to find the sum of the following

Sum = 1 + 3 + 5 +-------- + 101

```
void main()
{
        int a = 1, sum = 0;
        while(a<=101)
        {
                sum = sum + a;
                a = a + 2;
```

```
        }
        printf("sum = %d      ", sum);
}
```

9. **Write a program to find the sum of the following**

   Sum = 1 + 2 + 3 + - - - + n

```
void main()
{
        int a = 1, sum = 0, n;
        printf("Enter n\n");
        scanf("%d", &n);
        while(a<=n)
        {
                sum = sum + a;
                a = a + 1;
        }
        printf("sum = %d      ", sum);
}
```

10. **Write a program to find the sum of the following**

    Sum = 1 + 1/2 + 1/3 + - - - + 1/50

```
void main()
{
        float a = 1, sum = 0;
        while(a<=50)
        {
                sum = sum + 1/a;
                a = a + 1;
        }
        printf("sum = %d      ", sum);
}
```

11. **Write a program to find the product of the following**

    result = 1 * 2 * 3 * 4---------- * 10

```
void main()
{
        longint a = 1, result = 1;
        while(a<=10)
        {
                result = result * a;
                a = a + 1;
        }
        printf("result = %ld      ", result);
}
```

12. Write a program to find the factorial of a number n.

```
void main()
{
        int n, a = 1;
        longint fact = 1;
        printf("Enter a number \n");
        scanf("%d", &n);
        if(n<0)
        {
                printf("Invalid value");
                exit(0);
        }
        while(a<=n)
        {
                fact = fact * a;
                a = a + 1;
        }
        printf("factorial = %ld      ", fact);
}
```

13. **Write a program to input a binary number and find its equivalent decimal.**

```
void main()
{
        intnum, dec=0, rem, i=0;
        printf("Enter a binary number \n");
        scanf("%d", &num);
        while (num !=0)
        {
                rem = num % 10;
                if(rem!=0 && rem!=1)
                {
                        printf("Not a binary number");
                        exit(0);
                }
                dec = dec + rem * pow(2,i);
                i++;
                num = num/10;
        }
        printf("The decimal equivalent is %d ", dec);
}
```

14. **Write a program to input a decimal number and find its equivalent binary.**

```
void main()
{
        intnum, bin=0, rem, i=0;
        printf("Enter a decimal number \n");
        scanf("%d", &num);
        while (num !=0)
```

```
        {       rem = num % 2;
                bin = bin + rem * pow(10,i);
                i++;
                num = num/2;
        }
        printf("The binary equivalent is %d ", bin);
}
```

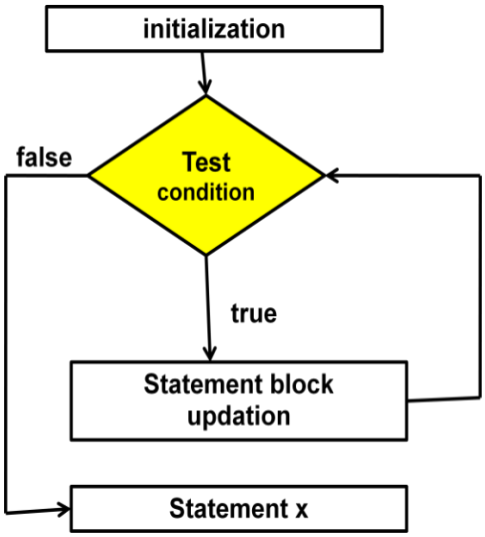
15. **Write a program to input an integer and add the digits.**
    Ex : n = 3512 sum = 3 + 5 + 1 + 2 = 11

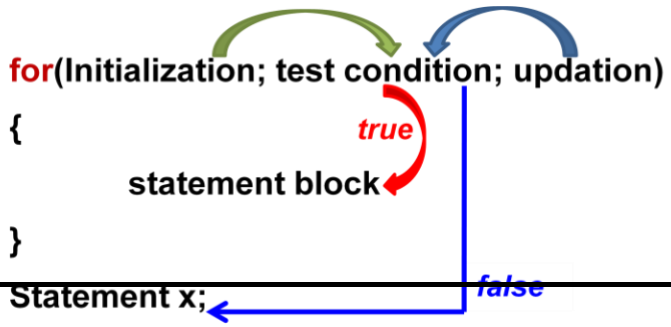
```
void main()
{       intnum, sum=0, rem;
        printf("Enter an integer\n");
        scanf("%d", &num);
        while (num !=0)
        {       rem = num % 10;
                sum = sum + rem;
                num = num / 10;
        }
        printf("Sum of digits = %d", sum);
}
```

for loop

Syntax :

```
for(Initialization; test condition; updation)
{
        statement block
}
Statement x;
```





Working :

1. *Initilaization only once* at the beginning of for loop.
2. The test condition is checked.
   If **true**
   - Statement block and updations are executed
   - The control is transferred again to test condition, goto step 2

   If **false**
   Goto step 3

while loop	for loop
Preferred when the exact number of iterations are not known.	Preferred when exact number of iterations are known.

3. Statement x is executed

Difference between for and while loops

Examples

1. **Write a for loop to generate the following series**
   50 51 52 - - - 100

```
void main()
{
        int a;
        for(a = 50; a<=100 ; a++)
        printf("%d    ", a);
}
```

2. **Write a for loop to generate the following**
   2 4 6 8 - - - - 200

```
void main()
{
        int a;
        for(a=2 ; a<=200; a = a+2)
        printf("%d    ", a);
}
```

3. **Write a for loop to generate the following**
   3 6 9 - - - - 300

```
void main()
```

```
{
        inti=3;
        for(; i<=300; i = i+3)
        printf("%d      ", i);
}
```

4. **Write a for loop to find the sum of the following**
   Sum = 1 + 2 + 3 + - - - + 100

```
void main()
{
        int a, sum ;
        for(a=1, sum=0;a<=100;++a)
        sum = sum + a;
        printf("sum = %d      ", sum);
}
```

Statement x

5. **Write a for loop to find the sum of the series**
   Sum = $1^2 + 2^2 + 3^2 + 4^2$+-------- + n^2

```
void main()
{
        inti,n, sum=0;
        printf("Enter no. of terms\n");
        scanf("%d", &n);
        for(i=1 ; i<=n; i++)
        sum+ = i*i;
        printf("Sum=%d      ", sum);
}
```

6. **Write a for loop to find the sum of even numbers and odd numbers from 1 to 100.**

```
void main()
{
        inti, odd, even;
        for(i=1, odd=0, even=0 ; i<=100; i++)
        {
                if(i%2==1) odd+=i;
                else even+=i;
        }
        printf("Odd=%d    Even=%d", odd, even);
}
```

7. **Write a for loop to find the factorial of a number n.**

```
void main()
{
        int n, i;
        longint fact;
        printf("Enter a number \n");
        scanf("%d", &n);
```

```
        if(n<0)
        {
                printf("Invalid value");
                exit(0);
        }
        for(i=1; fact=1; i<=n;i++)
        fact* = i;
        printf("factorial = %ld      ", fact);
}
```

8. Write a for loop to find the multiplication table of a number.

```
void main()
{
        inti, n, result;
        printf("Enter a number \n");
        scanf("%d", &n);

        for(i=1; i<=10;i++)
        {
                result = n * i;
                printf("%d * %d = %d\n", n, i, result);
        }
}
```

9. Write a for loop generate the fibonacci series

0 1 1 2 3 5 8---------- n

```
 void main()
{       inti, n, first=0, second=1, third;
        printf("Enter the limit for fibonacci series\n");
        scanf("%d", &n);
        printf("Fibonacci series\n");
        printf("0                1\t");
        for(i=3; i<=n;i++)
        {
                third = first + second;
                printf("%d\t",  third);
                first = second;
                second = third;
        }
  }
```

Give the output for the following

1.
```
int i;
for(i=0;i<5;i++)
printf("%d\n", i);
printf("%d", i);
```

Output :
0
1
2
3
4
5

2.
```
int i;
for(i=1;i<=10;i++)
printf("%d    ", i++);
```

Output :
1 3 5 7 9

3.
```
int i;
for(i=1;i<=5;i++) ;
printf("%d\n", i);
printf("%d", i);
```

Output :
6
6

4.
```
int i;
for(i=6;i>=10;i++)
printf("%d    ", i);
```

Output :
No output
as the condition is false for the 1st iteration itself.

Nested for loop

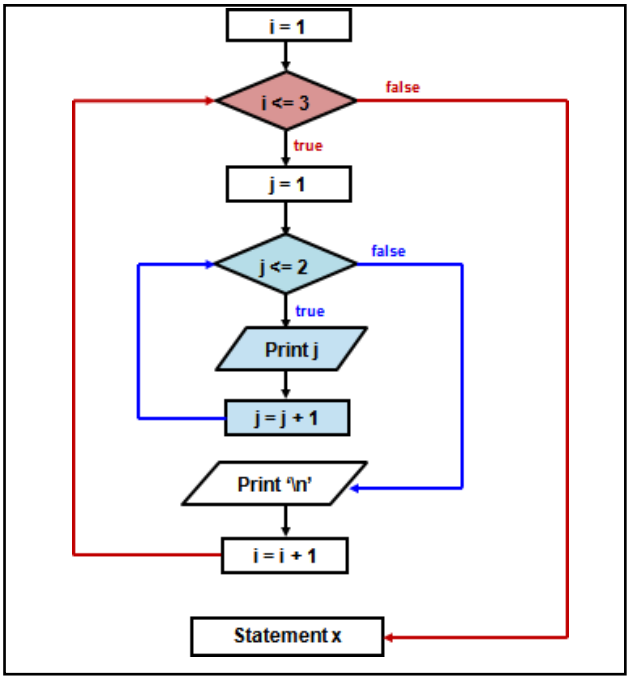
Use of one for loop within another is called as nesting of for loop.

Examples

```
int i , j;
for(i=1;i<=3;i++)  ---> No. of rows
{
    for(j=1;j<=2;j++)  ---> No. of columns
    printf("%d\t", j) ;
    printf("\n");
}
```

Statement x

i for loop is executed thrice (3 times)------> from i = 1 to i = 3
j for loop is a nested loop. j for loop is executed twice (2 times) for each i for loop.
So, total no. of iterations = 3 * 2 = 6 times.



Working

1) i = 1 1 < = 3 True
   a) j = 1 1 < = 2 True
      print j
      j = j + 1

   b) j = 2 2 < = 2 True
      print j
      j = j + 1

   c) j = 3 3 < = 2 False
      print '\n'
      move to next iteration of i loop.

2) i = 2 2 < = 3 True
   a) j = 1 1 < = 2 True
      print j
      j = j + 1

   b) j = 2 2 < = 2 True
      print j
      j = j + 1

   c) j = 3 3 < = 2 False
      print '\n'
      move to next iteration of i loop.

3) i = 3 3 < = 3 True
   a) j = 1 1 < = 2 True
      print j
      j = j + 1

   b) j = 2 2 < = 2 True
      print j
      j = j + 1

4) i = 4 4 < = 3 False
comes out of i loop

Output

1	2
1	2
1	2

More examples on nested for loop

1.
```
inti , j;
for(i=1;i<=3;i++)
{
        for(j=1;j<=2;j++)
        printf("%d\t", i) ;
        printf("\n");
}
```

Output

1	1
2	2
3	3

2.
```
inti , j , a = 10;
for(i=1;i<=4;i++)
{
        for(j=1;j<=3;j++)
        printf("%d\t", a++);
        printf("\n");
}
```

Output

10	11	12
13	14	15
16	17	18
19	20	21

3.
```
inti , j ;
for(i=1;i<=4;i++)
{
        for(j=1;j< = i; j++)
        printf("%d\t", j);
        printf("\n");
}
```

Output

1			
1	2		
1	2	3	
1	2	3	4

4.

Write a nested for loopto generate the following pattern

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

```
inti , j ;
for(i=5;i>=1;i--)
{
        for(j=1;j< = i; j++)
        printf("%d\t", j);
        printf("\n");
}
```

```
inti , j ;
for(i=1;i<=4;i++)
{
        for(j=1;j< = i; j++)
        printf("*\t");
        printf("\n");
}
```

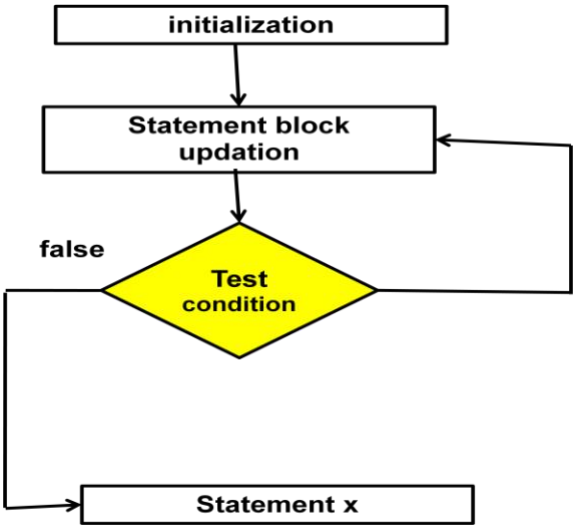
Output

```
1       2       3       4       5
1       2       3       4
1       2       3
1       2
1
```

do - while loop

Syntax :

```
Initialization
do
{
        statement block
        updation
} while(test condition) ;
Statement x;
```



Example

```
inti = 1;
do
{
        printf("%d ", i);
        i++;
}while(i<=10);
```

Output

1 2 3 4 5 6 7 8 9 10

Comparison between while and do-while

```
inti = 1 ;
while(i>10)
{
        printf("%d", i);
        i++;
}
printf("\nThank you");
```

```
inti = 1 ;
do
{
        printf("%d", i);
        i++;
} while(i>10) ;
printf("\nThank you");
```

Output
Thank you

S.N	While	Do - while
1	Pretest (entry controlled) loop	Posttest (exit controlled) loop
2	Condition is checked first and if is true, statement block is executed.	Statement block is executed first. Then the condition is checked and if it is true, loop continues.
3	Minimum no. of iterations is 0.	Minimum no. of iterations is 1.
4	Syntax	Syntax
5	Flowchart	Flowchart
6	Example with output	Example with output

Looping Applications

- **Summation**
  - to find the sum of numbers by taking initial value of the result as 0.
- **Powers**
  - to find the product of numbers by taking initial value of the result as 1.
- **Smallest and largest numbers**

Other statements related to looping

- break
- continue
- goto

break

- Used to break (exit) the loop – without executing statements which are after break statement.
- Can be used with switch, for, while and do – while statements.

Example for break statement

```
inti;
float a, b, c;
for(i=1;i<=4;i++)
{
        printf("Enter a and b\n");
        scanf("%f %f", &a, &b);
        if(b==0)
        {
                printf("Invalid b\n");
                break;
        }
        else
        {
                c = a/b;
                printf("c = %f\n",c);
        }
}
printf("\nThank you");
```

Output

```
Enter a and b
5 2
c = 2.5
Enter a and b
15  3
c = 5
Enter a and b
3  0
Invalid b
Thank you
```

continue

- Used to continue with the next iteration of the loop without executing statements which are after continue.
- Can be used with for, while and do – while statements.

Example for continue statement

```
inti;
float a, b, c;

for(i=1;i<=4;i++)
{
        printf("Enter a and b\n");
        scanf("%f %f", &a, &b);
        if(b==0)
        {
                printf("Invalid b\n");
                continue;
        }
```

Output

```
Enter a and b
5 2
c = 2.5
Enter a and b
15 3
c = 5
Enter a and b
3 0
Invalid b
Enter a and b
15 4
```

```
        c = a/b;
```

goto

Used to perform a transfer of control from one statement to another in a program.

Example for goto statement

```
int a=1;
st :
printf("%d  ", a);
a++;
if(a<=5)
gotost;
printf("\nThank you");
```

Output :

1 2 3 4 5
Thank you

Rewrite the while code using goto statement

```
inti = 50, sum=0;
do
{
        sum+=i;
        i = i – 5;
} while(i>=0);
printf("sum=%d", sum);
```

Using goto statement

```
inti = 50, sum=0;
st:
sum+=i;
i = i – 5;
if(i>=0)
gotost;
printf("sum=%d", sum);
```

1. Write a program to generate prime numbers from 1 to n.

```
void main()
{
        int n, i, j, c;
        printf("Enter Range To Print Prime Numbers\n");
        scanf("%d", &n);
        printf("Prime Numbers are\n");
        for(i=1;i<=n;i++)
        {
                c=0;
```

```
                for(j=1;j<=i;j++)
                {
                        if(i%j==0)
                        c++;
                }
                if(c==2)
                printf("%d ",i);
        }
}
```

2. **For the given value of x and n, write a C program to evaluate the series**
   $y = 1 + x + x^2 + x^3 +$---------$+ x^n$

```
#include<stdio.h>
#include<math.h>
void main()
{       float x, y;
        inti, n;
        printf("Enter x and n values\n");
        scanf("%f%d", &x, &n);
        for(i=0,y=0;i<=n;i++)
        y = y + pow(x,i);
        printf("y = %f ", y);
}
```

3. **Write a C program to find GCD of two non – zero integer numbers. If the first number is less than the second, then the program must exchange the two numbers before computing the GCD.**

```
#include<stdio.h>
void main( )
{
        int num1,num2,num,den,rem,gcd;
        printf("Enter two non-zero integers\n");
        scanf("%d%d",&num1,&num2);
        if(num1!=num2)
        {
                if(num1>num2)
                {       num=num1;
                        den=num2;
                }
                else
                {       num = num2;
                        den=num1;
                }
                rem=num%den;
                while(rem!=0)
                {
                        num=den;
                        den=rem;
                        rem=num%den;
                }
                gcd=den;
```

```
        }
        else
        gcd=num1;
        printf("\nGCDof given integers = %d", gcd);
}
```

4. **Write a C program that will read the value of x and evaluate the following.**
   y = 1 + x when n = 1
   y = 1 + x^n when n = 2
   y = 1 + nx otherwise
   using (i) else if statement (ii) switch statement

else if statement

```
#include<stdio.h>
#include<math.h>
void main()
{
        float x, y;
        int n;
        printf("Enter values of x and n\n");
        scanf("%f %d", &x, &n);
        if(n==1)
        y = 1 + x;
        else if(n==2)
        y = 1 + pow(x, n);
        else
        y = 1 + n*x;
        printf("y = %f", y);
}
```

switch statement

```
#include<stdio.h>
#include<math.h>
void main()
{
        float x, y;
        int n;
        printf("Enter values of x and n\n");
        scanf("%f %d", &x, &n);
        switch(n)
        {
                case 1 :
                y = 1 + x;
                break;
                case 2 :
                y = 1 + pow(x, n);
                break;
                default :
                y = 1 + n*x;
        }
        printf("y = %f", y);
}
```

5. **Write a program to compute the value of Euler's number e , that is used as the base of natural logarithm. Use the following formula. Use while statement.**

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \ldots\ldots\ldots\ldots \text{ upto acc} = 0.0001.$$

```
#include<stdio.h>
void main()
{
        float e=1.0 , acc = 0.0001, term=1.0;
        inti = 1, den=1;
        while(term>acc)
        {
                term = 1/den;
                e+ = term ;
                i++;
                den = den * i ;
```

```
        }
        printf("e = %f", e);
}
```

6. Write a C program using switch statement to perform the following operations between two variables. The operations are 1 – addition, 2 – subtraction, 3 – multiplication , 4 – division. Print error message for default statement.

```
#include<stdio.h>
#include<math.h>
void main()
{       float a, b, c ;
        int n;
        printf("Enter 2 numbers\n");
        scanf("%f%f", &a, &b);
        printf("1. addition\n2. subtraction\n3. multiplication\n4. division\n");
        printf("enter choice\n");
        scanf("%d", &n);
        switch(n)
        {
                case 1 :
                c = a + b;
                printf("sum = %f", c);
                break;
                case 2 :
                c = a - b;
                printf("difference = %f", c);
                break;
                case 3 :
                c = a * b;
                printf("product = %f", c);
                break;
                case 4 :
                if(b==0)
                exit(0); c
                = a / b;
                printf("quotient = %f", c);
                break;
                default :
                printf("Invalid choice");
        }
}
```

Question paper questions

1. What is two way selection statement? Explain if, it-else, nested if else and cascaded if else with examples and syntax
2. Write a program that takes three coefficients (a, b, and c) of a quadratic equation: (ax2 + bx + c) as input and compute all possible roots and print them with appropriate messages.
3. Explain switch statement with an example.
4. What is a loop? Explain the different loops in C language.
5. Show how break and continue statements are used in a C program, with example.
6. What are different types of conditional decision making statements? Explain each with examples.
7. Write a C program to simulate simple calculator that performs arithmetic operations using switch statement. Error message should be displayed, if any attempt is made to divide by zero.
8. List four differences between while loop and do-while loop along with syntax and example.
9. Design and develop a C program to reverse a given four digit integer number and check whether it is a palindrome or not.
10. Explain the, syntax of do-while statement. Write a C program to find the factorial of number using do-while, where the number n is entered by user.
11. What is two way selection statement? Explain if, if else, and cascaded if-else with examples.
12. Write a C program that takes from user an arithmetic operator ('+', '-','*' or '/') and two operands. Perform the corresponding arithmetic operation on the operands using switch statement
13. Explain the ELSE - IF ladder with syntax & example.
14. List the types of loops. Explain the working of any one type of loop with syntax and example.
15. Write a program to read a year as input and find whether it is a LEAP YEAR or not.
16. Explain SWITCH statement with syntax and example.
17. Differentiate between while and do - WHILE loops.
18. Write a program to find reverse of a number and check whether it is a palindrome or not.