

KARACHI EDU COACHING CENTER • COMPUTER SCIENCE CLASS X

Chapter 4: Control Structures in C++

Detailed Coaching Notes • Sequential, Selection (if, switch), and Repetition (loops)

What is a Control Structure?



Defining Flow Control

A control structure directs the order of execution of statements in a program based on certain conditions or loops.



Three Main Categories

- **Sequential:** Statement-by-statement execution.
- **Selection (Branching):** Making decisions.
- **Repetition (Looping):** Repeating code.

Sequential Flow of Control

Default Execution Flow

By default, C++ executes statements in the exact order they appear from top to bottom, one after another without skipping.

```
int a = 5;  
int b = 10;  
int sum = a + b;  
cout << sum;
```

Flow Diagram Concept

Step 1 ↓ Step 2 ↓ Step 3.

Every standard C++ program starts with sequential instructions inside `main()` before control structures are applied.

Decision Making: `if` & `if-else`

The `if-else` Statement

Evaluates a boolean condition. If true, block A runs; if false, block B runs.

```
if (marks ≥ 50) {  
    cout << "Passed";  
} else {  
    cout << "Failed";  
}
```

Nested `if` Statements

An `if` statement inside another `if` block used for multi-layered conditional filtering (e.g., grading systems).

Essential for evaluating Karachi Edu grading criteria (A-1, A, B, C grades).

Multi-Way Selection with `switch`

Switch-Case Syntax

An alternative to long `if-else-if` ladders when checking a single integer or character expression.

```
switch(choice) {  
    case 1: cout << "Math"; break;  
    case 2: cout << "Computer"; break;  
    default: cout << "Invalid";  
}
```

Key Rules of `switch`

- The `break` statement is crucial to prevent fall-through execution.
- The `default` case handles any unlisted inputs.
- Expressions must evaluate to integral types or characters.

Introduction to Loops in C++



Why Use Loops?

Loops allow running a block of code multiple times continuously, eliminating code redundancy and automating repetitive calculations.



Types of Loops in C++

- **for loop:** Counter-controlled iteration.
- **while loop:** Entry-controlled condition loop.
- **do-while loop:** Exit-controlled loop.

Counter-Controlled Iteration: for

Syntax Structure

Combines initialization, condition check, and increment/decrement into a single line.

```
for(int i = 1; i ≤ 10; i++) {  
    cout << i << endl;  
}
```

Execution Breakdown

1. **Initialization:** Sets loop counter (i=1 once).
2. **Condition:** Evaluates before each iteration.
3. **Increment:** Updates counter after execution block.

Condition-Controlled & Exit-Controlled Loops

while Loop

Entry-controlled loop. Checks condition first; if false initially, loop body never executes.

```
while(x > 0) { x--; }
```

do-while Loop

Exit-controlled loop. Guarantees the code block runs **at least once** before checking condition.

```
do { cin >> n; } while(n ≠ 0);
```

Interrupting Control: break & continue

break Statement

Immediately terminates the execution of the nearest enclosing loop or switch statement, transferring control outside.

continue Statement

Skips the remaining statements in the current iteration of a loop and immediately proceeds to the next iteration check.

Chapter 4 Review Complete

KARACHI EDU • Coaching Center Notes
