

Assignment-2

CDAC Mumbai Lab Assignment

Section 1: Error-Driven Learning in Java

Objective: This assignment focuses on understanding and fixing common errors encountered in Java programming. By analyzing and correcting the provided code snippets, you will develop a deeper understanding of Java's syntax, data types, and control structures.

Instructions:

1. **Identify the Errors:** Review each code snippet to identify the errors or issues present.
 2. **Explain the Error:** Write a brief explanation of the error and its cause.
 3. **Fix the Error:** Modify the code to correct the errors. Ensure that the code compiles and runs as expected.
 4. **Submit Your Work:** Provide the corrected code along with explanations for each snippet.
-

Snippet 1:

```
public class Main {  
    public void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

- What error do you get when running this code?

The program will not execute because the `main` method is incorrectly declared. The method signature for the `main` method must be: `public static void main(String[] args)`

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main.java
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main.java  
error: 'main' method is not declared 'public static'
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

//Corrected code-

```
|  
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

```
12 D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main.java
```

```
13 D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main.java  
14 Hello, World!
```

```
15 D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 2:

```
public class Main {  
    static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

- **What happens when you compile and run this code?**

Ans-

Error-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main2.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main2.java  
error: 'main' method is not declared 'public static'  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

//Corrected Code-

```
public class Main2 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main2.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main2.java  
Hello, World!  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 3:

```
public class Main {  
    public static int main(String[] args) {  
        System.out.println("Hello, World!");  
        return 0;  
    }  
}
```

- **What error do you encounter? Why is void used in the main method?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main3.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main3.java  
error: 'main' method is not declared with a return type of 'void'  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

//Corrected code-

```
public class Main3 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main3.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main3.java  
Hello, World!  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 4:

```
public class Main {  
    public static void main() {  
        System.out.println("Hello, World!");  
    }  
}
```

- What happens when you compile and run this code? Why is String[] args needed?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main4.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main4.java  
error: can't find main(String[]) method in class: Main4  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

//Corrected Code-

```
public class Main4 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main4.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main4.java  
Hello, World!  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 5:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Main method with String[] args");  
    }  
    public static void main(int[] args) {  
        System.out.println("Overloaded main method with int[] args");  
    }  
}
```

- **Can you have multiple main methods? What do you observe?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main5.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main5.java  
Main method with String[] args  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 6:

```
public class Main {  
    public static void main(String[] args) {  
        int x = y + 10;  
        System.out.println(x);  
    }  
}
```

- What error occurs? Why must variables be declared?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main6.java  
Main6.java:4: error: cannot find symbol  
    int x = y + 10;  
            ^  
    symbol:   variable y  
    location: class Main6  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code-  
public class Main6 {  
    public static void main(String[] args) {  
        int y= 5;  
        int x = y + 10;  
        System.out.println(x);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main6.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main6.java  
15  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 7:

```
public class Main {  
    public static void main(String[] args) {  
        int x = "Hello";  
        System.out.println(x);  
    }  
}
```

- **What compilation error do you see? Why does Java enforce type safety?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main7.java  
Main7.java:4: error: incompatible types: String cannot be converted to int  
    int x = "Hello";  
           ^  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code-  
public class Main7 {  
    public static void main(String[] args) {  
        String x = "Hello";  
        System.out.println(x);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main7.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main7.java  
Hello  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```


Snippet 8:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!")  
    }  
}
```

- **What syntax errors are present? How do they affect compilation?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main8.java  
Main8.java:4: error: ')' expected  
        System.out.println("Hello, World!"  
                               ^  
Main8.java:7: error: class, interface, enum, or record expected  
What syntax errors are present? How do they affect compilation?  
^  
2 errors  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

//Corrected code-

```
public class Main8 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main8.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main8.java  
Hello, World!  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 9:

```
public class Main {  
    public static void main(String[] args) {  
        int class = 10;  
        System.out.println(class);  
    }  
}
```

- **What error occurs? Why can't reserved keywords be used as identifiers?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main9.java  
Main9.java:4: error: not a statement  
    int class = 10;  
    ^  
Main9.java:4: error: ';' expected  
    int class = 10;  
    ^  
Main9.java:4: error: <identifier> expected  
    int class = 10;  
    ^  
Main9.java:5: error: illegal start of expression  
    System.out.println(class);  
    ^  
Main9.java:5: error: <identifier> expected  
    System.out.println(class);  
    ^  
5 errors  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

//Corrected code-

```
public class Main9 {  
    public static void main(String[] args) {  
        int a = 10;  
        System.out.println(a);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main9.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main9.java  
10  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 10:

```
public class Main {
    public void display() {
        System.out.println("No parameters");
    }
    public void display(int num) {
        System.out.println("With parameter: " + num);
    }
    public static void main(String[] args) {
        display();
        display(5);
    }
}
```

- What happens when you compile and run this code? Is method overloading allowed?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main10.java
Main10.java:10: error: non-static method display() cannot be referenced from a static context
    display();
    ^
Main10.java:11: error: non-static method display(int) cannot be referenced from a static context
    display(5);
    ^
2 errors
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Java program has a **compilation error** because we trying to call the display() method inside main() without an object. Since display() is a non-static method, it must be called using an instance of the class.

```
public class Main10 {
    public void display() {
        System.out.println("No parameters");
    }
    public void display(int num) {
        System.out.println("With parameter: " + num);
    }
    public static void main(String[] args) {
        Main10 obj = new Main10();
        obj.display();
        obj.display(5);
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main10.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main10.java
No parameters
With parameter: 5
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 11:

```
public class Main {  
    public static void main(String[] args) {  
        int[] arr = {1, 2, 3};  
        System.out.println(arr[5]);  
    }  
}
```

- What runtime exception do you encounter? Why does it occur?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main11.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main11.java  
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 3  
    at Main11.main(Main11.java:5)  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

The runtime exception you encounter is an **ArrayIndexOutOfBoundsException**. This occurs because you're trying to access an index (`arr[5]`) that does not exist in the array. The array `arr` is initialized with three elements, which are stored at indices 0, 1, and 2. Attempting to access index 5 goes beyond the bounds of the array, causing this exception.

```
//Corrected code-  
public class Main11 {  
    public static void main(String[] args) {  
        int[] arr = {1, 2, 3};  
        System.out.println(arr[2]);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main11.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main11.java  
3  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 12:

```
public class Main {  
    public static void main(String[] args) {  
        while (true) {  
            System.out.println("Infinite Loop");  
        }  
    }  
}
```

- **What happens when you run this code? How can you avoid infinite loops?**

Ans-

Error- Infinite loop

```
//Corrected code-  
  
public class Main12 {  
    public static void main(String[] args) {  
        int count = 0;  
        while (count < 10) {  
            System.out.println("Loop iteration " + count);  
            count++;  
        }  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main12.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main12.java  
Loop iteration 0  
Loop iteration 1  
Loop iteration 2  
Loop iteration 3  
Loop iteration 4  
Loop iteration 5  
Loop iteration 6  
Loop iteration 7  
Loop iteration 8  
Loop iteration 9  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 13:

```
public class Main {  
    public static void main(String[] args) {  
        String str = null;  
        System.out.println(str.length());  
    }  
}
```

- What exception is thrown? Why does it occur?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main13.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main13.java  
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.length()" because "<local1>" is null  
    at Main13.main(Main13.java:5)  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

To correct the code and avoid the **NullPointerException**, you can either initialize `str` to a non-null value or add a check before accessing its length. Here are two approaches:

//Corrected code-

```
public class Main13 {  
    public static void main(String[] args) {  
        String str = "Hello"; // now str is not null  
        System.out.println(str.length()); // prints the length of the string  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main13.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main13.java  
5  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>^S_
```

Snippet 14:

```
public class Main {  
    public static void main(String[] args) {  
        double num = "Hello";  
        System.out.println(num);  
    }  
}
```

- **What compilation error occurs? Why does Java enforce data type constraints?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main14.java  
Main14.java:4: error: incompatible types: String cannot be converted to double  
    double num = "Hello";  
                ^  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code  
public class Main14 {  
    public static void main(String[] args) {  
        String num = "Hello";  
        System.out.println(num);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main14.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main14.java  
Hello  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 15:

```
public class Main {  
    public static void main(String[] args) {  
        int num1 = 10;  
        double num2 = 5.5;  
        int result = num1 + num2;  
        System.out.println(result);  
    }  
}
```

- **What error occurs when compiling this code? How should you handle different data types in operations?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main15.java  
Main15.java:6: error: incompatible types: possible lossy conversion from double to int  
        int result = num1 + num2;  
                        ^  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

When you perform operations on different data types, Java uses type promotion rules. In this case, adding an `int` and a `double` results in a `double`. If you try to assign that result to an `int` variable without explicit conversion, you'll get a compilation error.

//Corrected code-

```
public class Main15 {  
    public static void main(String[] args) {  
        int num1 = 10;  
        double num2 = 5.5;  
        double result = num1 + num2; // Now result is of type double  
        System.out.println(result);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main15.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main15.java  
15.5  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```


Snippet 16:

```
public class Main {  
    public static void main(String[] args) {  
        int num = 10;  
        double result = num / 4;  
        System.out.println(result);  
    }  
}
```

- What is the result of this operation? Is the output what you expected?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main16.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main16.java  
2.0  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

//Corrected code-

```
public class Main16 {  
    public static void main(String[] args) {  
        int num = 10;  
        double result = num / 4.0; // Using 4.0 instead of 4  
        System.out.println(result); // This will print 2.5  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main16.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main16.java  
2.5  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 17:

```
public class Main {  
    public static void main(String[] args) {  
        int a = 10;  
        int b = 5;  
        int result = a ** b;  
        System.out.println(result);  
    }  
}
```

- **What compilation error occurs? Why is the ** operator not valid in Java?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main17.java  
Main17.java:6: error: illegal start of expression  
    int result = a ** b;  
                   ^  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

This error occurs because Java does not support the ** operator for exponentiation. In Java, the ** operator is not defined for any data type

```
//Corrected code-  
public class Main17 {  
    public static void main(String[] args) {  
        int a = 10;  
        int b = 5;  
        int result = a * b;  
        System.out.println(result);  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main17.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main17.java  
50  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 18:

```
public class Main {  
    public static void main(String[] args) {  
        int a = 10;  
        int b = 5;  
        int result = a + b * 2;  
        System.out.println(result);  
    }  
}
```

- **What is the output of this code? How does operator precedence affect the result?**

Ans-

Operator Precedence: In Java, multiplication (*) has a higher precedence than addition (+). This means the multiplication $b * 2$ is evaluated before the addition.

Calculation: With $a = 10$ and $b = 5$, the expression $b * 2$ evaluates to $5 * 2 = 10$. Then, $a + 10$ equals

$10 + 10 = 20$.

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main18.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main18.java  
20  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 19:

```
public class Main {  
    public static void main(String[] args) {  
        int a = 10;  
        int b = 0;  
        int result = a / b;  
        System.out.println(result);  
    }  
}
```

- What runtime exception is thrown? Why does division by zero cause an issue in Java?

Ans-

In Java, **dividing an integer by zero is not allowed** because it is mathematically undefined.

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main19.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main19.java  
Exception in thread "main" java.lang.ArithmeticException: / by zero  
    at Main19.main(Main19.java:6)  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 20:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello, World")  
    }  
}
```

- **What syntax error occurs? How does the missing semicolon affect compilation?**

ANs-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main20.java  
Main20.java:4: error: ';' expected  
    System.out.println("Hello, World")  
                                ^  
1 error  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code-  
public class Main20 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World");  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main20.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main20.java  
Hello, World  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 21:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
        // Missing closing brace here  
    }  
}
```

- **What does the compiler say about mismatched braces?**

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main21.java  
Main21.java:7: error: ';' expected  
What does the compiler say about mismatched braces?  
    ^  
Main21.java:7: error: ';' expected  
What does the compiler say about mismatched braces?  
    ^  
Main21.java:7: error: ';' expected  
What does the compiler say about mismatched braces?  
    ^  
Main21.java:7: error: ';' expected  
What does the compiler say about mismatched braces?  
    ^  
Main21.java:7: error: reached end of file while parsing  
What does the compiler say about mismatched braces?  
    ^  
  
5 errors  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

//Corrected code-

```
public class Main21 {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
        // Missing closing brace here  
    }  
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main21.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main21.java  
Hello, World!  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 22:

```
public class Main {  
    public static void main(String[] args) {  
        static void displayMessage() {  
            System.out.println("Message");  
        }  
    }  
}
```

- What syntax error occurs? Can a method be declared inside another method?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main22.java  
Main22.java:4: error: illegal start of expression  
    static void displayMessage() {  
    ^  
Main22.java:8: error: class, interface, enum, or record expected  
    }  
    ^  
2 errors  
error: compilation failed  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code-  
public class Main22 {  
    public static void main(String[] args) {  
        displayMessage(); // Call the method  
    }  
  
    static void displayMessage() {  
        System.out.println("Message");  
    }  
}
```

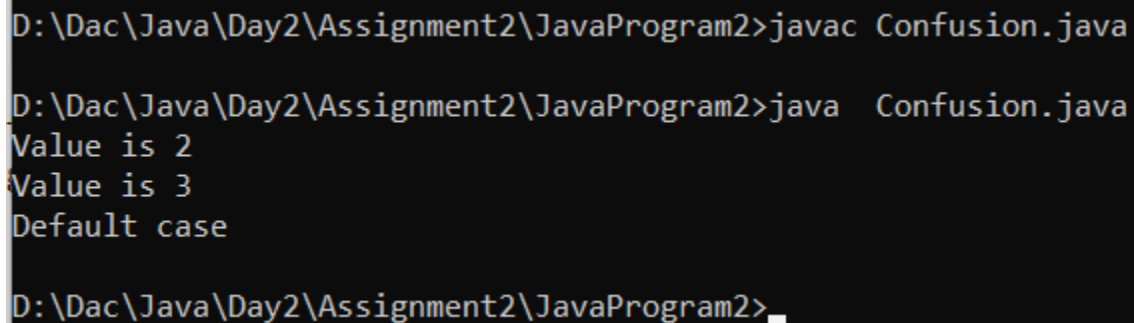
```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Main22.java  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Main22.java  
Message  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 23:

```
public class Confusion {  
    public static void main(String[] args) {  
        int value = 2;  
        switch(value) {  
            case 1:  
                System.out.println("Value is 1");  
            case 2:  
                System.out.println("Value is 2");  
            case 3:  
                System.out.println("Value is 3");  
            default:  
                System.out.println("Default case");  
        }  
    }  
}
```

- **Error to Investigate:** Why does the default case print after "Value is 2"? How can you prevent the program from executing the default case?

Ans-



```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Confusion.java  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Confusion.java  
Value is 2  
Value is 3  
Default case  
  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

The issue is caused by "fall-through" behavior in switch statements. When you don't include a `break` statement at the end of a case, Java continues to execute the subsequent cases until it hits a `break` or the switch ends. In your code, when `value` is 2, it executes case 2, then falls through to case 3 and the default case.


```
//Corrected code-
public class Confusion {
    public static void main(String[] args) {
        int value = 2;
        switch(value) {
            case 1:
                System.out.println("Value is 1");
                break;
            case 2:
                System.out.println("Value is 2");
                break;
            case 3:
                System.out.println("Value is 3");
                break;
            default:
                System.out.println("Default case");
                break;
        }
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Confusion.java

D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Confusion.java
Value is 2

D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 24:

```
public class MissingBreakCase {  
    public static void main(String[] args) {  
        int level = 1;  
        switch(level) {  
            case 1:  
                System.out.println("Level 1");  
            case 2:  
                System.out.println("Level 2");  
            case 3:  
                System.out.println("Level 3");  
            default:  
                System.out.println("Unknown level");  
        }  
    }  
}
```

- **Error to Investigate:** When level is 1, why does it print "Level 1", "Level 2", "Level 3", and "Unknown level"? What is the role of the break statement in this situation?

Ans-

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java MissingBreakCase.java  
Level 1  
Level 2  
Level 3  
Unknown level  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

When `level` is 1, the switch statement starts at case 1 and prints "Level 1". Since there's no break statement after case 1, the execution continues ("falls through") into case 2, then case 3, and finally the default case, printing all messages sequentially.

The break statement is used to exit the switch block once a matching case is executed. Without it, execution continues into the subsequent cases until a break is encountered or the switch block ends.

```
//Corrected code-
public class MissingBreakCase {
    public static void main(String[] args) {
        int level = 1;
        switch(level) {
            case 1:
                System.out.println("Level 1");
                break;
            case 2:
                System.out.println("Level 2");
                break;
            case 3:
                System.out.println("Level 3");
                break;
            default:
                System.out.println("Unknown level");
                break;
        }
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac MissingBreakCase.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java MissingBreakCase.java
Level 1
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Snippet 25:

```
public class Switch {
    public static void main(String[] args) {
        double score = 85.0;
        switch(score) {
            case 100:
                System.out.println("Perfect score!");
                break;
            case 85:
                System.out.println("Great job!");
                break;
            default:
                System.out.println("Keep trying!");
        }
    }
}
```

- **Error to Investigate:** Why does this code not compile? What does the error tell you about the types allowed in switch expressions? How can you modify the code to make it work?

Ans-

Allowed Types in Switch: In Java, the switch statement accepts only certain types such as int, byte, short, char, their corresponding wrapper classes, enum types, and String (since Java 7). The type double is not permitted.

Why Not Double?: The switch statement is designed for discrete values. Floating-point numbers (like double) can have precision issues, making them unsuitable for switch-case comparisons.

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Switch25.java
Switch25.java:5: error: patterns in switch statements are a preview feature and are disabled by default.
    switch(score) {
    ^
    (use --enable-preview to enable patterns in switch statements)
Switch25.java:6: error: constant label of type int is not compatible with switch selector type double
    case 100:
    ^
Switch25.java:9: error: constant label of type int is not compatible with switch selector type double
    case 85:
    ^
3 errors
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

```
//Corrected code-
public class Switch25 {
    public static void main(String[] args) {
        int score = 85;
        switch(score) {
            case 100:
                System.out.println("Perfect score!");
                break;
            case 85:
                System.out.println("Great job!");
                break;
            default:
                System.out.println("Keep trying!");
        }
    }
}
```

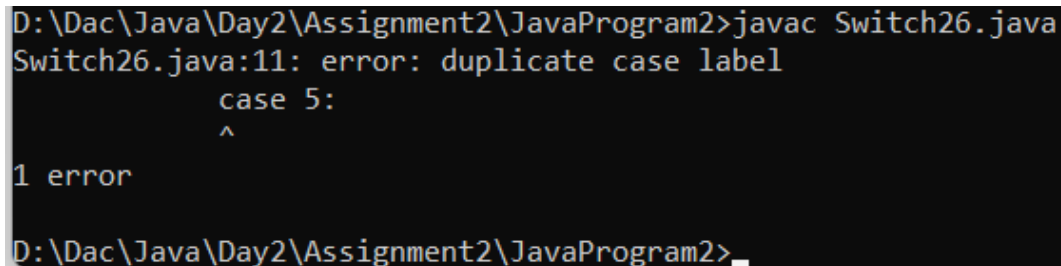
```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Switch25.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java Switch25.java
Great job!
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```

Snippet 26:

```
public class Switch {  
    public static void main(String[] args) {  
        int number = 5;  
        switch(number) {  
            case 5:  
                System.out.println("Number is 5");  
  
                break;  
            case 5:  
                System.out.println("This is another case 5");  
                break;  
            default:  
                System.out.println("This is the default case");  
        }  
    }  
}
```

- **Error to Investigate:** Why does the compiler complain about duplicate case labels? What happens when you have two identical case labels in the same switch block?

Ans-

A screenshot of a command prompt window showing the compilation of a Java file. The command 'javac Switch26.java' is entered. The output shows an error: 'Switch26.java:11: error: duplicate case label' followed by 'case 5:' and a caret '^' pointing to the second 'case 5:'. Below this, it says '1 error'. The prompt then shows 'D:\Dac\Java\Day2\Assignment2\JavaProgram2>' with a cursor.

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac Switch26.java  
Switch26.java:11: error: duplicate case label  
        case 5:  
        ^  
1 error  
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

The compiler complains about duplicate case labels because each case in a switch must be associated with a unique constant value. When you have two cases with the same label (in this case, both are `case 5:`), it creates ambiguity and violates Java's rules. As a result, the compiler throws an error indicating that the case label is duplicated.

In a switch statement, if duplicate labels were allowed, the runtime would not be able to determine which block of code to execute when the case value matches, leading to logical inconsistencies. That's why Java enforces uniqueness for case labels within the same switch block.

Section 2: Java Programming with Conditional Statements

Question 1: Grade Classification

Write a program to classify student grades based on the following criteria:

- If the score is greater than or equal to 90, print "A"
- If the score is between 80 and 89, print "B"
- If the score is between 70 and 79, print "C"
- If the score is between 60 and 69, print "D"
- If the score is less than 60, print "F"

Ans-

```
import java.util.Scanner;

public class GradeClassification {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the student's score: ");
        int score = scanner.nextInt();

        if(score >= 90) {
            System.out.println("A");
        } else if(score >= 80) {
            System.out.println("B");
        } else if(score >= 70) {
            System.out.println("C");
        } else if(score >= 60) {
            System.out.println("D");
        } else {
            System.out.println("F");
        }

        scanner.close();
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac GradeClassification.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java GradeClassification.java
Enter the student's score: 72
C
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Question 2: Days of the Week

Write a program that uses a nested switch statement to print out the day of the week based on an integer input (1 for Monday, 2 for Tuesday, etc.). Additionally, within each day, print whether it is a weekday or weekend.

Ans-

```
import java.util.Scanner;

public class DaysOfWeek {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number for the day of the week (1 for Monday, ..., 7 for Sunday): ");
        int day = scanner.nextInt();

        switch(day) {
            // For days 1-5, these are weekdays.
            case 1:
            case 2:
            case 3:
            case 4:
            case 5:
                // Nested switch to print the day name for weekdays.
                switch(day) {
                    case 1:
                        System.out.println("Monday");
                        break;
                    case 2:
                        System.out.println("Tuesday");
                        break;
                    case 3:
                        System.out.println("Wednesday");
                        break;
                    case 4:
                        System.out.println("Thursday");
                        break;
                    case 5:
                        System.out.println("Friday");
                        break;
                }
                System.out.println("Weekday");
                break;

            // For days 6-7, these are weekends.
            case 6:
            case 7:
                // Nested switch to print the day name for weekends.
                switch(day) {
                    case 6:
                        System.out.println("Saturday");
                        break;
                    case 7:
                        System.out.println("Sunday");
                        break;
                }
                System.out.println("Weekend");
                break;

            default:
                System.out.println("Invalid day");
                break;
        }
    }
}
```



```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac DaysOfWeek.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java DaysOfWeek.java
Enter a number for the day of the week (1 for Monday, ..., 7 for Sunday): 2
Tuesday
Weekday
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Question 3: Calculator

Write a program that acts as a simple calculator. It should accept two numbers and an operator (+, -, *, /) as input. Use a switch statement to perform the appropriate operation. Use nested if-else to check if division by zero is attempted and display an error message.

Ans-

```
import java.util.Scanner;

public class SimpleCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Prompt the user for two numbers and an operator.
        System.out.print("Enter the first number: ");
        double num1 = scanner.nextDouble();

        System.out.print("Enter the second number: ");
        double num2 = scanner.nextDouble();

        System.out.print("Enter an operator (+, -, *, /): ");
        char operator = scanner.next().charAt(0);

        double result = 0;
        boolean validOperation = true;

        // Use a switch statement to perform the appropriate operation.
        switch(operator) {
            case '+':
                result = num1 + num2;
                break;

            case '-':
                result = num1 - num2;
                break;

            case '*':
                result = num1 * num2;
                break;

            case '/':
                // Check for division by zero using a nested if-else.
                if(num2 != 0) {
                    result = num1 / num2;
                } else {
                    System.out.println("Error: Division by zero is not allowed.");
                    validOperation = false;
                }
                break;

            default:
                System.out.println("Error: Invalid operator.");
                validOperation = false;
        }

        // Display the result if a valid operation was performed.
        if(validOperation) {
            System.out.println("Result: " + result);
        }
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac SimpleCalculator.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java SimpleCalculator.java
Enter the first number: 10
Enter the second number: 20
Enter an operator (+, -, *, /): *
Result: 200.0
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Question 4: Discount Calculation

Write a program to calculate the discount based on the total purchase amount. Use the following criteria:

- If the total purchase is greater than or equal to Rs.1000, apply a 20% discount.
- If the total purchase is between Rs.500 and Rs.999, apply a 10% discount.
- If the total purchase is less than Rs.500, apply a 5% discount.

Additionally, if the user has a membership card, increase the discount by 5%.

Ans-

```
import java.util.Scanner;

public class DiscountCalculation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Prompt the user for the total purchase amount
        System.out.print("Enter the total purchase amount (Rs.): ");
        double totalPurchase = scanner.nextDouble();

        // Prompt the user for membership status (true if the user has a membership card)
        System.out.print("Do you have a membership card? (true/false): ");
        boolean hasMembership = scanner.nextBoolean();

        // Determine the base discount percentage based on the purchase amount
        double discountPercent = 0.0;
        if (totalPurchase >= 1000) {
            discountPercent = 20;
        } else if (totalPurchase >= 500) {
            discountPercent = 10;
        } else {
            discountPercent = 5;
        }

        // Increase discount by 5% if the user has a membership card
        if (hasMembership) {
            discountPercent += 5;
        }

        // Calculate the discount amount and final amount after discount
        double discountAmount = totalPurchase * discountPercent / 100;
        double finalAmount = totalPurchase - discountAmount;

        // Display the discount details
        System.out.println("Discount Percentage: " + discountPercent + "%");
        System.out.println("Discount Amount: Rs. " + discountAmount);
        System.out.println("Final Amount to be paid: Rs. " + finalAmount);

        scanner.close();
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac DiscountCalculation.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java DiscountCalculation.java
Enter the total purchase amount (Rs.): 100
Do you have a membership card? (true/false): true
Discount Percentage: 10.0%
Discount Amount: Rs. 10.0
Final Amount to be paid: Rs. 90.0
D:\Dac\Java\Day2\Assignment2\JavaProgram2>
```

Question 5: Student Pass/Fail Status with Nested Switch

Write a program that determines whether a student passes or fails based on their grades in three subjects. If the student scores more than 40 in all subjects, they pass. If the student fails in one or more subjects, print the number of subjects they failed in.

Ans-

```
import java.util.Scanner;

public class StudentPassFail {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Read marks for three subjects
        System.out.print("Enter marks for subject 1: ");
        int subject1 = scanner.nextInt();
        System.out.print("Enter marks for subject 2: ");
        int subject2 = scanner.nextInt();
        System.out.print("Enter marks for subject 3: ");
        int subject3 = scanner.nextInt();

        // Count the number of subjects failed (score must be greater than 40 to pass)
        int failCount = 0;
        if(subject1 <= 40) {
            failCount++;
        }
        if(subject2 <= 40) {
            failCount++;
        }
        if(subject3 <= 40) {
            failCount++;
        }

        // Use an outer switch based on the number of failures
        switch(failCount) {
            case 0:
                System.out.println("Student passes");
                break;
            default:
                // Nested switch to print detailed failure message based on number of failed subjects
                switch(failCount) {
                    case 1:
                        System.out.println("Student fails in 1 subject");
                        break;
                    case 2:
                        System.out.println("Student fails in 2 subjects");
                        break;
                    case 3:
                        System.out.println("Student fails in all subjects");
                        break;
                }
                break;
        }

        scanner.close();
    }
}
```

```
D:\Dac\Java\Day2\Assignment2\JavaProgram2>javac StudentPassFail.java
D:\Dac\Java\Day2\Assignment2\JavaProgram2>java StudentPassFail.java
Enter marks for subject 1: 52
Enter marks for subject 2: 72
Enter marks for subject 3: 92
Student passes
D:\Dac\Java\Day2\Assignment2\JavaProgram2>_
```