كليـــة العلـــــوم
قسم الأمن السيبراني

**Subject: Programming Fundamentals**

**First Stage**

**Lecturer: Dr. Abdulkadhem A. Abdulkadhem**

**Lecture (7)**

**Loop Statements in C++**

**Objective:**

By the end of this lecture, students will:

1. Understand the purpose of loop statements.
2. Learn the syntax and use cases for **while, do...while, and for** loops.
3. Differentiate between the three types of loops and apply them in practical examples.

**1. Introduction to Loops**

Loops allow us to execute a block of code repeatedly until a certain condition is met.

* They help reduce redundancy and make code more efficient and readable.
* Types of loops in C++:
	+ **while loop**
	+ **do...while loop**
	+ **for loop**

**2. The while Loop**

* **Definition**: Executes a block of code as long as the given condition is **true**.
* **Syntax**:



* **Key Point**: The condition is evaluated *before* the loop body runs.

**Example:**

|  |
| --- |
| #include <iostream>using namespace std;int main() { int i = 1; while (i <= 5) { cout << "Iteration " << i << endl; i++; // Increment } return 0;} |

**Use Case: Reading input until a valid value is provided.**

**3. The do...while Loop**

* **Definition**: Executes the block of code once, and then repeats as long as the condition is **true**.
* **Syntax**:



* **Key Point**: The condition is evaluated *after* the loop body runs, guaranteeing at least one execution.

**Example:**

|  |
| --- |
| #include <iostream>using namespace std;int main() { int number; do { cout << "Enter a number greater than 10: "; cin >> number; } while (number <= 10); cout << "You entered: " << number << endl; return 0;} |

**Use Case: Input validation where at least one attempt is needed.**

**4. The for Loop**

* **Definition**: Ideal for situations where the number of iterations is known beforehand.
* **Syntax**:



* **Key Point**: Combines initialization, condition-checking, and increment in one line.

**Example:**

|  |
| --- |
| #include <iostream>using namespace std;int main() { for (int i = 1; i <= 5; i++) { cout << "Iteration " << i << endl; } return 0;} |

**Use Case: Iterating through arrays or a fixed range of values.**

**5. Comparison of Loop Statements**

| **Feature** | **while** | **do...while** | **for** |
| --- | --- | --- | --- |
| **Condition Check** | Before the loop | After the loop | Before the loop |
| **Best Use Case** | Unknown iterations | At least one iteration needed | Known iterations |

**6. Practical Example**

Task: Print the sum of numbers from 1 to 10 using all three loops.

|  |
| --- |
| **Using while:** |
| int sum = 0, i = 1;**while (i <= 10) {** **sum += i;** **i++;****}**cout << "Sum: " << sum << endl; |
| **Using *do...while*:** |
| int sum = 0, i = 1;**do {** **sum += i;** **i++;****} while (i <= 10);**cout << "Sum: " << sum << endl; |
| **Using for:** |
| int sum = 0;**for (int i = 1; i <= 10; i++) {** **sum += i;****}**cout << "Sum: " << sum << endl; |

**7. Common Mistakes**

1. **Infinite Loops**: Forgetting to update the variable in the loop body.
Example:

|  |
| --- |
| int i = 1;while (i <= 5) { cout << i << endl; // No increment} |

1. **Incorrect Condition**: Using a wrong or overly restrictive condition.

|  |
| --- |
|  int number = 0; // Trying to enter a positive number **do {** **cout << "Enter a positive number: ";** **cin >> number;** **} while (number > 0);** **// Incorrect condition: This should be "number <= 0"** cout << "You entered: " << number << endl; |