Name: Shriniwas Vasant Pawar

Center: Kharghar

Email: pawar.shriniwas26@gmail.com

Snippet 1:

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

Why does this loop run infinitely? How should the loop control variable be adjusted?

Ans: This loop runs infinitely because 'i' decrements at every iteration by one and condition i<10 cannot be false throughout the loop. If we increment i by 1 In for loops increment section then loop will run 10 time and works fine.

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--;
    }
  }
}
```

Why does the loop not execute as expected? What is the issue with the condition in the `while` loop?

Ans: Incompatible types error occurs. It states that int cannot be converted to Boolean at while loop condition. In this code there is assignment operator is used. If we use ">" comparison operator instead of that then the code will run smoothly.

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);
  }
}
```

Why does the loop only execute once? What is wrong with the loop condition in the `do while` loop?

Ans: This loop runs infinitely in above code. If we want to execute exactly once then we should decrement num by 1 in do while block.

Corrected code:

```
public class DoWhileIncorrectCondition {
  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
  }
}
```

What is the issue with the loop boundaries? How should the loop be adjusted to meet the expected output?

Ans: In this code it prints numbers from 1 to 10 but expected output is numbers from 1 to 9. So to achieve this we should write i<10 in for condition.

Corrected code:

```
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);
    }
  }
}
```

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?

Ans: This loop goes infinite. Because i>=0 condition never goes false because I is incrementing in every iteration by 1. If we decrement I by 1 in update statement then code runs as expected.

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");
    }
}
```

Why does "Done" print only once, outside the loop? How should the loop body be enclosed to include all statements within the loop?

Ans: Here "Done" print only once because it is out side of the loop. In this code for loop doesn't have opening and closing curly brackets. So the statement which is below occurring first is considered inside loop. So to include all statements inside the loop we should write them inside the opening and closing curly brackets of that loop.

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++;
    }
  }
}
```

Why does this code produce a compilation error? What needs to be done to initialize the loop variable properly?

Ans: This code produce error: variable count might not have been initialized. It is syntax error. To solve this problem we should initialize the count variable to integer value like 0.

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);
  }
}
```

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

Ans: This code print number as 1 but to print the numbers from 1 to 5 we should increment num variable by 1 in do while block at end and in do while condition we should set 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);
    }
  }
}
```

Why does the loop print unexpected results or run infinitely? How should the loop update expression be corrected?

Ans: This loop will print numbers as: 0 2 4

Snippet 10:

```
public class IncorrectWhileLoopControl {
  public static void main(String[] args) {
    int num = 10;
    while (num = 10) {
      System.out.println(num);
      num--;
    }
  }
}
```

Why does the loop execute indefinitely? What is wrong with the loop condition?

Ans: incompatible type error occurs here. Here syntax error is occurred. We should give Equality operator in while like num == 10 to give input as Boolean to while because while only inputs Boolean condition.

Corrected code:

```
public class IncorrectWhileLoopControl {
  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
    }
  }
}
```

What will be the output of this loop? How should the loop variable be updated to achieve the desired result?

Ans: Output: 0 2 4 . To achieve the desired result in while loop we should increment I by 1 instead of i += 2. Then output will be 0 1 2 3 4.

Corrected code:

```
public class IncorrectLoopUpdate {
  public static void main(String[] args) {
    int i = 0;
```

```
        while (i < 5) {
            System.out.println(i);
            i += 1; // Error: This may cause unexpected results in output
        }
    }
}
```

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
    }
}
```

Why does the variable 'x' cause a compilation error? How does scope?

Ans: X variable cause compiler error because x is defined locally and we are accessing it outside of its scope.

SECTION 2: Guess the Output

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();
    }
  }
}
```

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);
  }
}
```

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);
    }
}
```

Output: 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);
    }
}
```

Output: 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);
    }
}
```

Output: 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);
  }
}
```

Output: 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);
  }
}
```

Output: 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);
  }
}
```

Output: - 4