Lab 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: loop runs infinte bcoz in each iteration i is decremented.. so it becomes more negative which means i will always be less than 10. That’s why the loop never terminates.

Corrected code:

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: count = 0 assigns the value 0 to count this will always evaluate to 0

Corrected code: To solve this we can use == comparison operator

int count = 5;

while (count != 0) { // Loop until count is not equal to 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);

}

}

Soln: At start num=0 then condition checked if num>0 i.e false

In do-while loop the body executes at least once, but since num is 0 after the first execution, the loop condition fails and no further iterations occur.

**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

}

}

Soln: change the loop condition so that it stops before reaching 10.. Use i < 10 instead of i <= 10

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

System.out.println(i);

}

// This will now print numbers from 1 to 9 as expected

**Snippet 5:**

public class WrongInitializationForLoop {

public static void main(String[] args) {

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

System.out.println(i);

}

}

}

The value of i keeps increasing bcoz the loop condition (i >= 0) remains true indefinitely once.. this will lead to an infinite loop because i never decreases

Corrected code:

for (int i = 10; i >= 0; i--) { // Use i-- to decrement 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: there are 2 statements inside the loop and braces are missing

Soln: To add multiple statements we need to enclose them in braces

Corrected code:

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: variable count is declared but not initialized before it is used in the while loop.

In Java, local variables must be initialized before they are used.

Corrected code:

int count = 0; // Initialize count to a starting value

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);

}

}

the loop starts by printing 1..then decrements num to 0 and prints it and finally exits the loop after printing 0 bcoz num is no longer greater than 0.

To print numbers from 1 to 5 increement num instead of decrement

Corrected code:

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);

}

}

}

Soln: here i+=2 means i=i+2 . The loop does not run infinitely ..it stops when I becomes 6

**Snippet 10:**

public class IncorrectWhileLoopControl {

public static void main(String[] args) {

int num = 10;

while (num = 10) {

System.out.println(num);

num--;

}

}

}

Error: in while loop condition is assignment of value to num instead of comparison ==

Corrected code:

int num = 10;

while (num > 0) { // Correct condition for looping

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

}

}

}

Loop will print

0

2

4

**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

}

}

The variable x causes a compilation error because it is out of scope when you try to access it outside the for loop.

Corrected code:

public class LoopVariableScope {

public static void main(String[] args) {

int x = 0; // Declare x outside the loop

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

x = i \* 2;

}

System.out.println(x); // Now x is 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();

}

}

}

Dry run:

1. 1st element of outer loop will work with inner loop’s all elements i.e ,

i =1 🡪 j = 1,2   
o/p 🡪1 1 1 2

1. Same goes here  
   i =2 🡪 j = 1, 2  
   o/p 🡪 2 1 2 2
2. i =3 🡪 j =1, 2  
   o/p 🡪 3 1 3 3

Final 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);

}

}

Dry run:

Total =0;

i=5; I > 0; i - -

1. i= 5  
   total += i 🡪 0+5 = 5  
   (i == 3) is false  
   total -= 1 🡪 5-1= 4  
   now total = 4
2. i= 4  
   total += i 🡪 4 + 4 =8  
   (i == 3) is false  
   total -= 1 🡪 8 – 1 =7  
   now total = 7
3. i= 3  
   total += i 🡪 7+3 = 10  
   (i == 3) is true  
   so total -= 1 will not execute  
   now total = 10
4. i= 2  
   total += i 🡪 10 + 2= 12  
   (i == 3) is false  
   total -= 1 🡪 12 -1 =11  
   now total = 11
5. i= 1  
   total += i 🡪 11 + 1 =12  
   (i == 3) is false  
   total -= 1 🡪 12 -1 = 11  
   now total = 11 🡨 final value 11

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);

}

}  
  
Dry run:

count = 0

1. count < 5 is true

(count + " ") 🡪 Prints 0

count++ 🡪 count = 1

(count == 3) is false

1. count < 5 is true

count + " ") 🡪 Prints 1

count++ 🡪 count = 2

(count == 3) is false

1. count < 5 is true

(count + " ") 🡪 Prints 2

count++ 🡪 count = 3

(count == 3) is true 🡪 break the loop

Output:

0 1 2 3

Snippet 4:

public class DoWhileLoop {

public static void main(String[] args) {

int i = 1;

do {

System.out.print(i + " "); //1 2 3 4

i++; //2 3 4 5

} while (i < 5);

System.out.println(i); //5

}

}

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);

}

}

Dry run:

int num = 1; // num = 1

1. i = 2

if (i % 2 == 0) { // i = 1, so 1 % 2 != 0, false

num += i; // Not executed

} else { // Condition is true

num -= i; // num = 1 - 1 = 0

}

1. i = 2

if (i % 2 == 0) { // i = 2, so 2 % 2 == 0, true

num += i; // num = 0 + 2 = 2

} else { // Not executed

num -= i;

}

1. i = 3

if (i % 2 == 0) { // i = 3, so 3 % 2 != 0, false

num += i; // Not executed

} else { // Condition is true

num -= i; // num = 2 - 3 = -1

}

1. i = 4

if (i % 2 == 0) { // i = 4, so 4 % 2 == 0, true

num += i; // num = -1 + 4 = 3

} else { // Not executed

num -= i;

}

System.out.println(num) // Prints 3

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); // Prints the result of y

}

}

Dry run:

y = ++x - x-- + --x + x++

y = 6 - 6 + 4 + 4 = 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);

}

}

Dry run:

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);

}

}

Dry run: