Assignment 3

SECTION 1: Error-Driven Learning Assignment: Loop Errors

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?

1. **Why does the error or unexpected behavior occur?**
   -> The loop goes into infinity. Error occurred because we initialized int i to 0. & every time when loop executes the condition it becomes true because the i value is less than 10 and it decrementing the value i value eveytime.
2. **How can the code be corrected to achieve the intended behavior?**

   -> If we want to achieve intended output just increment i instead decrementing.

Corrected code:

```
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:

```
IncorrectWhileCondition.java:4: error: incompatible types: int cannot be converted to
boolean
 while (count = 0) {
          ^
1 error
```

Explanation: count = 0 is an assignment statement, which assigns 0 to count. The while loop expects a boolean condition, but count = 0 is of type int, leading to the error. The condition should be a comparison (>), not an assignment (=).

Corrected code:

```
public class IncorrectWhileCondition {
 public static void main(String[] args) {
 int count = 5;
 while (count > 0)
{
 System.out.println(count);
 count--;
 }
 }
}
```

Snippet 3:

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

Output: The loop goes into infinity.

Explanation : First num has 0 value. When it goes into do block as it prints value of num then on next line it increments the num by 1. Which means num is now 1 & the while condition num>0 becomes true. Because the value is incrementing everytime it is greater than 0 & goes into infinity state.

Corrected code:

```
public class DoWhileIncorrectCondition {
 public static void main(String[] args) {
                int num = 0;
                do {
                        System.out.println(num);
                        num++;
                 } while (num < 5);
 }
}
```

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
 }
}
```

Output:

1
2
3
4
5
6
7
8
9
10

Explanation: Because in for loop we are starting it from 1 and continued till i is less than equal to 10. We should give only less than(<) 10 to loop it from 1 to 9.

Corrected output:

```
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
}
}
```

Snippet 5:

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

Output: Infinity values from 10 ...

Explanation: As we assigned the value of i to 10 & checking only if it greater than 0 & incrementing everytime. The loop condition will becomes true everytime because the value of i incrementing from 10..11..12..13... likewise it goes to infinity because the condition never becomes false.

Corrected code:

```
public class WrongInitializationForLoop {
 public static void main(String[] args) {
 for (int i = 10; i >= 0; i--) {
 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");
 }
}
```

Output:

0

1

2

3

4

Done

Explanation: The for-loop executes only the next single statement if curly braces {} are not used. That's why when cursor comes out of loop then the next print statement is printed only one time. We must provide curly braces for the body if we want multiple statements to executed.

Corrected code:

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

Snippet 7:

```
public class UninitializedWhileLoop {
 public static void main(String[] args) {
 int count;
 while (count < 10) {
```

```
System.out.println(count);
count++;
}
}
}
```

Error: UninitializedWhileLoop.java:4: error: variable count might not have been initialized

```
while (count < 10) {
    ^
```

Explanation: Compilation error because the variable count is declared but not initialized before being used in the while condition. In Java, local variables (like count inside main()) do not get a default value. Before using count, Java requires it to be explicitly initialized, otherwise, the compiler throws an error. Instance variables (fields in a class) get default values, but local variables do not. We need to assign value to i if we want the code to work.

Corrected code:

```
public class UninitializedWhileLoop {
public static void main(String[] args) {
int count=0;
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);
 }
}
```

Output : 1

Why does this loop print unexpected numbers? What adjustments are needed to print the numbers from 1 to 5?

Explanation : num is holding value 1. When cursor moves to do block it prints the value 1 & decrements by 1, which then becomes 0. Checking for while condition 0>0 no so the condition becomes false and the cursor comes out of the loop.

If we want to print numbers till 5 change the decrement to increment operation & while condition from num>0 change to num<=5.

Corrected code:

```
public class OffByOneDoWhileLoop {
 public static void main(String[] args) {
 int num = 1;
 do {
 System.out.println(num);
 num++;
 } while (num <=5);
 }
}
```

Snippet 9:

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

```
}
```

Output: 0

2

4

Explanation : Everytime when loop checks condition prints the value of i & increments with +2. Initially i is 0, +2=2, +2=4 then 5<5 becomes false & loop ends. If we want numbers from 0 to 4 we can change the incrementation to just ++.

Corrected code:

```
public class InfiniteForLoopUpdate {
 public static void main(String[] args) {
 for (int i = 0; i < 5; i ++) {
 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:

IncorrectWhileLoopControl.java:4: error: incompatible types: int cannot be converted to boolean

```
 while (num = 10) {
```

^

1 error

Explanation : num = 10 is an assignment statement, which assigns 10 to num. The while loop expects a boolean condition, but num = 10 is of type int, leading to the error. The condition should be a comparison (>=), not an assignment (=).

Corrected code:

```
public class IncorrectWhileLoopControl {
 public static void main(String[] args) {
 int num = 10;
 while (num >= 1) {
 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
 }
 }
}
```

Output: 0

2

4

Explanation : Incrementing i everytime with 2 is making the code to not generate desired output. Just make it +1/++ to get desired output.

Corrected code:

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

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: LoopVariableScope.java:6: error: cannot find symbol
        System.out.println(x); // Error: 'x' is not accessible here
                           ^
         symbol:   variable x
         location: class LoopVariableScope
        1 error
```

Explanation : Compilation error because the variable x is declared inside the for-loop block, making it out of scope when accessed outside the loop. A variable is accessible only within the block {} where it is declared. x is declared inside the for-loop block, meaning it only exists within that block. Once the loop ends, x is destroyed, making it inaccessible outside.

If we want the code to work either we can write SOP statement inside loop or declare the x variable outside for loop.

Corrected code:

```
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
 }
}
```