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?
```

Solution→ this loop will run infinitely because in for loop condition for i always turns out to be true.

The loop control can be adjusted by changing the increment/decrement part in for loop i.e. i—to i++.

```
class loop {
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?
```

Solution→ here the condition un while loop is wrong. Here the condition should result in some Boolean value but here assign is done using = operator instead of = we should use == ,>,<,>=,<=.

```
class loop {
 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);
```

Solution→ actually this code will execute infinitely because after executing the do part and printing num value and incrementing the num value when while condition is checked it always turns out to be true.

In order to execute the above code one time only while condition needs to be changed

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

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?
```

Solution→here the loop is executed till 10 i.e. for i= 10 the condition evaluated to true and 10 is printed. So in order to meet the expected result i.e. to print number 1 to 9. The condition under for loop needs to be changed.

```
class loop {

public static void main(String[] args) {

 for (int i = 1; i < 10; i++) {

 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?
```

Solution→ here the code will execute infinite number of times because in for loop condition for i always evaluates to true because after updation i always remains >=0. So in order to print the expected result updation should be changed.

```
class loop {

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");
    }
}
// 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?
```

Solution→ here the { } are not used properly because of which desired output is not printed.

```
class loop {
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 to investigate: Why does this code produce a compilation error? What needs to be done to initialize the loop variable properly?
```

Solution→ here the compilation error occurs because the variable named count is not initialized with any value. So in order to execute the above code count value should be initialized with some value.

```
class loop {

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);
   }
}
// Error to investigate: Why does this loop print unexpected numbers? What adjustments are needed to print the
numbers from 1 to 5?
```

Solution→ this code is printing the unexpected value because the updation and conditional statement is not correct. So in order to print the number between 1 to 5 the code should be.

```
class loop {

 public static void main(String[] args) {

 int num = 1;

 do {

 System.out.println(num);

 num++;

 } while (num < 6);

 }

}
```

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?
```

Solution → here the code runs perfectely and finite times. Its just it only print the even numbers including 0.

```
class loop {

 public static void main(String[] args) {

 for (int i = 0; i < 5; i += 2) {

 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?
```

Solution→ here the error occurs because the while loop condition is not resulting into a Boolean value. So in order to execute the code right way the condition must be changed to.

```
class loop {

public static void main(String[] args) {

 int num = 10;

 while (num == 10) {
```

```
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?
```

Solution→ here the code runs perfectely and finite times. Its just it only print the even numbers including 0.

The output is

0

1

2

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

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
```

Solution→ variable x causes a compilation error because compiler cant find the symbol x.

As print statement of x is written outside the for loop there doesn't exist any variable named x outside the for loop. So in order to print the values of x scopes should be changed.

```
class loop {

public static void main(String[] args) {

 for (int i = 0; i < 5; i++) {

 int x = i * 2;


 System.out.println(x);

 }

 }

 }
```

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();
        }
    }
}
```

Solution → 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.
```

Solution → 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.
```

Solution → 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.
```

Solution → 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.
```

Solution → 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.
```

Solution → 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.
```

Solution → 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);
    }
```

Solution → -4