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?

**Error Explanation:**

* The loop runs infinitely because the loop control variable i is being decremented (i--) instead of incremented. This means i will never reach 10, causing the loop to run indefinitely.

**Correction:**

* Change i-- to i++ to correctly increment the loop variable:

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?

**Error Explanation:**

* The loop condition count = 0 is an assignment, not a comparison. The assignment will always set count to 0, which evaluates to false, so the loop never executes.

**Correction:**

* Use == for comparison:

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 `dowhile` loop?

**Error Explanation:**

* The loop condition num > 0 will become false after the first increment, causing the loop to execute only once.

**Correction:**

* If you want the loop to run while num is less than a certain value, change the condition:

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?

**Error Explanation:**

* The loop runs correctly for numbers 1 to 10, but if the expected output was only 1 to 9, then the loop's condition is off by one.

**Correction:**

* Adjust the loop condition to stop at 9:

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?

**Error Explanation:**

* The loop is intended to count down from 10 to 0, but the loop control variable i is not being decremented, so the loop condition i >= 0 is always true.

**Correction:**

* Decrement i in the loop control:

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?

**Error Explanation:**

* The "Done" statement is outside the loop because it is not enclosed within braces {}. This causes "Done" to print only once after the loop completes.

**Correction:**

* Enclose the loop body in braces:

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 loop variable properly?

**Error Explanation:**

* The variable count is not initialized, leading to a compilation error.

**Correction:**

* Initialize count before using it:

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 the numbers from 1 to 5?

**Error Explanation:**

* The loop condition num > 0 causes the loop to run once and then exit because num is decremented to 0 after the first iteration.

**Correction:**

* To print numbers from 1 to 5, you need to start from 5 and decrement or adjust the condition to increment up to 5:

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 update expression be corrected?

**Error Explanation:**

* The loop does not run infinitely, but it will print 0, 2, and 4, which might be unexpected. The loop ends after i reaches 6.

**Correction:**

* This loop works as expected, but if you want to print all numbers from 0 to 4, you should increment i by 1:

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?

**Error Explanation:**

* The condition num = 10 is an assignment, not a comparison, causing an infinite loop since num is always set to 10.

**Correction:**

* Use == for comparison:

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?

**Error Explanation:**

* The loop prints 0, 2, and 4 because i is incremented by 2. If you expected a different output (like 0, 1, 2, 3, 4), the increment step is too large.

**Correction:**

* Increment i by 1 if you want to print every number:

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

**Error Explanation:**

* The variable x is declared inside the loop and is not accessible outside of it, causing a compilation error.

**Correction:**

* Declare x outside the loop if you need to access it later:

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.

**Output:**

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.

Output:

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.

0123

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.

12345

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.

49

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.

-9

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 be 30 (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:

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

* 1. 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

* 1. Write a program to print the following pattern:

\*

\*\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

* 1. Write a program to print the following pattern:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

* 1. Write a program to print the following pattern:

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

* 1. Write a program to print the following pattern:

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

* 1. Write a program to print the following pattern:

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

* 1. Write a program to print the following pattern:

1

1\*2

1\*2\*3

1\*2\*3\*4

1\*2\*3\*4\*5

* 1. Write a program to print the following pattern:

5

5\*4

5\*4\*3

5\*4\*3\*2

5\*4\*3\*2\*1

* 1. Write a program to print the following pattern:

1

1\*3

1\*3\*5

1\*3\*5\*7

1\*3\*5\*7\*9

* 1. Write a program to print the following pattern:

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

* 1. Write a program to print the following pattern:

11111

22222

33333

44444

55555

* 1. Write a program to print the following pattern:

1

22

333

4444

55555

* 1. Write a program to print the following pattern:

1

12

123

1234

12345

* 1. Write a program to print the following pattern:

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15