**CHAPTER 4**

**ANSWERS TO EXERCISE 1**

1. The error is the same as in C++: a misplaced semicolon after the while condition. The trailing } is also problematic because it doesn't have a matching opening brace.

**int i = 1; // Declare and initialize i**

**while (i <= 10) { // Semicolon removed, curly braces added**

**++i;**

**}**

**How to correct it**

1. **int i = 1;**: The variable i is now properly declared and initialized. In Java, you must declare the type of a variable.
2. **Semicolon removed:** The semicolon after the while condition is removed.
3. **Curly braces added:** Curly braces {} are added to define the loop's body, ensuring that ++i; is executed in each iteration.
4. Floating-point numbers (like float and double in Java) are represented in binary format. Many decimal fractions cannot be represented exactly in binary. This leads to rounding errors during calculations.In this specific case, repeatedly adding 0.1 to k might not result in an exact 1.0. Due to the accumulation of tiny rounding errors, k might become something like 0.999999999 or 1.000000001 instead of exactly 1.0. As a result, the condition k != 1.0 might never become false, causing the loop to run indefinitely (an infinite loop).

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

**double k = i \* 0.1;**

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

**How to correct it**

1. **Using an integer counter: The most reliable way is to use an integer counter and scale it to the desired range (i.e This completely avoids floating-point comparisons and is the cleanest and most accurate solution in this specific scenario.)**
2. The error in this Java switch statement is the missing break statement after the first case.

switch (n) {

case 1:

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

break; // Added break here

case 2:

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

break;

default:

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

break;

}

**How to correct it**

1. **To fix this, you need to add a break statement after each case that you want to be treated as a separate, independent block of code:**
2. The issue is that n++ (post-increment) increments n *after* its current value is used. Because the loop condition is n < 10, the loop stops when n is 9, and then n is incremented to 10 *after* the last print statement, so 10 is never printed.

Here are two ways to correct the code:

1. **Pre-increment:** Use ++n (pre-increment) instead of n++ (post-increment):

int n = 1; // It's good practice to declare the type of n

while (n < 11) { // Or while (n <= 10)

System.out.println(++n); // n is incremented BEFORE being printed

}

1. **Increment after printing:** Move the increment to a separate statement *after* the println:

int n = 1;

while (n <= 10) { // Condition changed to include 10

System.out.println(n);

n++; // Increment AFTER printing

}

**ANSWERS TO EXERCISE 2**

4.5) Describe the four basic elements of counter-controlled repetition.

1. **Initialization:** This step initializes the *control variable* (also known as the loop counter). The control variable is what keeps track of the number of iterations. Initialization typically involves declaring the control variable and assigning it an initial value.

* **Example (for loop):** for (int i = 0; ...) initializes i to 0.
* **Example (while loop):** int i = 0; while (... initializes i to 0.

1. **Loop-Continuation Condition:** This is a boolean expression that determines whether the loop should continue executing. The loop continues as long as this condition is true. When the condition becomes false, the loop terminates.

* **Example (for loop):** for (int i = 0; i < 10; ...) the condition is i < 10. The loop continues as long as i is less than 10.
* **Example (while loop):** while (i < 10) ... the condition is i < 10.

1. **Increment/Decrement:** This step updates the control variable after each iteration of the loop. This update usually involves incrementing or decrementing the control variable by a specific amount. This is crucial for eventually making the loop-continuation condition false and preventing an infinite loop.

* **Example (for loop):** for (int i = 0; i < 10; i++) the increment is i++ (post-increment).
* **Example (while loop):** i++; (usually placed at the end of the loop body).

1. **Loop Body:** This is the block of code that is executed repeatedly during each iteration of the loop.

4.6) Compare and contrast the while and for repetition statements.

|  |  |  |
| --- | --- | --- |
| **Feature** | **while loop** | **for loop** |
| **Structure** | while (condition) { // loop body } | for (initialization; condition; increment/decrement) { // loop body } |
| **Control Variables** | Initialization, increment/decrement of the loop control variable are handled *separately* from the loop condition. | Initialization, condition, and increment/decrement of the loop control variable are all handled *within the for loop header*. |
| **Typical Use Cases** | Used when the number of iterations is *not known in advance* or depends on a condition that might change during the loop's execution. Examples: reading input until a specific value is entered, processing data until a certain condition is met. | Used when the number of iterations is *known in advance* or when you need to iterate over a range of values. Examples: iterating through an array, performing a task a specific number of times. |
| **Readability (for counting)** | Can be less readable for simple counting loops because the initialization, condition, and increment are separated. . | Generally more readable for simple counting loops because all loop control elements are in one place |
| **Example** | java<br>int i = 0;<br>while (i < 10) {<br> System.out.println(i);<br> i++;<br>} | java<br>for (int i = 0; i < 10; i++) {<br> System.out.println(i);<br>} |
| **Infinite Loops** | Easier to accidentally create an infinite loop if the increment/decrement is forgotten or the condition is never met. | Less prone to accidental infinite loops (but still possible if the condition is always true or the increment/decrement is incorrect). |

4.7) Discuss a situation in which it would be more appropriate to use a do…while statement…

Answer

A do...while loop is more appropriate than a while loop when you need to guarantee that the loop body executes *at least once*, regardless of the initial condition.

4.8) Compare and contrast the break and continue statements.

Answer

COMPARE

Both break and continue are control flow statements in Java (and many other programming languages) used to alter the normal execution flow of loops (for, while, do-while) and switch statements, but they do so in different ways.

CONTRASTS

**break Statement:**

* **Purpose:** Terminates the *entire* loop or switch statement immediately. Execution continues with the statement immediately following the loop or switch.
* **Usage in Loops:** When break is encountered inside a loop, the loop is exited, regardless of whether the loop condition is still true.
* **Usage in switch:** break is essential in switch statements to prevent "fallthrough" (executing the code of subsequent case labels).

**Continue Statement:**

* **Purpose:** Skips the *rest* of the *current* iteration of the loop and proceeds to the *next* iteration. The loop condition is still checked to determine if further iterations should occur.
* **Usage in Loops:** When continue is encountered inside a loop, the remaining statements in the current iteration are skipped, and the loop proceeds to the next iteration (checking the loop condition).
* **Not Applicable to switch:** continue cannot be used within a switch statement.

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

* + 1. For (i = 100, i >= 1, i++) System.out.println(i);

**ANSWER**

**Error:**

**Case-sensitivity:** In Java, keywords are case-sensitive. For should be written as for.

**Corrected Code:**

* 1. for (int i = 100; i >= 1; i--) System.out.println(i);
     1. The following code should print whether integer value is odd or even:

switch (value % 2) {

case 0:

System.out.println("Even integer");

case 1:

System.out.println("Odd integer");

}

**ANSWER**

**Errors:**

* **Missing break statements:** In a switch case, if a break statement is not encountered, the program continues to execute the subsequent cases. This leads to incorrect output in this case.

**Corrected Code:**

* + 1. **switch (value % 2) {**

**case 0:**

**System.out.println("Even integer");**

**break;**

**case 1:**

**System.out.println("Odd integer");**

**break;**

**default:**

**System.out.println("Invalid input");**

**}**

* + 1. The following code should output the odd integers from 19 to 1:

for (i = 19; i >= 1; i += 2)

System.out.println(i);

**ANSWER**

**ERROR:**

* **Missing variable declaration:** The loop variable i is used without being declared.

**CORRECTED CODE:**

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

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

* + 1. The following code should output the even integers from 2 to 100:

counter = 2;

do {

System.out.println(counter);

counter += 2;

} While (counter < 100);

**ANSWER**

**ERRORS:**

* **Missing variable declaration:** The variable counter is used without being declared.
* **Case-sensitivity:** The keyword While should be written as while.

**CORRECTED CODE:**

* 1. **int counter = 2;**

**do {**

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

**counter += 2;**

**} while (counter <= 100);**

4.10) What does the following program do?

Exercise 4.10: Printing.java

public class Printing {

public static void main(String[] args) {

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

for (int j = 1; j <= 5; j++)

System.out.print('@');

System.out.println();

}

}

}

**ANSWER**

This code will print a rectangle shape on the console consisting of 10 rows and 5 columns, each filled with the @ symbol**.**

4.11) (Find the Smallest Value) Write an application that finds the smallest of several integers. Assume that the first value read specifies the number of values to input from the user.

**ANSWER**

import java.util.Scanner;

public class FindSmallest {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the number of integers: ");

int numValues = input.nextInt();

if (numValues <= 0) {

System.out.println("Invalid input. Number of integers must be greater than 0.");

return;

}

System.out.print("Enter " + numValues + " integers: ");

int smallest = Integer.MAX\_VALUE; // Initialize with maximum possible integer value

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

int number = input.nextInt();

if (number < smallest) {

smallest = number;

}

}

System.out.println("The smallest integer is: " + smallest);

}

}

4.12) (Calculating the Product of Odd Integers)

Write an application that calculates the product of the odd integers from 1 to 15.

**ANSWER**

public class OddIntegerProduct {

public static void main(String[] args) {

int product = 1;

for (int i = 1; i <= 15; i += 2) {

product \*= i;

}

System.out.println("The product of odd integers from 1 to 15 is: " + product);

}

}

4.13) (Factorials) Factorials are used frequently in probability problems. The factorial of a positive integer n (written n! and pronounced “n factorial”) is equal to the product of the positive integers from 1 to n. Write an application that calculates the factorials of 1 through 20. Use type long. Display the results in tabular format. What difficulty might prevent you from calculating the factorial of 100?

**ANSWER**

public class Factorial {

public static void main(String[] args) {

System.out.println("n\t\tn!");

System.out.println("------------------");

for (int n = 1; n <= 20; n++) {

long factorial = calculateFactorial(n);

System.out.println(n + "\t\t" + factorial);

}

}

public static long calculateFactorial(int n) {

long factorial = 1;

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

factorial \*= i;

}

return factorial;

}

}

4.14 (Modified Compound-Interest Program) Modify the compound-interest application to repeat its steps for interest rates of 5%, 6%, 7%, 8%, 9% and 10%. Use a for loop to vary the interest rate.

**ANSWER**

public class CompoundInterestModified {

public static void main(String[] args) {

final double PRINCIPAL = 1000.0; // Principal amount

final int YEARS = 10; // Number of years

System.out.printf("%s%20s\n", "Year", "Amount on deposit");

// Loop through interest rates from 5% to 10%

for (double rate = 0.05; rate <= 0.10; rate += 0.01) {

double amount = PRINCIPAL \* Math.pow(1 + rate, YEARS);

System.out.printf("%4d%,20.2f\n", YEARS, amount);

}

}

}

**Making a Difference**

**4.30 (Global Warming Facts Quiz)**

**The controversial issue of global warming has been widely publicized by the film “An Inconvenient Truth,” featuring former Vice President Al Gore. Mr. Gore and a U.N. network of scientists, the Intergovernmental Panel on Climate Change, shared the 2007 Nobel Peace Prize in recognition of “their efforts to build up and disseminate greater knowledge about man-made climate change.” Research both sides of the global warming issue online (you might want to search for phrases like “global warming skeptics”). Create a five-question multiplechoice quiz on global warming, each question having four possible answers (numbered 1–4). Be objective and try to fairly represent both sides of the issue. Next, write an application that administers the quiz, calculates the number of correct answers (zero through five) and returns a message to the user. If the user correctly answers five questions, print “Excellent”; if four, print “Very good”; if three or fewer, print “Time to brush up on your knowledge of global warming,” and include a list of some of the websites where you found your facts.**

**ANSWER**

import java.util.Scanner;

public class GlobalWarmingQuiz {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int correctAnswers = 0;

// Quiz questions

String[][] questions = {

{"1. Is the Earth's average temperature increasing?",

"1. Yes", "2. No", "3. No significant change", "4. Data is inconclusive"},

{"2. What is the primary cause of global warming, according to the IPCC?",

"1. Volcanic eruptions", "2. Solar radiation", "3. Human activities", "4. Natural cycles"},

{"3. Do some scientists dispute the link between human activities and global warming?",

"1. Yes", "2. No", "3. Only a few", "4. All agree"},

{"4. What is the main greenhouse gas emitted by human activities?",

"1. Water vapor", "2. Carbon dioxide", "3. Methane", "4. Nitrous oxide"},

{"5. What are the potential consequences of global warming?",

"1. Rising sea levels", "2. More extreme weather events", "3. Changes in ecosystems", "4. All of the above"}

};

// Correct answers (index of the correct answer in each question)

int[] correctAnswersIndex = {0, 2, 0, 1, 3};

// Administer the quiz

for (int i = 0; i < questions.length; i++) {

System.out.println("\nQuestion " + (i + 1) + ": " + questions[i][0]);

System.out.println("1. " + questions[i][1]);

System.out.println("2. " + questions[i][2]);

System.out.println("3. " + questions[i][3]);

System.out.println("4. " + questions[i][4]);

System.out.print("Enter your answer (1-4): ");

int userAnswer = input.nextInt();

if (userAnswer == correctAnswersIndex[i]) {

correctAnswers++;

}

}

// Evaluate results

if (correctAnswers == 5) {

System.out.println("\nExcellent!");

} else if (correctAnswers == 4) {

System.out.println("\nVery good!");

} else {

System.out.println("\nTime to brush up on your knowledge of global warming.");

System.out.println("\nHere are some resources for further learning:");

System.out.println("- IPCC (Intergovernmental Panel on Climate Change): https://www.ipcc.ch/");

System.out.println("- NASA Climate Change: https://science.nasa.gov/climate-change/");

System.out.println("- NOAA Climate.gov: https://www.noaa.gov/climate");

}

}

}

**4.31 (Tax Plan Alternatives; The “FairTax”)**

**There are many proposals to make taxation fairer. Check out the FairTax initiative in the United States at www.fairtax.org. Research how the proposed FairTax works. One suggestion is to eliminate income taxes and most other taxes in favor of a 23% consumption tax on all products and services that you buy. Some FairTax opponents question the 23% figure and say that because of the way the tax is calculated, it would be more accurate to say the rate is 30%—check this carefully. Write a program that prompts the user to enter expenses in various expense categories they have (e.g., housing, food, clothing, transportation, education, health care, vacations), then prints the estimated FairTax that person would pay.**

**ANSWER**

import java.util.HashMap;

import java.util.Scanner;

public class FairTaxCalculator {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

double fairTaxRate = 0.23; // Assuming a 23% FairTax rate (adjust if needed)

// Expense categories (modify or add as needed)

HashMap<String, String> expenseCategories = new HashMap<>();

expenseCategories.put("housing", "Housing");

expenseCategories.put("food", "Food");

expenseCategories.put("clothing", "Clothing");

expenseCategories.put("transportation", "Transportation");

expenseCategories.put("education", "Education");

expenseCategories.put("healthcare", "Healthcare");

expenseCategories.put("vacations", "Vacations");

// Get user input for expenses

double totalExpenses = 0.0;

for (String category : expenseCategories.keySet()) {

System.out.printf("Enter your monthly %s expenses: $", expenseCategories.get(category));

double expense = input.nextDouble();

totalExpenses += expense;

}

// Calculate estimated annual FairTax

double annualFairTax = totalExpenses \* 12 \* fairTaxRate;

// Print results

System.out.printf("\nYour estimated annual FairTax would be: $%.2f\n", annualFairTax);

System.out.println("\n\*\*Disclaimer:\*\*");

System.out.println("- This is a simplified calculator and may not reflect the exact FairTax you would pay.");

System.out.println("- The actual FairTax may vary depending on specific circumstances and potential exemptions.");

System.out.println("- It's recommended to consult with a tax professional for accurate calculations.");

}

}

**4.32 (Facebook User Base Growth)**

**According to CNNMoney.com, Facebook hit one billion users in October 2012. Using the compound-growth technique you learned in Fig. 4.6 and assuming its user base grows at a rate of 4% per month, how many months will it take for Facebook to grow its user base to 1.5 billion users? How many months will it take for Facebook to grow its user base to two billion users?**

**ANSWER**

public class FacebookGrowth {

public static void main(String[] args) {

final double INITIAL\_USERS = 1.0e9; // 1 billion users (as a double)

final double TARGET\_USERS\_1\_5 = 1.5e9; // 1.5 billion users

final double TARGET\_USERS\_2 = 2.0e9; // 2 billion users

final double MONTHLY\_GROWTH\_RATE = 0.04; // 4% monthly growth rate

int monthsToReach1\_5 = calculateMonthsToTarget(INITIAL\_USERS, TARGET\_USERS\_1\_5, MONTHLY\_GROWTH\_RATE);

int monthsToReach2 = calculateMonthsToTarget(INITIAL\_USERS, TARGET\_USERS\_2, MONTHLY\_GROWTH\_RATE);

System.out.println("Months to reach 1.5 billion users: " + monthsToReach1\_5);

System.out.println("Months to reach 2 billion users: " + monthsToReach2);

}

public static int calculateMonthsToTarget(double initialUsers, double targetUsers, double growthRate) {

int months = 0;

double currentUsers = initialUsers;

while (currentUsers < targetUsers) {

currentUsers \*= (1 + growthRate);

months++;

}

return months;

}

}