**Lab 06 -Nested For Loop, While Loop**

**Lab Outcomes:**

* **Understand Nested Loop Concepts**: Gain a solid understanding of how nested loops operate, including how the outer and inner loops interact to produce patterns or iterate through multidimensional data.
* **Pattern-Based Problems**: Practice creating various patterns (e.g., triangles, squares, pyramids) using nested loops, developing logic and control over loop iterations.

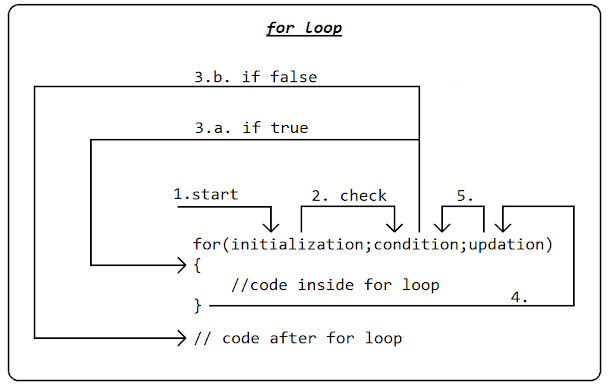
**Revise:**

**Definition of for Loop in C++?**

A for loop is a control structure that allows you to repeat a block of code a specific number of times. It is often used when you know in advance how many times you want to execute a particular task.

**Why Do We Use for Loop?**

We use for loop to automate repetitive tasks in a program. It helps to avoid writing the same code multiple times, making the code shorter and easier to read. By using a loop, we can execute a task as many times as we need without manually repeating the code.

****

**Problems Solved by for Loop:**

The for loop solves the problem of repetition in programming. For example, if you want to print numbers from 1 to 10, instead of writing a print statement ten times, you can use a for loop that repeats the printing process for each number.

**Example:**

Imagine you are a student helping to arrange chairs for a classroom. You have 10 chairs and you need to put them in a straight line. Instead of picking up one chair, placing it, then repeating the same for each chair, you decide to automate your actions:

1. **Initialization:** Start with chair 1.
2. **Condition:** As long as the number of chairs is less than or equal to 10.
3. **Increment:** Move to the next chair each time.

**Code:**

**Display a text 5 times**

**for (int i = 1; i <= 10; i++) {**

**cout << "Placing chair " << i << endl;**

**}**

In computer programming, loops are used to repeat a block of code.

For example, let's say we want to show a message 100 times. Then instead of writing the print statement 100 times, we can use a loop.

There are **3** types of loops in C++.

1. For Loop
2. While Loop
3. Do while

**Nested Loop:**

A **nested for loop** is a loop inside another loop. You use this structure when you need to perform repetitive tasks within another repetitive task.

**Problem Solved by Nested Loops:**

Nested loops are helpful when working with **multi-dimensional data**, like a table or a grid. They allow us to process or manipulate data that has rows and columns or any structure with multiple layers.

**For example:**

* **Tables**: Processing data in rows and columns, like a table of students’ scores.

**Syntax for Nested For loop:**

for ( initialization; condition; increment ) {

for ( initialization; condition; increment ) {

// statement of inside loop

}

// statement of outer loop

}

Example Code 1.1:

#include<iostream>

using namespace std;

// Outer loop

int main(){

for (int i = 1; i <= 2; i++) {

  cout << "Outer: " << i << "\n"; // Executes 2 times

  // Inner loop

  for (int j = 1; j <= 3; j++) {

    cout << " Inner: " << j << "\n"; // Executes 6 times (2 \* 3)

  }

}

return 0;

}

**Output: **

**Example Code 1.2:**

#include<iostream>

using namespace std;

// Outer loop

int main(){

for (int i = 1; i <= 5; i++) {

  // Inner loop

  for (int j = 1; j <= 5; j++) {

    cout << "#"; // Executes 25 times (5 \* 5)

  }

  cout<<endl;

}

return 0;

}

**Output:**

****

**Example Code 1.3:**

#include<iostream>

using namespace std;

int main(){

    int i=5;

    int j;

    for(i;i>=1;i--){

        for(j=1;j<=i;j++){

            cout<<"\*";

        }

        cout<<endl;

    }

    return 0;

}

**Output:**

****

# **While Loop:**

Loops can execute a block of code as long as a specified condition is reached.

Loops are handy because they save time, reduce errors, and they make code more readable.

**Another Definition**

A while loop is a way to repeat a set of instructions as long as a specific condition is true.

**Syntax:**

while (condition) {  
*// code block to be executed*  
}

**How it works:**

1. The loop starts by checking a condition.
2. If the condition is true, the code inside the loop runs.
3. After running, it goes back and checks the condition again.
4. This keeps going in a "loop" until the condition is no longer true, at which point it stops.

For example: If you're stacking books until you reach a height of 10 books, you'd keep adding a book, then check, “Is it 10 yet?” If not, you add another. Once you reach 10, you stop. This is how a while loop works!

**Example Code:**

int i = 1;  
while (i <= 5) {  
  cout << i <<endl;  
  i++;  
}

**Example Code for nested while loop:**

#include <iostream>

using namespace std;

int main() {

    int i = 1; // Initialize the outer loop variable

    while (i <= 2) { // Outer loop condition

        cout << "Outer: " << i << endl; // Executes 2 times

        int j = 1; // Initialize the inner loop variable

        while (j <= 3) { // Inner loop condition

            cout << " Inner: " << j << endl; // Executes 6 times (2 \* 3)

            j++; // Increment inner loop variable

        }

        i++; // Increment outer loop variable

    }

    return 0;

}

**Output:**

****

**Lab Task**

**Note: Implement Each Task using nested loop and nested while loop.**

**Problem Statement:**

**Task 1:** Write a C++ program that will print the following pattern.

\*

\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Task 2:** Write a program to print a pyramid pattern with numbers where each row has incrementally increasing numbers.

1

1 2

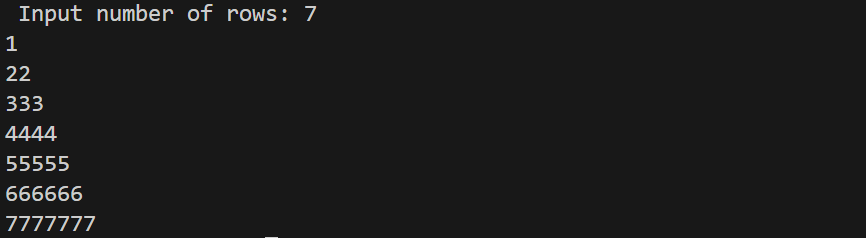
1 2 3

1 2 3 4

1 2 3 4 5

**Task 3:** Write a program in C++ to print a pattern based on a user-input number of rows. The pattern should display increasing rows where each row i contains the integer i repeated i times.

**Output:**

****

**Task 4:** Create a program that outputs the days for a specified number of weeks. This will help you practice using nested loops in C++ and improve your understanding of iteration and output formatting.

**Requirements:**

1. **Define the Number of Weeks:**
   * You should set the number of weeks to display to **3**.
2. **Display Days in Each Week:**
   * Each week should display **7 days**.
   * Each day should be numbered from **1** to **7**.
3. **Output Format:**
   * The output should clearly indicate the week number followed by each day in that week, with proper indentation for readability.

**Task 5:** Let’s consider a **seating arrangement in a theater**. Imagine a theater with multiple rows of seats, where each row has several seats numbered from left to right. If we want to display the seat numbers in each row.

There are 5 rows and each row has 10 seats. A nested loop can help us print each seat's number in each row.

**Rubrics Based LAB**

**Lab #06 Marks distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ER1** | **ER6** | **ER8** |
| **Task** | 3 points | 3 points | 4 points |

**Lab # 06 Rubric Evaluation Guideline:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Qualities & Criteria** | **0 < Poor <=1** | **1 < Satisfactory <= 2** | **2 < Excellent <=3** |
| **ER1** | **Task Completion** | Minimal or no program functionality was achieved. | Some tasks were completed, but the program has errors or incomplete functionalities. | All tasks were completed, and the program runs without errors. |
| **#** | **Qualities & Criteria** | **0 < Poor <=1** | **1 < Satisfactory <= 2** | **2< Excellent <=3** |
| **ER6** | Program Output | Output is inaccurate or poorly presented. | Output is mostly accurate but may lack labels, captions, or formatting. | Output is clear, accurate, and well presented with labels, captions, and proper formatting. |
| **#** | **Qualities & Criteria** | **0 < Poor <= 2** | **2< Satisfactory <= 3** | **3< Excellent <= 4** |
| **ER8** | Question & Answer | Answers some questions but not confidently or based on lab task knowledge. | Answers most questions confidently and based on lab task knowledge. | Answers all questions confidently and demonstrates a deep understanding of the given lab task. |