|  |  |
| --- | --- |
|  | ***Patuakhali Science and Technology University*** |

Assignment on

***“***Deitel book exercise solve***”***

Course Code: CCE-122

Course Title: Object Oriented Programming

Level - I; Semester - II

|  |
| --- |
| **Submitted By**  **Name: M.D. Sakibul Islam Shovon**  **ID:** 2302056  **REG:** 11834  **Session:** 2023-2024  Faculty of Computer Science and Engineering |

|  |
| --- |
| **Submitted To**  **Prof. Dr. Md. Samsuzzaman**  Professor of Computer and Communication Engineering Department  Faculty of Computer Science and Engineering |

**Deitel book exercise solve**

**5.1 Fill in the blanks in each of the following statements:**

a) Typically, for statements are used for counter-controlled repetition and while statements for sentinel-controlled repetition.

b) The do-while statement tests the loop-continuation condition after executing the loop’s body; therefore, the body always executes at least once.

c) The switch statement selects among multiple actions based on the possible values of an integer variable or expression, or a String.

d) The continue statement, when executed in a repetition statement, skips the remaining statements in the loop body and proceeds with the next iteration of the loop.

e) The && (logical AND) operator can be used to ensure that two conditions are both true before choosing a certain path of execution.

f) If the loop-continuation condition in a for header is initially false, the program does not execute the for statement’s body.

g) Methods that perform common tasks and do not require objects are called static methods.

**5.2 State whether each of the following is true or false. If false, explain why:**

a) The default case is required in the switch selection statement. (False)

Explanation: The default case is optional in a switch statement. It executes only if no other case matches.

b) The break statement is required in the last case of a switch selection statement. (False)

Explanation: The break is not required in the last case (or default) because the switch exits naturally. However, omitting break in non-last cases causes fall-through (execution continues to the next case).

c) The expression ((x > y) && (a < b)) is true if either x > y is true or a < b is true. (False)

Explanation: The && (AND) operator requires both conditions to be true. For "either/or" logic, use || (OR).

d) An expression containing the || operator is true if either or both of its operands are true. (True)

e) The comma (,) formatting flag in a format specifier (e.g., %,20.2f) indicates that a value should be output with a thousands separator. (True)

* Explanation: The , flag formats numbers with thousands separators (e.g., 1,000.00).

f) To test for a range of values in a switch statement, use a hyphen (–) between the start and end values of the range in a case label. (False)

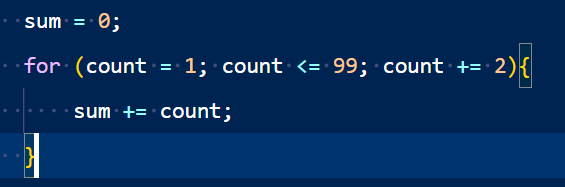
Explanation: Java does not support range syntax (e.g., case 1–3:). Instead, list values explicitly (e.g., case 1: case 2: case 3:).

g) Listing cases consecutively with no statements between them enables the cases to perform the same set of statements. (True)

**5.3 Write a Java statement or a set of Java statements to accomplish each of the following tasks:**

a) Sum the odd integers between 1 and 99, using a for statement. Assume that the integer variables sum and count have been declared.

Solution:



b) Calculate the value of 2.5 raised to the power of 3, using the pow method.

Solution:

A blue background with pink and white text

AI-generated content may be incorrect.

c) Print the integers from 1 to 20, using a while loop and the counter variable i. Assume that the variable i has been declared, but not initialized. Print only five integers per line. [Hint: Use the calculation i % 5. When the value of this expression is 0, print a newline character; otherwise, print a tab character. Assume that this code is an application. Use the System.out.println() method to output the newline character, and use the Sys tem.out.print('\t') method to output the tab character.]

Solution:

A computer screen shot of code

AI-generated content may be incorrect.

d) Repeat part (c), using a for statement.

Solution:

A screen shot of a computer code

AI-generated content may be incorrect.

**5.4 Find the error in each of the following code segments, and explain how to correct it:**

a) i = 1;

while (i <= 10);

++i;

}

Solution:

Error: The semicolon after the while header causes an infinite loop, and there’s a miss ing left brace. Correction: Replace the semicolon by a {, or remove both the ; and the }.

b) for (k = 0.1; k != 1.0; k += 0.1)

System.out.println(k);

Solution:

Error: Using a floating-point number to control a for statement may not work, because floating-point numbers are represented only approximately by most computers. Correction: Use an integer, and perform the proper calculation in order to get the values you desire:

for (k = 1; k != 10; k++)

System.out.println((double) k / 10);

c) switch (n)

{

case 1:

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

case 2:

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

break;

default:

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

}

Solution:

Error: The missing code is the break statement in the statements for the first case. Correction: Add a break statement at the end of the statements for the first case. This omission is not necessarily an error if you want the statement of case 2: to execute every time the case 1: statement executes.

d) The following code should print the values 1 to 10:

n = 1;

while (n < 10)

System.out.println(n++);

Solution:

Error: An improper relational operator is used in the while’s continuation condition. Correction: Use <= rather than.

**5.5 Describe the four basic elements of counter-controlled repetition.**

The four basic elements are:

1. Initialization – Setting the loop control variable to its starting value.
2. Condition – The loop continues executing as long as this condition is true.
3. Increment/Decrement – Modifying the loop control variable after each iteration.
4. Loop Body – The statements that execute repeatedly while the condition is true.

**5.6 Compare and contrast the while and for repetition statements.**

|  |  |  |
| --- | --- | --- |
| Feature | while Loop | for Loop |
| Initialization | Done before the loop | Done inside the loop header |
| Condition Check | Before each iteration | Before each iteration |
| Update | Inside the loop body | In the loop header |
| Best Use Case | When the number of iterations is unknown | When the number of iterations is known |

**5.7 If you need to execute the body of a loop at least once, would it be better to use a do…while statement or a while statement?**

Use do…while because it checks the condition after the first iteration, ensuring the loop body runs at least once.

A while loop checks the condition before entering, so it may not run at all.

**5.8 Compare and contrast the break and continue statements.**

|  |  |  |
| --- | --- | --- |
| Feature | break | continue |
| Effect | Exits the entire loop immediately | Skips the current iteration and proceeds to the next |
| Usage | Used in loops and switch statements | Only used in loops |
| Example | while (true) { if (x == 5) break; } | for (int i=0; i<10; i++) { if (i%2==0) continue; } |

**5.9 Find and correct the error(s) in each of the following segments of code:**

**a) while (i = 1; i <= 10, i+)**

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

**Solve:**

**Errors:**

* while should not have semicolons inside parentheses (it's not a for loop).
* i+ is invalid syntax (should be i++).
* i is not declared.

**Corrected:**

int i = 1;

while (i <= 10) {

System.out.println(i);

i++;

}

**b) The following code should print whether an integer value is negative or zero: switch (value)**

**{**

**Case value < 0:**

**System.out.println("Negative");**

**case 0:**

**System.out.println("Zero");**

**}**

**Errors:**

* Case should be lowercase (case).
* case conditions must be constant values (not expressions like value < 0).

Corrected (using if-else instead):

if (value < 0) {

System.out.println("Negative");

} else if (value == 0) {

System.out.println("Zero");

}

**c) The following code should output the odd integers from 19 to 1:**

**for (int i = 19; i > 1; i =+ 1)**

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

**Errors:**

* i =+ 1 is wrong (should be i -= 2 to decrement by 2 for odd numbers).
* The loop stops at i > 1, missing i = 1.

**Corrected:**

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

System.out.println(i);

**d) The following code should output the even integers from 1 to 50:**

**counter = 0;**

**do**

**{**

**System.out.println(counter + 1);**

**counter += 2;**

**} while (counter <= 51);**

**Errors:**

* counter + 1 prints odd numbers (should be counter).
* Loop runs until 51, which is unnecessary (should be counter < 50).

Corrected:

int counter = 0;

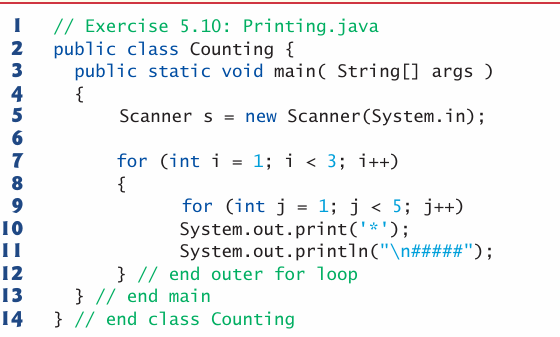
do {

System.out.println(counter);

counter += 2;

} while (counter <= 50);

**5.10 What does the following program do?**



**Solution:**

Uses nested for loops to print a pattern of stars (\*) and hash symbols (#####). Here's a breakdown of its behavior:

1. Outer Loop (i loop)

* Runs 2 times (i = 1 to i < 3).
* For each iteration of i, the inner loop runs.

2. Inner Loop (j loop)

* Runs 4 times (j = 1 to j < 5).
* Each iteration prints a single \* (without a newline due to System.out.print).
* After the inner loop completes, a newline (\n) is printed, followed by #####.

3. Output Structure

For each iteration of the outer loop (i), the program prints:

1. Four stars (\*\*\*\*) (from the inner loop).
2. A newline followed by #####.

**Output:**

\*\*\*\*

#####

\*\*\*\*

#####