Course Name:	**Structured Programming Methodology**	**Semester:**	**I**
Date of Performance:	**23/9/2025**	**DIV/ Batch No:**	**A1**
Student Name:	**Arav Arun**	**Roll No:**	**16010125013**

Experiment No: 3

Title: Write C++ programs to demonstrate the use of looping control structures

Aim and Objective of the Experiment:

Write a menu-driven program for the following option

a. Print Numbers 1 to 10 using for loop

b. Find the factorial of a number using a while loop.

c. Asks the user to enter numbers and calculates their sum, using a do…while loop.

COs to be achieved:

CO: Apply basic concepts of C++ programming for problem-solving. (CO1 and CO2).

Theory:

Loops in programming are used to repeat a block of code until the specified condition is met. A loop statement allows programmers to execute a statement or group of statements multiple times without code repetition.

There are mainly two types of loops in C++:

- Entry Controlled loops: In Entry Controlled loops, the test condition is checked before entering the main body of the loop. **For Loop and While Loop are Entry-controlled loops**.
- Exit Controlled loops: In Exit controlled loops the test condition is evaluated at the end of the loop body. The loop body will execute at least once, irrespective of whether the condition is true or false. **do-while Loop is Exit Controlled loop**.

for Loop

for loop is a repetition control structure that allows programmers to write a loop that will be executed a specific number of times. for loop enables programmers to perform n steps together in a single line.

Syntax:

```
for (initialize expression; test expression; update expression)
{
    //
    // body of for loop
    //
```

```
      }
```

For Example:-

```
for(int i = 0; i < n; ++i)
{
   printf("Body of for loop which will execute till n");
}
```

While Loop

While loop does not depend upon the number of iterations. In the for loop the number of iterations was previously known to us but in the While loop, the execution is terminated on the basis of the test condition. If the test condition becomes false then it will break from the while loop else body will be executed.

```
while (test_expression)
{
    // body of the while loop

    update_expression;
}
```

do-while Loop

The do-while loop is similar to a while loop but the only difference lies in the do-while loop test condition which is tested at the end of the body. In the do-while loop, the loop body will execute at least once, irrespective of the test condition.

```
do
{
    // body of do-while loop

    update_expression;

} while (test_expression);
```

Problem Statements:

a. Print numbers from a to b using for loop

b. Find the factorial of a number using a while loop.

c. Ask the user to enter non-negative integers and calculate their sum. The program should keep asking until the user enters -1, using a do…while loop.

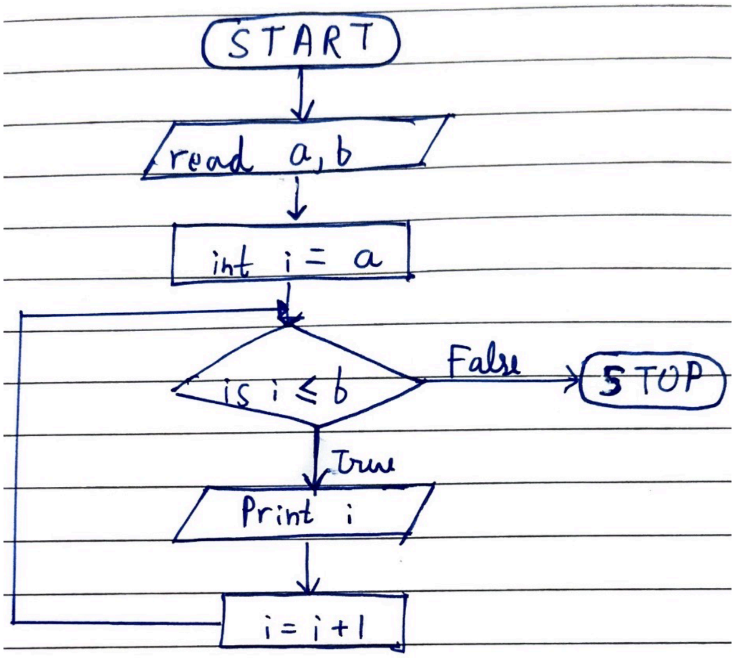
Sample run

```
Enter numbers to add (enter 0 to stop):
5
10
3
0
```

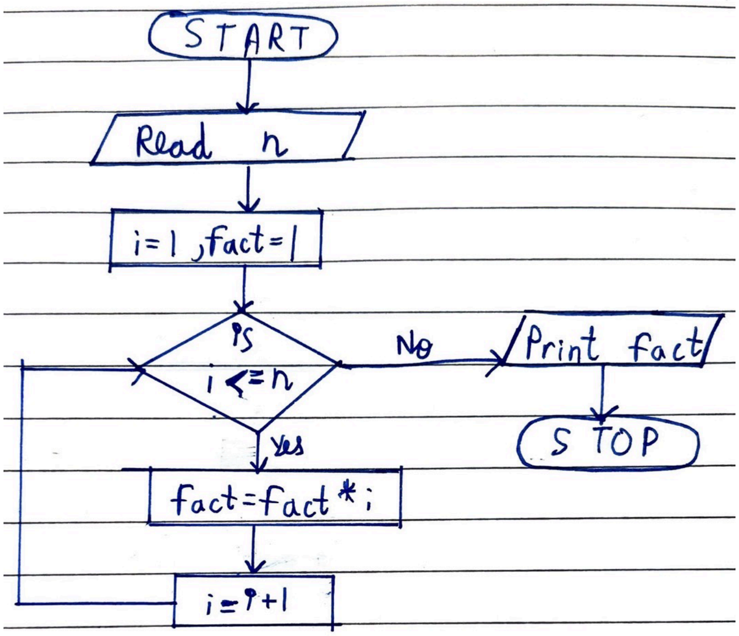
```
Sum of entered numbers = 18
```

Flow chart:

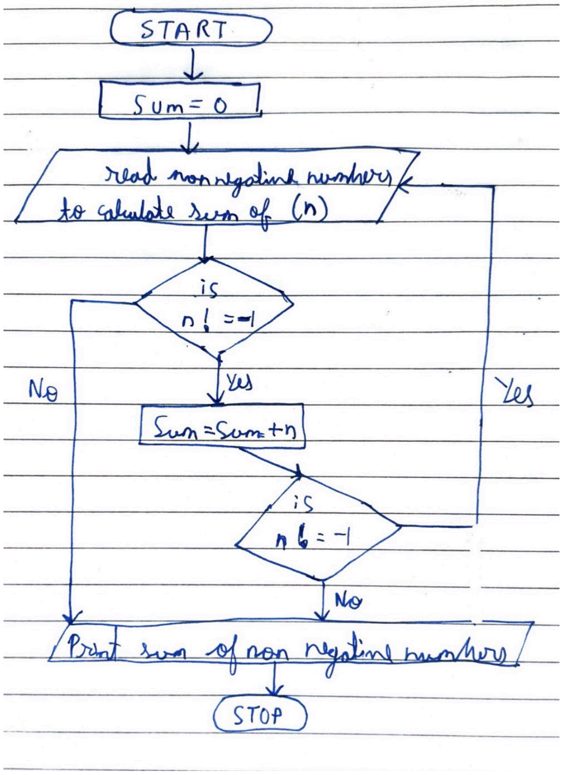
a)



b)



c)


START
Sum=0
read non negative numbers to calculate sum of (n)
is n != -1
Yes
No
Sum=Sum+n
is n 6 = -1
Yes
No
Print sum of non negative numbers
STOP

Code :

Program a :

```
#include <iostream>
using namespace std;

int main()
{
    int a=0,b=0;
    cout<<"Enter the number 'a' from which you want to start printing : "<<endl;
    cin>>a;
    cout<<"Enter the number 'b' at which you want to stop printing : "<<endl;
    cin>>b;
    cout<<"Numbers from "<<a<<" to "<<b<<" are as follows : "<<endl;
    for(int i =a;i<=b;i++)
    {
        cout<<i<<endl;
    }
    return 0;
}
```

Program b :

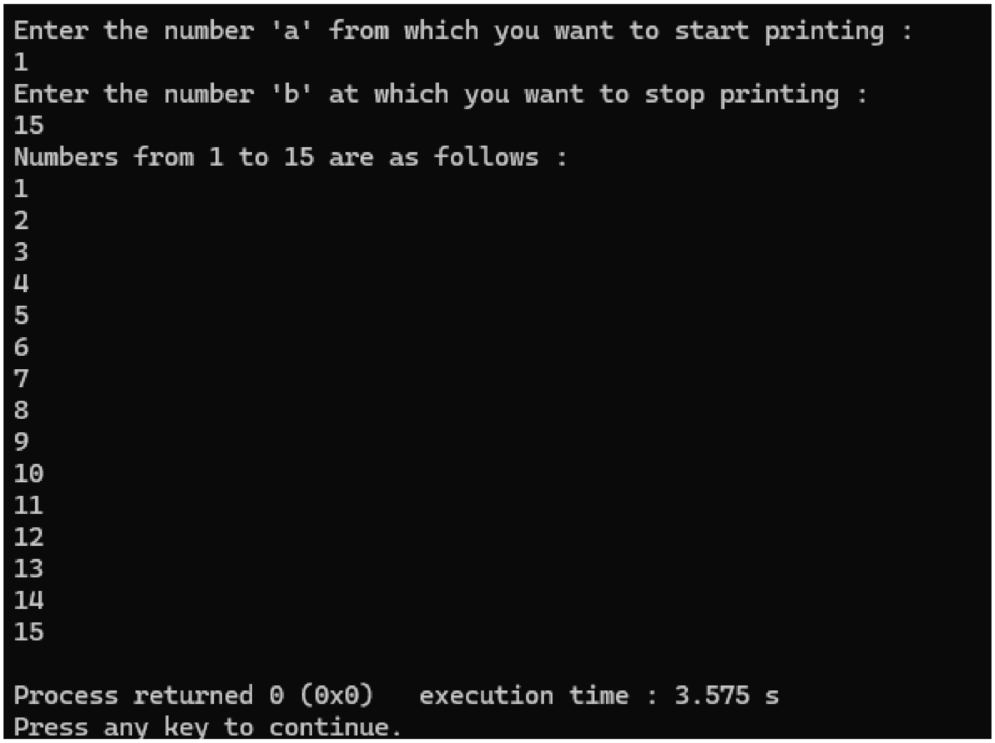
```
#include <iostream>
using namespace std;

int main()
{
    int n = 0,i=1,fact=1;
    cout <<"Enter number to find factorial of :"<<endl;
    cin>>n;

    while(i<=n)
    {
        fact=fact*i;
        i++;
    }
    cout<<"The factorial of "<<n<<" is "<<fact;

    return 0;
}
```

Program c :

```
#include <iostream>
using namespace std;

int main()
{
    int n=0,sum=0;
    cout << "Enter non negative numbers to calculate sum of :"<<endl;
    do
    {
       cin>>n;
       if(n!=-1)
       {
           sum=sum+n;
       }
    }while (n!=-1);
    cout<<"Sum of non negative numbers entered is : "<<sum;
    return 0;
}
```

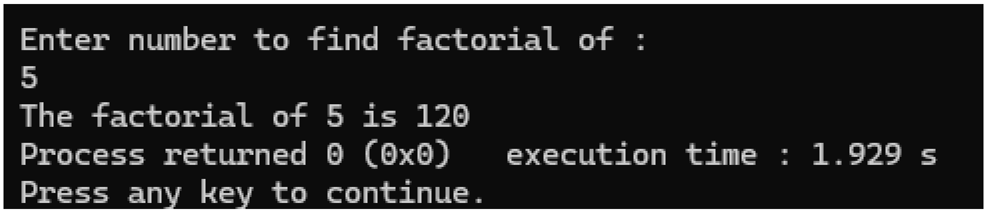
Output:

Output for Program a :



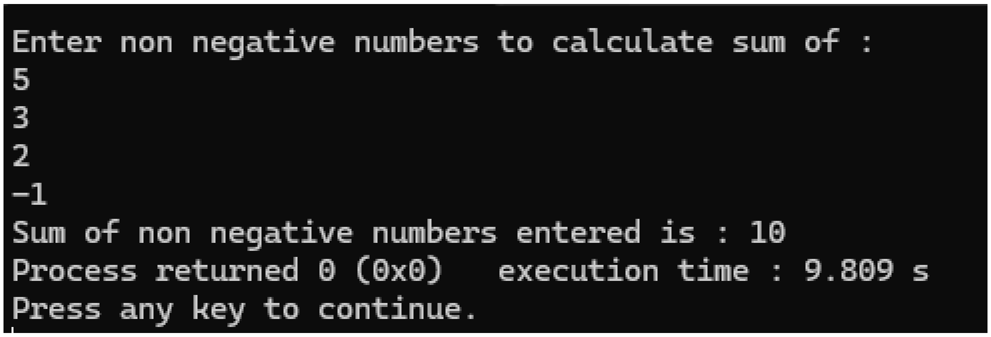
```
Enter the number 'a' from which you want to start printing :
1
Enter the number 'b' at which you want to stop printing :
15
Numbers from 1 to 15 are as follows :
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Process returned 0 (0x0)   execution time : 3.575 s
Press any key to continue.
```

Output for Program b :



```
Enter number to find factorial of :
5
The factorial of 5 is 120
Process returned 0 (0x0)   execution time : 1.929 s
Press any key to continue.
```

Output for Program c :



```
Enter non negative numbers to calculate sum of :
5
3
2
-1
Sum of non negative numbers entered is : 10
Process returned 0 (0x0)   execution time : 9.809 s
Press any key to continue.
```

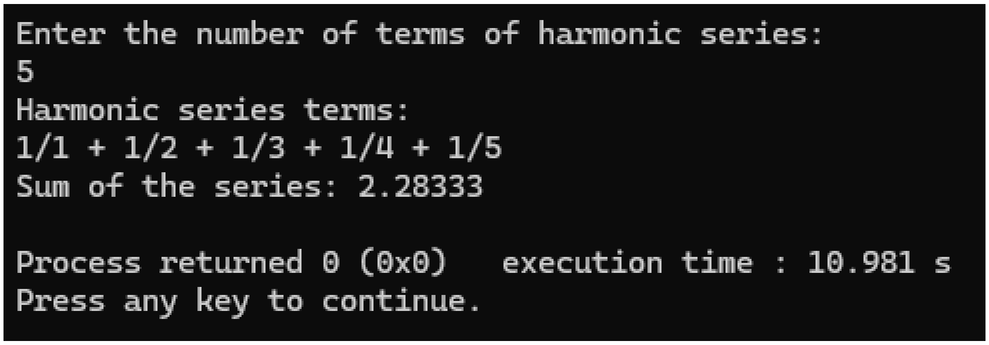
Post Lab Subjective/Objective type Questions:

1.Write a program in C++ to display the n terms of a harmonic series and their sum.
1 + 1/2 + 1/3 + 1/4 + 1/5 ... 1/n terms

Sol:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    double sum = 0.0;

    cout << "Enter the number of terms of harmonic series: "<<endl;
    cin >> n;

    cout << "Harmonic series terms:"<<endl;

    for (int i = 1; i <= n; i++) {
        cout << "1/" << i;
        if (i != n)
            cout << " + ";
        sum += 1.0 / i;
    }
    cout << endl;
    cout << "Sum of the series: " << sum << endl;

    return 0;
}
```

Output:



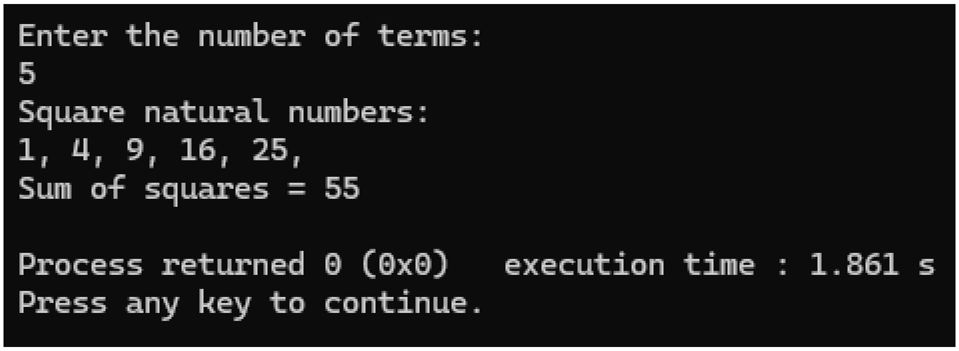
```
Enter the number of terms of harmonic series:
5
Harmonic series terms:
1/1 + 1/2 + 1/3 + 1/4 + 1/5
Sum of the series: 2.28333

Process returned 0 (0x0)   execution time : 10.981 s
Press any key to continue.
```

2.Write a C++ program that displays the n terms of the square natural numbers and their sum. 1 4 9 16 ... n Terms

Sol:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Enter the number of terms: "<<endl;
    cin >> n;

    int sum = 0;
    cout << "Square natural numbers: "<<endl;
    for (int i = 1; i <= n; ++i) {
        int square = i * i;
        cout << square << ", ";
        sum += square;
    }
    cout << "\nSum of squares = " << sum << endl;

    return 0;
}
```

Output:



```
Enter the number of terms:
5
Square natural numbers:
1, 4, 9, 16, 25,
Sum of squares = 55

Process returned 0 (0x0)   execution time : 1.861 s
Press any key to continue.
```

Conclusion:

During this experiment, I learnt how to implement and use different looping constructs in C. I understood how a for loop is useful when the number of iterations is known, how a while loop can repeatedly execute until a condition is met (like calculating factorial), and how a do-while loop ensures the loop runs at least once (useful for taking input and calculating sums). This helped me clearly see the differences between the three loops and when to apply each of them in problem-solving.

Signature of faculty in-charge with Date: