**Chapter 4: Conditionals and Loops**

**Multiple Choice Questions**:

1) Which of the following statements best describes the flow of control in the main method of a Java program that has no conditionals or loops?

a) Program statements are all executed at the same time.

b) Program statements are executed according to their priority, which is specified by the programmer.

c) Program statements are executed linearly, with earlier statements being executed first.

d) Program statements are executed linearly, with later statements being executed first.

e) Some program statements are executed at the same time, and others are executed in a linear manner.

Answer: c

Explanation: Program statements in a Java program are executed linearly when there are no conditionals or loops. This means that statements that appear earlier in the code are executed before statements that appear later in the code.

2) Which of the following best describes this code snippet?

if (count != 400)

System.out.println("Hello World!");

a) If the variable count is exactly equal to 400, "Hello World" will be printed.

b) If the variable count is not equal to 400, "Hello World" will be printed.

c) If the variable count is close to, but not greater than, 400, "Hello World" will be printed.

d) If the variable count is exactly equal to 399 or 401, "Hello World" will be printed.

e) This code will not compile.

Answer: b

Explanation: The != operator means *is not equal to*. Therefore if the variable count is not equal to 400, the following line will be executed. The boolean operator to test *is equal to* is ==. There are no boolean operators that directly test the cases specified in choices c and d.

3) In Java, a *block statement* is

1. a set of statements that are all indented to the same column.
2. a set of statements enclosed in { and }.
3. statements that form a rectangular set of characters on the screen.
4. a statement that prevents other statements from executing.
5. a statement that ends the execution of the program.

Answer: b

Explanation: Java uses block statements to execute a set of statements as if it was a single statement. Block statements are commonly used in the body of loop statements and in the execution of if and if-else statements. Java does not pay attention to whitespace, indentation, or other formatting, so choices a) and c) are false.

4) Let a and b be valid boolean expressions. Which of the following best describes the result of the expression a || b?

a) It will evaluate to true if a evaluates to true and b evaluates to true. It will evaluate to false otherwise.

b) It will evaluate to false if a evaluates to false and b evaluates to false. It will evaluate to true otherwise.

c) It will evaluate to true if a evaluates to false and b evaluates to false. It will evaluate to true otherwise.

d) It will evaluate to true if a evaluates to false or b evaluates to false. It will evaluate to true otherwise.

e) None of the above statements correctly describes the evaluation of the expression.

Answer: b

Explanation: The || operator represents the logical or. Therefore the expression will evaluate to true whenever a is true or b is true, or if they are both true. Thus, it will evaluate to false only when both a is false and b is false, and it will evaluate to true otherwise.

5) Which of the following expressions best represents the condition "if the grade is between 75 and 100"?

1. if (75 < grade && grade < 100)
2. if (grade != 75 && grade != 100)
3. if (75 < grade < 100)
4. if (75 > grade || grade < 100)
5. if (75 < grade || grade < 100)

Answer: a

Explanation: Choice a best represents the condition specified. Choice b best represents "if the grade is not 75 and the grade is not 100." Choice c is not valid Java code. Choice d represents "if the grade is less than 75 or the grade is greater than 100." Choice e represents "if the grade is greater than 75 or the grade is less than 100," and it will always evaluate to true.

6) A set of statements must be executed an unknown number of times, and possibly not executed at all. Which loop statement should not be used to control the execution of this set of statements?

1. while
2. for
3. do
4. repeat
5. Any of these loop statements can be used.

Answer: c

Explanation: Because the condition that controls the execution of a do loop comes after the body, the body of a do loop always executes at least once. It is not a choice to use if the statements may not need to be executed at all. There is no repeat loop in Java.

7) Suppose we wanted to process a text file called "input.txt" using the Scanner object. Which of the following lines of code correctly creates the necessary Scanner object?

1. Scanner inputFile = new Scanner("input.txt");
2. Scanner inputFile = new Scanner(new InputFile("input.txt");
3. Scanner inputFile = new Scanner(new File(input.txt);
4. Scanner inputFile = new Scanner(new InputFile(input.txt);
5. Scanner inputFile = new Scanner(new File("input.txt");

Answer: e

Explanation: The Scanner constructor needs to take a reference to a File object as a parameter, so choice a, b and d are incorrect. The constructor for a File object takes in a String, so choice c is incorrect. Therefore, choice e is correct.

8) The code below is supposed to add the numbers from 1 up to and including 10. It does not calculate the correct sum. The problem is caused by a(n) \_\_\_\_\_\_\_\_ error.

int sum = 0;

for (int count = 1; count < 10; count++)

sum += count;

1. syntax
2. compilation
3. requirement
4. off-by-one
5. testing

Answer: d

Explanation: An off-by-one error occurs when a loop body is executed either one too many times or one too few times. The problem is solved if the boolean condition is replaced by count <= 10.

9) What happens if a case in a switch statement does not end with a break statement?

a) The program will not compile.

b) The switch statement will never execute.

c) It will cause an infinite loop.

d) The switch statement will execute the next case statement as well.

e) The case will never be executed.

Answer: d

Explanation: If the case statement does not end in a break statement, the next case will also be executed. Therefore choice d is correct.

10) A switch statement does not have a case that matches the value of the expression, and it does not have a default case. What happens?

a) The first case will be executed because there is no default case.

b) The last case will be executed because there is no default case.

c) The case whose value is closest to the expression will be executed.

d) None of the cases will be executed.

e) An execution error will occur if there is no matching case and no default case.

Answer: d

Explanation: If none of the cases are matched, then none of the cases will be executed.

11) Suppose we want to write an if statement to test whether two String objects, referenced by stringOne and stringTwo, are the same. Which of the following is the correct way to achieve this?

1. if(stringOne == stringTwo)
2. if(stringOne.compareTo(stringTwo))
3. if(stringOne.equals(stringTwo))
4. if(stringOne != stringTwo)
5. if(stringOne === stringTwo)

Answer: c

Explanation: In Java, a programmer should not use conditional operators to compare objects for equality. Instead, the equals() method should be used. Therefore choice c is correct, and choices a and d are incorrect. The compareTo() method returns an int and not a boolean, so choice b will not compile. There is no operator in Java that is represented by ===, so choice e is incorrect.

12) A(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is an object that has methods that allow you to process a collection of items one at a time.

a) iterator

b) loop

c) conditional

d) palindrome

e) nested loop

Answer: a

Explanation: An iterator is an object that has methods that allow you to process a collection of items one at a time. None of the other choices are objects in Java.

13) A logical expression can be described by a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ that lists all possible combinations of values for the variables involved in an expression.

a) palindrome

b) nested loop

c) equality operator

d) switch statement

e) truth table

Answer: e

Explanation: A truth table lists out all possible combinations of values for the variables involved in an input expression and also lists the evaluation of a logical expression.

14) The expression that is evaluated at the start of a switch statement must not be which primitive data type below?

1. byte
2. short
3. int
4. char
5. double

Answer: e

Explanation: The possible primitive data types for the expression of a switch statement are byte, short, int, and char.

15) Which of the following for loop headers will cause the body of the loop to be executed 100 times?

1. for(int i = 0; i <= 100; i++)
2. for(int i = 1; i < 100; i++)
3. for(int i = 1; i <= 101; i++)
4. for(int i = 0; i < 100; i++)
5. none of these for loops will execute the loop body 100 times

Answer: d

Explanation: For choice a, the variable i has the values 0, 1, 2, …, 99, and 100, which are 101 values. For choice b, the values of i are 1, 2, 3, …, 97, and 98, which are 98 values. For choice c, the values of i are 1, 2, 3, …, 100, and 101, which are 101 values. **True/False Questions**:

1) Every if statement requires an associated else statement, but not every else statement requires an associated if statement.

Answer: False

Explanation: Every else statement requires an associated if statement, but not every if statement requires an associated else statement.

2) In a nested if statement an else clause is matched to the closest unmatched if.

Answer: True

Explanation: Any ambiguity among if-else statements is resolved using this rule in Java.

3) In Java, a boolean expression is limited to having exactly 2 logical operators.

Answer: False

Explanation: A boolean expression can have any number of logical operators, so it is possible to construct sophisticated conditions using a single boolean expression.

4) A do statement should be used to avoid creating an infinite loop.

Answer: False

Explanation: It is possible to create an infinite loop with any of the loop statements. Proper coding and attention to the condition that controls loop execution are essential to avoiding infinite loops.

5) A while statement always executes its loop body at least once.

Answer: False

Explanation: A do statement always executes its loop body at least once. A while statement will not execute its loop body if its condition evaluates to false on the first pass.

6) The initialization portion of a for loop header can be used to declare a variable that is used during loop execution.

Answer: True

Explanation: If a variable is only needed during the execution of the body of a for loop, it can be declared and initialized in the for loop header.

7) The relational operators should not be used to test the equality of objects.

Answer: True

Explanation: To test the equality of objects, the equals()method should be used. The compareTo() method may also be used to test for equality when it returns 0.

8) It is possible to implement a switch statement using if statements.

Answer: True

Explanation: A programmer can implement any switch statement using a series of nested if statements. The code for a switch statement may be clearer and more readable, however.

9) An infinite loop is a compile-time error.

Answer: False

Explanation: An infinite loop is usually caused by a logical error, and will not be caught by the compiler.

10) The Scanner object can be used to read text files.

Answer: True

Explanation: The Scanner object can be used to read text files by passing a File object to its constructor.**Short Answer Questions**:

1) Using the following declarations and initializations:

int num1 = 5,

num2 = 12,

num3 = 13;

write a boolean expression that is true when the sum of num1 and num2 is more than num3, or when the value of num1 is an odd number.

Answer: ( (num1 + num2 > num3) || (num1 % 2 == 1) )

2) Write a snippet of code that determines which of two integer variables, intOne and intTwo, contains a larger number, and print out the larger one. If they are equal, the output should say that.

Answer:

if(intOne > intTwo)

System.out.println(intOne + " is larger than " + intTwo);

else if(intOne < intTwo)

System.out.println(intTwo + " is larger than " + intOne);

else

System.out.println(intOne + " and " + intTwo + " are equal!");

3) name is a String object that contains user input. Write a segment of code that determines if name contains "George". If it does, print the message "Hey, that's my name too! "

Answer:

if(name.equals("George"))

System.out.println("Hey, that's my name too!");

4) What is output by the following code fragment?

int num = 0;

int max = 10;

while(num < max) {

System.out.print(num + " ");

num += 2;

}

Answer:

This code will output the following:

0 2 4 6 8

5) Rewrite the following code fragment using a for loop instead of a while loop.

int i = 0;

while(i < 50) {

System.out.println(i);

i+=2;

}

Answer:

for(int i = 0; i < 50; i+=2)

System.out.println(i);

6) Write a short application that takes in a String from the user and prints it out backwards.

Answer:

import java.util.Scanner;

public class StringReverse {

public static void main(String [] args) {

String inputString;

Scanner input = new Scanner(System.in);

System.out.print("Please enter a string: ");

inputString = input.nextLine();

for(int i = inputString.length – 1; i >= 0; i--)

System.out.print(inputString.charAt(i));

}//end main

}//end class

7) The following code compiled, but while running it the program appears to hang (e.g. nothing happens). This is a sign that there may be an infinite loop. What part of this code fragment may be causing an infinite loop?

while(i < 50); {

System.out.println(i);

i+=2;

}

Answer: There is a misplaced semi-colon after the while statement. This causes the loop to have an empty body, meaning that the i variable is never updated. This leads to an infinite loop.

8) grade is a char variable that holds a letter grade 'A', 'B', 'C', 'D', or 'F'. Write a switch statement that prints one of the messages in the table below based on the value in grade.

|  |  |
| --- | --- |
| **Grade** | **Message** |
| A | Excellent! |
| B | Very Good |
| C | Good |
| D | You can do better |
| F | You must do better |
| any other value | Error – invalid grade detected |

Answer:

switch (grade){

case 'A':

System.out.println("Excellent!");

break;

case 'B':

System.out.println("Very Good");

break;

case 'C':

System.out.println("Good");

break;

case 'D':

System.out.println("You can do better");

break;

case 'F':

System.out.println("You must do better");

break;

default:

System.out.println("Error – invalid grade detected");

} // end switch

9) How many times will the body of a while loop be executed if the boolean expression is false the first time that the while statement is encountered?

Answer: The body will not be executed at all. The expression of a while loop is evaluated before the body is executed, so a false expression means that the body will not be executed.

10) Write a short code fragment that uses a while loop to verify that the user enters a positive integer as input. You may assume that a Scanner object named input has already been created.

Answer:

System.out.print("Please enter a positive value: ");

int value = input.nextInt();

while(value <= 0) {

System.out.println("Error (the number was not positive)");

System.out.print("Please enter a positive value: ");

value = input.nextInt();

}

11) Write a do loop that verifies that the user enters an odd value. You may assume that a Scanner object named input has already been created.

Answer:

int value;

do {

System.out.print("Please enter an odd value: ");

value = input.nextInt();

if(value%2 != 1)

System.out.println("Error (the number was not odd)");

} while(value%2 != 1);

12) Write a switch statement that switches on an integer variable named val. If val is 2 or 15, then output "Hello World." For all other values, output "Goodbye World."

Answer:

switch(val) {

case 2:

System.out.println("Hello World!");

break;

case 15:

System.out.println("Hello World!");

break;

default:

System.out.println("Goodbye World!");

}

13) Write a code fragment that allows a user to continue inputting numbers until she enters a sentinel value of 0. Then print the sum of all the numbers she entered. You may assume that a Scanner object named input has already been created.

Answer:

int sum = 0;

int inputInt;

do {

System.out.print("Please enter an integer (0 to quit): ");

inputInt = input.nextInt();

sum += inputInt;

}while(inputInt != 0);

System.out.println("The sum of the numbers is " + sum);

14) Write a code fragment that determines how many times the character 'A' appears in a String object called name.

Answer:

int countA = 0;

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

if(name.charAt(a) == 'A')

countA++;

System.out.println("A appears " + countA + " times in " + name + ".");

15) line is a String object that holds an unknown number of int values separated by spaces. Write a segment of code that will compute and display the sum of the values in line.

Answer:

int sum = 0;

int val; // the next value from line

Scanner s = new Scanner(line);

while (s.hasNext()) {

val = s.nextInt();

sum += val;

}

System.out.println("The sum is " + sum);