كليـــة العلـــــوم
قسم الأمن السيبراني

**Subject: Programming Fundamentals**

**First Stage**

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

**Lecture (6)**

**Switch Case Selection**

**1. Introduction to the switch Statement**

**Definition:** The switch statement in C++ is a control structure used to execute one block of code out of many options, based on the value of an expression.

**Key Characteristics:**

* Suitable for situations with multiple discrete choices.
* Provides a cleaner alternative to multiple if-else statements.

**2. Structure of the switch Statement**

****

**Syntax:**

|  |
| --- |
| switch (expression) { case value1: // Code to execute if expression == value1 break; case value2: // Code to execute if expression == value2 break; ... default: // Code to execute if no case matches break;} |

**Components:**

1. **Expression:** A value or variable evaluated to determine which case to execute.
2. **Case Labels:** Define values to compare against the expression.
3. **Break Statement:** Exits the switch block to prevent fall-through.
4. **Default Case:** An optional catch-all block executed if no case matches.

**3. Rules and Guidelines for Using switch**

* The expression must evaluate to an integer, character, or enumerated type.
* Case labels must be constants or literals (e.g., case 1: or case 'A':).
* The break statement is essential to prevent fall-through.
* Avoid using floating-point or string types as expressions in switch.

**4. Advantages and Limitations**

**Advantages:**

* Improves readability compared to nested if-else.
* Executes faster for large, discrete sets of choices.

**Limitations:**

* Limited to discrete values (no ranges or complex conditions).
* Does not support expressions with logical or relational operators.

**5. Examples with Code and Explanation of switch**

**Example 1:** Write a program to print the day of the week based on a number input.

|  |
| --- |
| #include <iostream>using namespace std;int main() { int day; cout << "Enter a number (1-7): "; cin >> day; switch (day) { case 1: cout << "Monday"; break; case 2: cout << "Tuesday"; break; case 3: cout << "Wednesday"; break; case 4: cout << "Thursday"; break; case 5: cout << "Friday"; break; case 6: cout << "Saturday"; break; case 7: cout << "Sunday"; break; default: cout << "Invalid input! Please enter a number between 1 and 7."; break; } return 0;} |

**Explanation:**

* The switch block evaluates the variable day.
* Each case corresponds to a day of the week.
* The default block handles invalid inputs.

**Example 2: Write a program to classify grades into categories based on score ranges.**

|  |
| --- |
| #include <iostream>using namespace std;int main() { int grade; cout << "Enter your grade (0-100): "; cin >> grade; switch (grade / 10) { // Divide by 10 to categorize into ranges case 10:  cout << "Excellent +"; break;  case 9: cout << "Excellent"; break; case 8: cout << "Very Good"; break; case 7: cout << "Good"; break; case 6: cout << "Satisfactory"; break; default: if (grade >= 0 && grade < 60) cout << "Fail"; else **cout << "Invalid grade! Please enter a value between 0 and 100.";** break; } return 0;} |

**Explanation**

1. **Grade Categorization:** The input grade is divided by 10, grouping it into ranges like 90-100, 80-89, etc.
2. **Case Labels:**
	* 10 and 9: "Excellent" for scores between 90 and 100.
	* 8: "Very Good" for scores between 80 and 89.
	* 7: "Good" for scores between 70 and 79.
	* 6: "Satisfactory" for scores between 60 and 69.
	* default: Handles scores below 60 as "Fail" or invalid values outside the 0-100 range.

**Example 3:** Write a C++ program that reads two integer numbers, an operation, and performs the selected operation using a switch statement

|  |
| --- |
| #include <iostream>using namespace std;int main() { int a, b; char x; // Prompt user for input cout << "Enter two numbers:\n"; cin >> a >> b; // Display menu options cout << "+ for addition\n"; cout << "- for subtraction\n"; cout << "\* for multiplication\n"; cout << "/ for division\n"; cout << "Enter your choice:\n"; cin >> x; // Perform operation based on user's choice **switch (x)** { **case '+':** cout << "Result: " << a + b << endl; break; **case '-':** cout << "Result: " << a - b << endl; break; **case '\*':** cout << "Result: " << a \* b << endl; break; **case '/':** if (b != 0) cout << "Result: " << a / b << endl; else cout << "Error: Division by zero is not allowed!" << endl; break; **default:** cout << "Invalid operation! Please select +, -, \*, or /." << endl; break; } return 0;} |

**Explanation of the Code**

1. **Input:**
	* The program prompts the user to enter two integers (a and b).
	* It also prompts the user to choose an operation (+, -, \*, /).
2. **Menu Options:**
	* Displays all valid operations for clarity.
3. **switch Statement:**
	* Checks the operation entered by the user (x) and performs the corresponding arithmetic.
	* The case labels handle specific operations:
		+ + for addition.
		+ - for subtraction.
		+ \* for multiplication.
		+ / for division.
	* The default block handles invalid input for the operation.
4. **Division by Zero Check:**
	* Ensures safe execution of division to avoid errors when dividing by zero.
5. **Output:**
	* Displays the result of the operation or an error message if the input is invalid.

### 6. Conditional (Ternary) Operator(?)

 The conditional (ternary) operator is a compact alternative to simple if-else statements. It evaluates a condition and returns one of two values based on the result.

**Syntax:**

|  |
| --- |
| condition ? expression1 : expression2; |

**Example 4: Write a C++ program to find the larger of two integers using Ternary Operator.**

|  |
| --- |
| #include <iostream>using namespace std;int main() { int a, b; cout << "Enter two numbers: "; cin >> a >> b; **int max = (a > b) ? a : b;** cout << "The larger number is: " << **max**; return 0;} |

**Explanation:**

* The expression (a > b) ? a : b evaluates whether a is greater than b.
* If true, it assigns a to max; otherwise, it assigns b.

**Use Cases:**

* Simple condition-based assignments.
* Avoid overusing it for complex logic to maintain code readability.

 **Advantages and Limitations**

**Advantages:**

* Improves readability compared to nested if-else.
* Executes faster for large, discrete sets of choices.

**Limitations:**

* Limited to discrete values (no ranges or complex conditions).
* Does not support expressions with logical or relational operators.

**Practice Questions**

1. Write a switch statement to output the corresponding month name for a given number (1-12).
2. Write a c++ program for using a ternary expression to check if a given integer n is even or odd. The expression should return "Even" if n is even and "Odd" if n is odd.