**Introduction to Computing-Lab**

**Topic: Loop Statement**

# Objectives

To enable students to understand the concept and implementation of the while loop in C++, empowering them to use this control structure effectively for repetitive tasks in their programs**.**

# Outcomes

By the end of this session, students will be able to:

1. Describe the purpose and structure of the while loop in C++.
2. Implement while loops in C++ to solve iterative problems.
3. Identify and resolve logical errors, such as infinite loops, in code using while loops.
4. Use while loops to handle practical programming scenarios, including user input validation, calculating sums, and managing conditional iterations.

# Content Overview

1. Introduction to Loops:
   * Definition and importance of loops in programming.
   * Overview of different types of loops in C++ (while, for, do-while).
2. The while Loop in C++:
   * Syntax and flowchart of the while loop.
   * Condition checking and execution flow.
3. Examples of while Loops:
   * Basic examples of iteration.
   * Handling user input (e.g., repeating until valid input is given).
   * Summation of numbers and factorial calculation.
4. Common Errors and Debugging:
   * Infinite loops and their causes.
   * Debugging tips for conditional errors in while loops.
5. Applications of while Loops:
   * Practical use cases (e.g., menu-driven programs, data validation).
   * Comparison with other loops in specific scenarios.
6. Hands-on Practice:
   * Write programs using while loops to solve practical problems.
   * Group activities for debugging and optimizing existing loop structures.

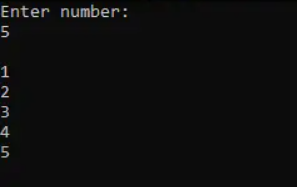
|  |  |
| --- | --- |
| **Repetition:**  In repetition, the program repeats particular statements a certain number of times based on some conditions. Normally three types repetition statement are used:  **Counter control:**  In this kind of repetition, a set of statement repeats specific number of times according to the requirement. You know exactly how many times certain statements need to be executed.  **Sentinel control:**  You do not always know how many pieces of data (or entries) need to be read, but you may know that the last entry is a special value, called a sentinel.  **Flag control:**  A flag variable is a bool variable that indicates whether a condition is true or false.  There are multiple repetition statements: | |
| **for:**  Syntax:    Flow of for loop: | **while:**  Syntax:    Flow of while loop: |
| The for loop executes as follows:   1. The initial statement executes. 2. The loop condition is evaluated. If the loop condition evaluates to true:    1. Execute the for loop statement.    2. Execute the update statement (the third expression in the parentheses). 3. Repeat Step 2 until the loop condition evaluates to false.   The initial statement usually initializes a variable (called the for loop control, or for indexed, variable). In C++, for is a reserved word. | The expression provides an entry condition to the loop. If it initially evaluates to true, the statement executes. The loop condition—the expression—is then reevaluated. If it again evaluates to true, the statement executes again. The statement (body of the loop) continues to execute until the expression is no longer true. A loop that continues to execute endlessly is called an infinite loop. To avoid an infinite loop, make sure that the loop’s body contains statement(s) that assure that the entry condition—the expression in the while statement—will eventually be false. In C++, while is a reserve word. |

## Lab Tasks:

## Task 1:

##### **Write a Program to Print numbers from 1 to n using while loop.**

**Sample Output**:



Hint: n is the ending number till you want to print.

## Task 2

Write a program in which the user enters a number **X** and your program prints all the numbers from **X** to zero

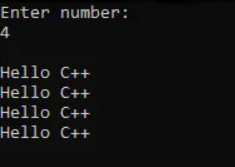
Expected Output

|  |
| --- |
| Enter a number: 5  5 4 3 2 1 0 |

## Task 3

##### **Program to print one statement for n time using while loop**

**Sample Output**:



## Task 4

Write a program in which the user enters a positive number and your program displays its digit in reverse order.

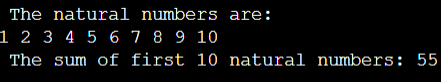
Expected Output:

Enter number: 1234

4321

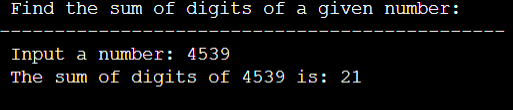
## Task 5

Write a program to find the sum of the first 10 natural numbers.



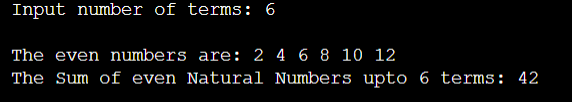
## Task 6

Write a program to find the sum of the digits of a given number.



## Task 7

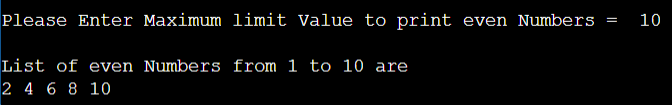
Write a program that displays the sum of the n terms of even natural numbers.



## Task 8

Write a program that prints even numbers from 2 to 10 using a while loop

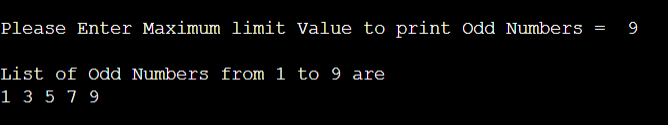
Expected Output:



## Task 9

Write a program that prints odd numbers from 1 to 9 using a while loop

Expected output:



## Task 10

Write a program that uses a while loop to print numbers from 5 to 1 in reverse order.

## Task 11

Write a program that prints the multiplication table of 3 using a while loop (up to 30).

## Task 12

Write a program that prints the powers of 3 up to 81 using a while loop.

## Task 13

Write a program that uses a while loop to print negative numbers from -3 to -1

## Task 14

You have to find sum of the square of every digits of the number

Example: number is 1234

**Output should be**

Number is 1234

Digit is 4 its square is 16

Digit is 3 its square is 9

Digit is 2 its square is 4

Digit is 1 its square is 1

Square Digit sum for the number is 30

## Task 15

Write a program to calculate and print the factorial of a given number (less than 5) using a while loop.

**Output should be:**

Factorial for number is [4 \* 3 \* 2 \* 1] is 24

**[Keep in mind that you cannot calculate factorial for very large numbers easily]**