**Chapter 4**

**Exercise 1**

1. **Error: i) The semicolon after ‘while (i <= 10)’ ends the loop prematurely.**

**ii) There’s also an extra closing brace ‘}’ with no matching opening brace ‘{‘ .**

**Corrected:**

**i = 1;**

**while (I <= 10){**

**++1;**

**}**

1. **Error: The floating-point numbers like 0.1 cannot be relied on for exact comparisons (e.g., k != 1.0) due to precision errors.**

**Corrected: Use a counter with integers instead:**

**For (int 1 = I; 1 <= 10; i++){**

**System.out.println(i \* 0.1);**

**}**

1. **Error: The code is missing ‘break’ after case 1. Without it execution “falls through” to case 2.**

**Corrected:**

switch (n) {

case 1:

System.out.println("The number is 1");

break;

case 2:

System.out.println("The number is 2");

break;

default:

System.out.println("The number is not 1 or 2");

break;

}

1. n = 1;

while (n < 10) {

System.out.println(n)

n++;

}

Exercise 2

4.5) a) Control Variable Initialization: This is where the loop control variable is given an initial value before the loop starts. For example, setting i = 0.

b) Loop-Continuation Condition: This condition is evaluated before each iteration of the loop to determine whether the loop to determine whether the loop to determine whether the loop should continue. For example, I < 10.

c) Control Variable Update: The control variable is modified (usually incremented or decremented) at the end of each loop iteration. For example, i = i + 1 or i++.

d) Body of the Loop: This contains the set of statements that are executed each time the loop iterates, as long as the loop-continuation condition is true.

4.6) Similarities:

a) Both are looping constructs used to repeat a block of code multiple times.

b) Both require a loop-continuation condition to determine when to stop.

c) Both can technically be used interchangeably (you can rewrite a for loop as a while loop and vice versa

Differences:

|  |  |  |
| --- | --- | --- |
| Feature | While Loop | For Loop |
| Purpose | Used when the number of iterations is unknown or not fixed in advance. | Used when the number of iterations is known or counted. |
| Syntax | While  (condition) { … } | For (init; condition; update) { … } |
| Readability | Easier to read for indefinite loops. | Easier to read for counter-controlled loops. |
| Structure | Seperates the initialization, condition, and update. | Combines initialization, condition, and update in one line. |
| Initialization/Update | Must be done manually, usually before and within the loop. | Done within the loop header. |

4.7) Example Situation: Imagine you’re writing a simple menu-driven program where the user must see the menu and make a choice before deciding whether to continue.

The menu must display at least once before the program checks if the user wants to continue. A while loop checks the condition before running the loop body, which could skip the menu entirely if the condition starts out false. In this case, using do…while ensures the menu is always shown at least once, which makes for better user experience and logic flow.

4.8) Similarities:

a) Both are used to alter the normal flow of loops (for, while, or do…while).

b) Both can be used inside conditional statements to make decisions during loop execution.

c) Both help in making loops more efficient or controlling when certain parts should be skipped or exited.

Differences

|  |  |  |
| --- | --- | --- |
| Feature | break statement | continue statement |
| Function | Terminates the entire loop immediately. | Skips the current iteration and moves to the next one. |
| Effect | Control exits the loop completely. | Control goes back to the beginning of the loop. |
| Common Use | Used to exit a loop early (e.g, when a condition is met). |  |
| Location in Loop | Often placed inside if conditions | Also placed inside if conditions to skip logic. |

**4.9)a)**

**for(int I = 100; 1 >= 1; i++){**

**System.out.println(i);**

**}**

**b) switch(value % 2){**

**case 0:**

**System.out.println(“Even integer”);**

**Break;**

**case 1:**

**System.out.println(“Old integer”);**

**Break;**

**}**

**c) for(int i = 19; i >= 1; i -=2){**

**System.out.println(i);**

**}**

d)

int counter = 2;  
do {  
 System.out.println(counter);  
 counter += 2;  
}

while (counter <= 100);