SECTION - 1 : Error-Driven Learning Assignment: Loop Errors

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?

   Because ' i ' is initialized to 0 and as per code, we are decrementing it, Instead, we can increment it by doing ' i++ '

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: incompatible types: int cannot be converted to boolean
   while (count = 0) {

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?

   The issue is that the condition num > 0 is only true for positive numbers, but the loop is intended to run at least once for num = 0.
   To fix this, you can change the condition to num >= 0, which includes 0

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?

   Instead of i<=10; we can simply do i<10. In this way, it will print only numbers from 1 to 9

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?

   The problem is that the initialization int i = 10 starts the loop from 10, but the update statement i++ increments i instead of decrementing it. To fix this, you should change the update statement to i-- to decrement i.
   for (int i = 10; i >= 0; i--)
   This way, the loop will print numbers in the expected order: 10, 9, 8, ..., 0.

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?

   The "Done" statement prints only once because it is outside the loop body. In Java, the loop body must be enclosed in curly braces {} to include multiple statements within the loop.

   Without the curly braces, only the immediate next statement (System.out.println(i);) is considered part of the loop body. The "Done" statement is not part of the loop and is executed only once after the loop finishes.

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?

   The code produces a compilation error because the variable count is not initialized before being used in the while loop condition.
   By initializing count to 0, you ensure that the loop starts with a valid value and will run 10 times, printing numbers from 0 to 9.

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?

   As the num is initialized with num=1, it will be only true for the same, but as we want to print numbers from 1 to 5, we can simply initialize them with num = 5

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?

   There's no error in the code, it is running smoothly,
   The output of the code is: 0 2 4

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?

    The loop executes indefinitely because the loop condition num = 10 is an assignment, not a comparison. The single equals sign (=) assigns the value 10 to num, making the condition always true.

    In each iteration, num is assigned 10, printed, and then decremented. However, the condition checks the assignment num = 10, which is always true, so the loop never terminates.

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?

    It is correct. There's nothing wrong with the code with its output being 0 2 4.

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?

    The variable x causes a compilation error because it is declared inside the for-loop block, making it a local variable with scope limited to that block.
    To print the values of x inside the loop, we can move the System.out.println(x) statement inside the loop.

SECTION - 2 : Guess the Output.

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();
     }
    }
   }

   //Output :
   1 1 1 2
   2 1 2 2
   3 1 3 2

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

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
```

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
```

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
```

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
```

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
```

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
```