

1. What is a **switch** Statement?

The **switch** statement is a decision-making statement in C++ that allows a program to select one option from multiple choices.

Real-life example: A university student selects an option from a menu:

1. View Student Profile
2. View Semester Result
3. Check Fee Status
4. Exit

Instead of writing many **if-else-if** conditions, we can use a **switch** statement.

2. Basic Syntax

```
switch (expression)
{
    case value1:
        // Statements
        break;

    case value2:
        // Statements
        break;

    default:
        // Statements if no case matches
}
```

Important keywords

- **switch**: Checks the value of an expression.
- **case**: Defines a possible value.
- **break**: Ends the switch block and prevents execution from continuing into the next case.
- **default**: Executes when no case matches. It is optional.

Pattern 1: Display a Message Based on a Number

Example 1: Day Selection

```
#include <iostream>
using namespace std;

int main()
{
    int day;

    cout << "Enter a day number (1-3): ";
```

```
cin >> day;

switch (day)
{
    case 1:
        cout << "Monday";
        break;

    case 2:
        cout << "Tuesday";
        break;

    case 3:
        cout << "Wednesday";
        break;

    default:
        cout << "Invalid day number";
}

return 0;
}
```

Sample output

```
Enter a day number (1-3): 2
Tuesday
```

Understand the code

1. The user enters a number.
2. `switch (day)` checks its value.
3. If the value is `2`, the program displays `Tuesday`.
4. `break` exits the switch.
5. If the user enters any other number, the `default` block executes.

Modify the code

1. Add Thursday as case `4`.
2. Add Friday as case `5`.
3. Change the prompt to accept numbers from 1 to 5.
4. Test the program with `1`, `4`, and `8`.

Pattern 2: Simple Calculator

A `switch` statement can select an arithmetic operation.

Example 2: Addition, Subtraction, Multiplication and Division

```
#include <iostream>
using namespace std;

int main()
{
    double num1, num2;
    char op;

    cout << "Enter first number: ";
    cin >> num1;

    cout << "Enter operator (+, -, *, /): ";
    cin >> op;

    cout << "Enter second number: ";
    cin >> num2;

    switch (op)
    {
        case '+':
            cout << "Result = " << num1 + num2;
            break;

        case '-':
            cout << "Result = " << num1 - num2;
            break;

        case '*':
            cout << "Result = " << num1 * num2;
            break;

        case '/':
            if (num2 != 0)
            {
                cout << "Result = " << num1 / num2;
            }
            else
            {
                cout << "Cannot divide by zero";
            }
            break;

        default:
            cout << "Invalid operator";
    }

    return 0;
}
```

Sample output

```
Enter first number: 10
Enter operator (+, -, *, /): *
Enter second number: 5
Result = 50
```

What to understand

- `char op` stores a single operator.
- Character cases use single quotes, such as `case '+'`.
- The division case uses an `if` statement inside a `switch` to prevent division by zero.

Modify the code

1. Add the modulus operator `%` using integer variables.
2. Add a message for an invalid operator.
3. Change the program to perform only addition and subtraction.
4. Test division with `num2 = 0`.

Pattern 3: Multiple Cases with the Same Result

When multiple choices should produce the same output, we can group cases together.

Example 3: Weekend or Weekday

```
#include <iostream>
using namespace std;

int main()
{
    int day;

    cout << "Enter day number (1-7): ";
    cin >> day;

    switch (day)
    {
        case 1:
        case 2:
        case 3:
        case 4:
        case 5:
            cout << "Weekday";
            break;

        case 6:
        case 7:
            cout << "Weekend";
            break;

        default:
```

```
        cout << "Invalid day number";  
    }  
  
    return 0;  
}
```

Sample output

```
Enter day number (1-7): 6  
Weekend
```

Here, cases 1 through 5 share the same statements. The cases do not contain `break` until the shared result is reached, so execution continues through the grouped labels.

Modify the code

1. Display "Working Day" for cases 1–5.
2. Display "Holiday" for cases 6–7.
3. Add a message for an invalid day number.

Pattern 4: Menu-Driven Program

Menu-driven programs are common in university systems, banking applications and management software.

Example 4: Student Management Menu

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    int choice;  
  
    cout << "\n--- Student Management System ---\n";  
    cout << "1. Student Profile\n";  
    cout << "2. Semester Result\n";  
    cout << "3. Fee Status\n";  
    cout << "4. Exit\n";  
  
    cout << "Enter your choice: ";  
    cin >> choice;  
  
    switch (choice)  
    {  
        case 1:  
            cout << "Displaying student profile.";  
            break;  
  
        case 2:
```

```
        cout << "Displaying semester result.";
        break;

    case 3:
        cout << "Displaying fee status.";
        break;

    case 4:
        cout << "Exiting the program.";
        break;

    default:
        cout << "Invalid choice.";
    }

    return 0;
}
```

Sample output

```
--- Student Management System ---
1. Student Profile
2. Semester Result
3. Fee Status
4. Exit
Enter your choice: 3
Displaying fee status.
```

This program demonstrates how a university application could direct users to different functions based on their selection. The current example displays messages rather than retrieving actual student records.

Modify the code

1. Add option 5 for Attendance Status.
2. Add option 6 for Course Registration.
3. Change the messages to suit your university.
4. Test the program with choices 2, 5, and 9.

Pattern 5: Grade or Result Category

A `switch` statement can also select a message based on a character grade.

Example 5: Grade Notification

```
#include <iostream>
using namespace std;

int main()
{
```

```
char grade;

cout << "Enter your grade (A, B, C, D, F): ";
cin >> grade;

switch (grade)
{
    case 'A':
        cout << "Excellent Performance";
        break;

    case 'B':
        cout << "Very Good Performance";
        break;

    case 'C':
        cout << "Good Performance";
        break;

    case 'D':
        cout << "You Passed";
        break;

    case 'F':
        cout << "You Failed";
        break;

    default:
        cout << "Invalid grade";
}

return 0;
}
```

Sample output

```
Enter your grade (A, B, C, D, F): A
Excellent Performance
```

Important: Character comparisons are case-sensitive. Entering lowercase **a** will not match uppercase **'A'** in this example.

Modify the code

1. Add a message for grade **B**.
2. Add a case for grade **W** to display **"Result Withheld"**.
3. Test the program with uppercase **A**, lowercase **a**, and **F**.

Pattern 6: Nested **if** Inside a **switch**

You can combine **switch** with **if-else** when a menu option requires additional checking.

Example 6: Fee Payment Status

```
#include <iostream>
using namespace std;

int main()
{
    int choice;
    double fee;

    cout << "1. Check Fee Status\n";
    cout << "2. Exit\n";
    cout << "Enter choice: ";
    cin >> choice;

    switch (choice)
    {
        case 1:
            cout << "Enter fee amount paid: ";
            cin >> fee;

            if (fee >= 50000)
            {
                cout << "Fee requirement completed.";
            }
            else
            {
                cout << "Remaining fee must be paid.";
            }
            break;

        case 2:
            cout << "Goodbye!";
            break;

        default:
            cout << "Invalid choice.";
    }

    return 0;
}
```

Sample output

```
1. Check Fee Status
2. Exit
Enter choice: 1
Enter fee amount paid: 30000
Remaining fee must be paid.
```

This is a simplified teaching example. In a real university system, the required fee would depend on the student's actual fee voucher and account balance.

Modify the code

1. Change the required fee from 50000 to 60000.
2. Add a message when the paid amount is exactly 50000.
3. Add a new menu option for viewing the semester result.

3. Difference Between `switch` and `if-else-if`

Feature	<code>switch</code>	<code>if-else-if</code>
Best use	Selecting among discrete values	Checking ranges and complex conditions
Example	Menu choice 1, 2 or 3	Marks greater than or equal to 80
Conditions	Matches case values	Supports relational and logical conditions
Range checking	Not directly supported	Supported
Data types	Integral types, <code>char</code> , and <code>enum</code> (also certain compatible integral types)	Conditions that evaluate to true or false
<code>break</code>	Commonly used to exit each case	Not required
<code>default</code> / <code>else</code>	Optional fallback	Optional fallback

Remember: `switch` is suitable for selecting a menu option or matching a grade. Use `if-else-if` for grade ranges such as 80–89 or for conditions involving multiple comparisons.

4. Practice Coding Exercises

Try solving these exercises without copying the examples.

1. Day of the Week

Easy

Input a number from 1 to 7 and display the corresponding day.

2. Simple Calculator

Easy

Input two numbers and an operator. Perform the selected operation.

3. Vowel or Consonant

Medium

Input a character and use switch cases to identify a, e, i, o, or u as vowels. Handle uppercase letters too.

4. University Menu

Medium

Create a menu with options for Profile, Result, Fee Status, Attendance and Exit.

5. Grade Notification

Medium

Input a character grade and display a suitable student notification.

6. Month Days

Advanced

Input a month number and display the number of days. For February, display 28 days and mention that leap years are not considered.

7. ATM Menu

Advanced

Create a menu for Balance Inquiry, Deposit, Withdrawal and Exit. Use switch to process the user's selection.

5. Common Mistakes to Avoid

1. Forgetting **break**: The program may continue into the following cases.
2. Using incorrect quotes: Use **case 'A':** for a character and **case 1:** for an integer.
3. Using duplicate case values: Each case value must be unique within the same switch.
4. Expecting range checks: **case 50-59:** is not valid C++ syntax.
5. Forgetting **default**: Without it, an unmatched value may produce no message.
6. Using floating-point values as case labels: **float** and **double** are not valid switch controlling types.

6. Quick Revision

Ask students to answer these questions:

1. What is the purpose of a **switch** statement?
2. What is the difference between **case** and **default**?
3. Why do we use **break**?
4. Can multiple cases display the same message?
5. Can we place an **if-else** statement inside a **switch**?
6. When should we use **if-else-if** instead of **switch**?