Assignment 3

Snippet 1:

public class InfiniteForLoop {

public static void main(String[] args) {

for (int i = 0; i < 10; i--) {

System.out.println(i);

}

}

}



public class InfiniteForLoop {

public static void main(String[] args) {

for (int i = 0; i < 10; i++) {

System.out.println(i);

}

}

}

The loop control condition is i < 10, meaning the loop should run as long as i is less than 10.

Snippet 2:

public class IncorrectWhileCondition {

public static void main(String[] args) {

int count = 5;

while (count = 0) {

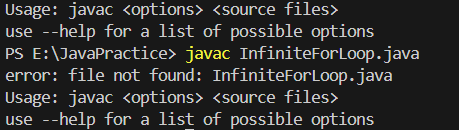
System.out.println(count);

count--;

}

}

}



Snippet 3:

public class DoWhileIncorrectCondition {

public static void main(String[] args) {

int num = 0;

do {

System.out.println(num);

num++;

} while (num > 0);



public class DoWhileIncorrectCondition

{

public static void main(String[] args) {

int num = 0;

do {

System.out.println(num);

num--;

} while (num > 0);

}

}

Reason:-

Snippet 4:

public class OffByOneErrorForLoop {

public static void main(String[] args) {

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

System.out.println(i);

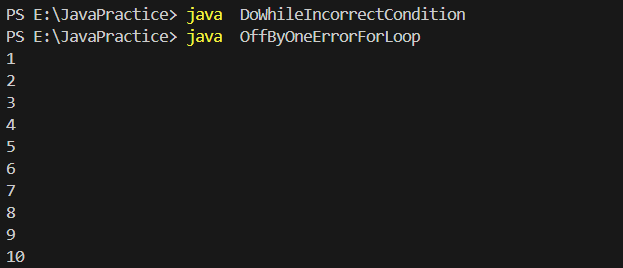
}

// Expected: 10 iterations with numbers 1 to 10

// Actual: Prints numbers 1 to 10, but the task expected only 1 to 9

}

}



public class OffByOneErrorForLoop {

public static void main(String[] args) {

for (int i = 1; i <= 9; i++) {

System.out.println(i);

}

}

}

Reason:

Snippet 5:

public class WrongInitializationForLoop {

public static void main(String[] args) {

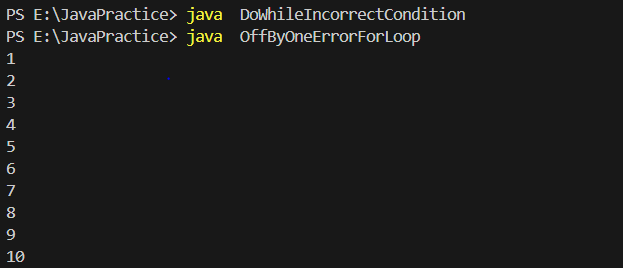
for (int i = 10; i >= 0; i++) {

System.out.println(i);

}

}

}



public class WrongInitializationForLoop {

public static void main(String[] args) {

for (int i = 0; i <= 10; i++) {

System.out.println(i);

}

}

}

Reason:-

Snippet 6:

public class MisplacedForLoopBody {

public static void main(String[] args) {

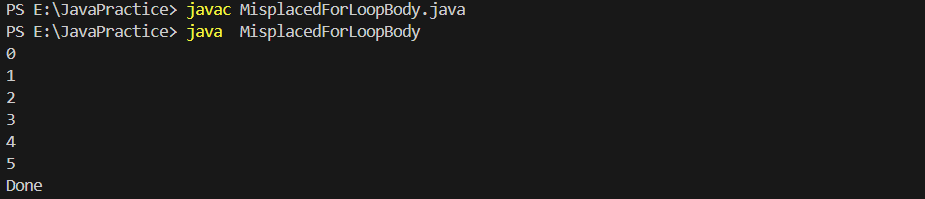
for (int i = 0; i < 5; i++)

System.out.println(i);

System.out.println("Done");

}

}



public class MisplacedForLoopBody {

public static void main(String[] args) {

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

System.out.println(i);

System.out.println("Done");

}

}

}

Reason:---

Snippet 7:

public class UninitializedWhileLoop {

public static void main(String[] args) {

int count;

while (count < 10) {

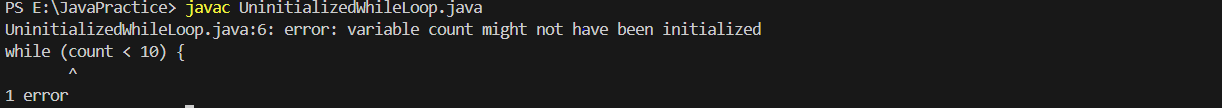
System.out.println(count);

count++;

}

}

}



public class UninitializedWhileLoop {

public static void main(String[] args) {

int count = 0;

while (count < 10) {

System.out.println(count);

count++;

}

}

}

Reason:-

Snippet 8:

public class OffByOneDoWhileLoop {

public static void main(String[] args) {

int num = 1;

do {

System.out.println(num);

num--;

} while (num > 0);

}

}



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) {

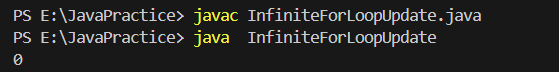
for (int i = 0; i < 5; i += 2) {

System.out.println(i);

}

}

}



public class InfiniteForLoopUpdate {

public static void main(String[] args) {

for (int i = 0; i <= 5; i += 1) {

System.out.println(i);

}

}

}

Reason:-

Snippet 10:

public class IncorrectWhileLoopControl {

public static void main(String[] args) {

int num = 10;

while (num = 10) {

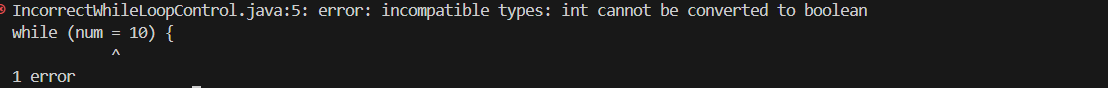
System.out.println(num);

num--;

}

}

}



public class IncorrectWhileLoopControl {

public static void main(String[] args) {

int num = 10;

while (num >= 10) {

System.out.println(num);

num--;

}

}

}

Reason:-

Snippet 11:

public class IncorrectLoopUpdate {

public static void main(String[] args) {

int i = 0;

while (i < 5) {

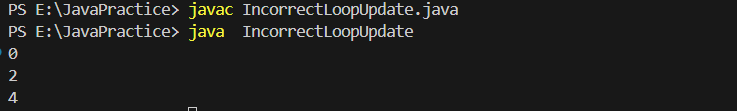
System.out.println(i);

i += 2; // Error: This may cause unexpected results in output

}

}

}



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

}

}

}

Reason:-

Snippet 12

public class LoopVariableScope {

public static void main(String[] args) {

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

int x = i \* 2;

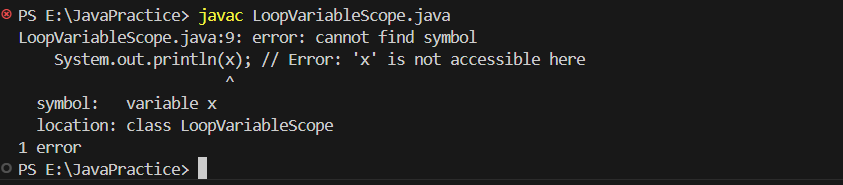
}

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



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

}

}

}

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 + " ");

}

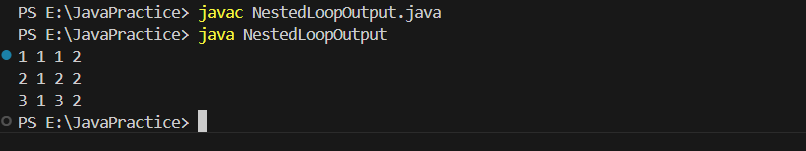
System.out.println();

}

}

}

// Guess the output of this nested loop.



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.

