CDAC MUMBAI
Lab Assignment

SECTION 1: Error-Driven Learning Assignment: Loop Errors

Instructions:

Analyze each code snippet for errors or unexpected behavior. For each snippet, determine:

1. Why does the error or unexpected behavior occur?
2. How can the code be corrected to achieve the intended behavior?

Snippet 1:

```
public class InfiniteForLoop {
   public static void main(String[] args) {
      for (int i = 0; i < 10; i--) {
         System.out.println(i);
      }
   }
}
// Error to investigate: Why does this loop run infinitely? How should the loop control variable be adjusted?
```

```
public class InfiniteForLoop {
    public static void main(String[] args) {
        for (int i = 0; i < 10; i++) {
        System.out.println(i);
    }
}
}
```

Snippet 2:

```
public class IncorrectWhileCondition {
   public static void main(String[] args) {
      int count = 5;
      while (count = 0) {
         System.out.println(count);
         count--;
      }
   }
}
// Error to investigate: Why does the loop not execute as expected? What is the issue with the condition in the
`while` loop?
```

```
public class InfiniteForLoop {
    public static void main(String[] args) {
        for (int i = 0; i < 10; i--) { // Decrementing i instead of incrementing
it
            System.out.println(i);
```

```
        }
    }
}
```

Snippet 3:

```
public class DoWhileIncorrectCondition {
    public static void main(String[] args) {
        int num = 0;
        do {
            System.out.println(num);
            num++;
        } while (num > 0);
```

}
}
// Error to investigate: Why does the loop only execute once? What is wrong with the loop condition in the `do-while` loop?

```
public class DoWhileIncorrectCondition { // No hyphen or spaces in the class name
    public static void main(String[] args) {
        int num = 0;
        do {
            System.out.println(num);
            num++;
        } while (num > 0); // The condition is incorrect
    }
}
```

Snippet 4:

```
public class OffByOneErrorForLoop {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
        // Expected: 10 iterations with numbers 1 to 10
        // Actual: Prints numbers 1 to 10, but the task expected only 1 to 9
    }
}
// Error to investigate: What is the issue with the loop boundaries? How should the loop be adjusted to meet the expected output?
public class OffByOneErrorForLoop {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) { // The loop runs 10 times
            System.out.println(i);
        }
    }
}
```

Snippet 5:

```
public class WrongInitializationForLoop {
    public static void main(String[] args) {
        for (int i = 10; i >= 0; i++) {
            System.out.println(i);
        }
    }
}
// Error to investigate: Why does this loop not print numbers in the expected order? What is the problem with the initialization and update statements in the `for` loop?
public class WrongInitializationForLoop {
    public static void main(String[] args) {
```

```
        for (int i = 10; i >= 0; i++) { // The loop should decrement, not
increment
            System.out.println(i);
        }
    }
}
```

Snippet 6:

```
public class MisplacedForLoopBody {
   public static void main(String[] args) {
      for (int i = 0; i < 5; i++)
         System.out.println(i);
         System.out.println("Done");
   }
}
// Error to investigate: Why does "Done" print only once, outside the loop? How should the loop body be enclosed to
include all statements within the loop?
```

```
public class MisplacedForLoopBody {
    public static void main(String[] args) {
        for (int i = 0; i < 5; i++)
            System.out.println(i); // Only this line is part of the loop
        System.out.println("Done"); // This line is outside the loop
    }
}
```

Snippet 7:

```
public class UninitializedWhileLoop {
   public static void main(String[] args) {
      int count;
```

```
        while (count < 10) {
            System.out.println(count);
            count++;
        }
    }
}
```

```
// Error to investigate: Why does this code produce a compilation error? What needs to be done to initialize the loopvariable properly?
public class UninitializedWhileLoop {
    public static void main(String[] args) {
    int count = 0;  // Initialize count
    while (count < 10) {
    System.out.println(count);
    count++;
    }
    }
   }
```

Snippet 8:

```
public class OffByOneDoWhileLoop {
    public static void main(String[] args) {
        int num = 1;
        do {
            System.out.println(num);
            num--;
        } while (num > 0);
    }
}
```

```
// Error to investigate: Why does this loop print unexpected numbers? What adjustments are needed to print thenumbers from 1 to 5? public class OffByOneDoWhileLoop {
    public static void main(String[] args) {
        int num = 1;
        do {
            System.out.println(num);
            num--;
        } while (num > 0); // The condition should be adjusted
    }
}
```

Snippet 9:

```
public class InfiniteForLoopUpdate {
    public static void main(String[] args) {
        for (int i = 0; i < 5; i += 2) {
            System.out.println(i);
        }
    }
```

```
}
// Error to investigate: Why does the loop print unexpected results or run infinitely? How should the loop updateexpression be corrected? public class InfiniteForLoopUpdate {
    public static void main(String[] args) {
        for (int i = 0; i < 5; i += 2) { // The loop might terminate early or not as expected
            System.out.println(i);
        }
    }
}
```

Snippet 10:

```
public class IncorrectWhileLoopControl {
   public static void main(String[] args) {
      int num = 10;
      while (num = 10) {
         System.out.println(num);
         num--;
      }
   }
}
// Error to investigate: Why does the loop execute indefinitely? What is wrong with the loop condition?
public class IncorrectWhileLoopControl {
    public static void main(String[] args) {
        int num = 10;
        while (num <= 10) { // Assignment instead of comparison
            System.out.println(num);
            num--;
        }
    }
}
```

Snippet 11:

```
public class IncorrectLoopUpdate {
   public static void main(String[] args) {
      int i = 0;
      while (i < 5) {
         System.out.println(i);
         i += 2; // Error: This may cause unexpected results in output
      }
   }
}
// Error to investigate: What will be the output of this loop? How should the loop variable be updated to achieve the desired result?
```

```java
public class IncorrectLoopUpdate {
    public static void main(String[] args) {
        int i = 0;
        while (i < 5) {
            System.out.println(i);
            i += 2; // Skips numbers, printing only even numbers
        }
    }
}
```

Snippet 12:

```
public class LoopVariableScope {
   public static void main(String[] args) {
      for (int i = 0; i < 5; i++) {
         int x = i * 2;
      }
      System.out.println(x); // Error: 'x' is not accessible here
   }
}
// Error to investigate: Why does the variable 'x' cause a compilation error? How does scope
```

```java
public class LoopVariableScope {
    public static void main(String[] args) {
        int x = 0;
        for (int i = 0; i < 5; i++) {
             x = i * 2;
        }
        System.out.println(x); // 'x' is out of scope here
    }
}
```

SECTION 2: Guess the Output

Instructions:

1. **Perform a Dry Run:** Carefully trace the execution of each code snippet manually to determine

the output.

2. **Write Down Your Observations:** Document each step of your dry run, including the values of variables at each stage of execution.
3. **Guess the Output:** Based on your dry run, provide the expected output of the code.
4. **Submit Your Assignment:** Provide your dry run steps along with the guessed output for each code snippet.

Snippet 1:

```
public class NestedLoopOutput {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 2; j++) {
                System.out.print(i + " " + j + " ");
            }
            System.out.println();
```

```
        }
    }
}
// Guess the output of this nested loop.
1 1 1 2
2 1 2 2
3 1 3 2
```

Snippet 2:

```
public class DecrementingLoop {
    public static void main(String[] args) {
        int total = 0;
        for (int i = 5; i > 0; i--) {
            total += i;
            if (i == 3) continue;
            total -= 1;
        }
        System.out.println(total);
    }
}
// Guess the output of this loop.
11
```

Snippet 3:

```
public class WhileLoopBreak {
    public static void main(String[] args) {
        int count = 0;
        while (count < 5) {
            System.out.print(count + " ");
            count++;
            if (count == 3) break;
        }
        System.out.println(count);
    }
}
// Guess the output of this while loop.
0 1 2 3
```

Snippet 4:

```
public class DoWhileLoop {
    public static void main(String[] args) {
        int i = 1;
        do {
            System.out.print(i + " ");
            i++;
        } while (i < 5);
        System.out.println(i);
    }
}
// Guess the output of this do-while loop.
1 2 3 4 5
```

Snippet 5:

```
public class ConditionalLoopOutput {
   public static void main(String[] args) {
      int num = 1;
      for (int i = 1; i <= 4; i++) {
         if (i % 2 == 0) {
            num += i;
         } else {
            num -= i;
         }
      }
      System.out.println(num);
   }
}
// Guess the output of this loop.
3
```

Snippet 6:

```
public class IncrementDecrement {
   public static void main(String[] args) {
      int x = 5;
      int y = ++x - x-- + --x + x++;
      System.out.println(y);
   }
}
// Guess the output of this code snippet.
8
```

Snippet 7:

```
public class NestedIncrement {
   public static void main(String[] args) {
      int a = 10;
      int b = 5;
      int result = ++a * b ----- a + b++;
      System.out.println(result);
   }
}
// Guess the output of this code snippet.
71
```

Snippet 8:

```
public class LoopIncrement {
   public static void main(String[] args) {
      int count = 0;
      for (int i = 0; i < 4; i++) {
         count += i++ - ++i;
      }
      System.out.println(count);
   }
```

}
// Guess the output of this code snippet.
-4

SECTION 3: Lamborghini Exercise:

Instructions:

1. **Complete Each Program:** Write a Java program for each of the tasks listed below.
2. **Test Your Code:** Make sure your code runs correctly and produces the expected output.
3. **Submit Your Solutions:** Provide the complete code for each task along with sample output.

Tasks:

1. Write a program to calculate the sum of the first 50 natural numbers.
2. Write a program to compute the factorial of the number 10.
3. Write a program to print all multiples of 7 between 1 and 100.
4. Write a program to reverse the digits of the number 1234. The output should be 4321.
5. Write a program to print the Fibonacci sequence up to the number 21.
6. Write a program to find and print the first 5 prime numbers.
7. Write a program to calculate the sum of the digits of the number 9876. The output should be30 (9 + 8 + 7 + 6).
8. Write a program to count down from 10 to 0, printing each number.
9. Write a program to find and print the largest digit in the number 4825.
10. Write a program to print all even numbers between 1 and 50.
11. Write a Java program to demonstrate the use of both pre-increment and post-decrement operators in a single expression
12. Write a program to draw the following pattern:

```
*****
*****
*****
*****
*****
```

13. Write a program to print the following pattern:

```
1
2*2
3*3*3
4*4*4*4
5*5*5*5*5
5*5*5*5*5
4*4*4*4
3*3*3
2*2
```

1

14. Write a program to print the following pattern:

```
*
 **
 ***
 *****
 *******
 *********
```

15. Write a program to print the following pattern:

```
   *
  **
 ***
****
*****
```

16. Write a program to print the following pattern:

```
   *
  ***
 *****
*******
*********
```

17. Write a program to print the following pattern:

```
*****
 ****
  ***
   **
    *
```

18. Write a program to print the following pattern:

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

19. Write a program to print the following pattern:

```
1
1*2
1*2*3
1*2*3*4
1*2*3*4*5
```

20. Write a program to print the following pattern:

```
5
5*4
5*4*3
5*4*3*2
5*4*3*2*1
```

21. Write a program to print the following pattern:

```
1
1*3
1*3*5
1*3*5*7
1*3*5*7*9
```

22. Write a program to print the following pattern:

```
*********
 *******
  *****
   ***
    *
   ***
  *****
 *******
*********
```

23. Write a program to print the following pattern:

```
11111
22222
33333
44444
55555
```

24. Write a program to print the following pattern:

```
1
22
333
4444
55555
```

25. Write a program to print the following pattern:

```
1
12
123
1234
12345
```

26. Write a program to print the following pattern:

1
2 3
4 5 6
7 8 9 10
11 12 13 14 15