Self-Review Exercises

2.1 Fill in the blanks:

a) A(n) left brace ({) begins the body of every method, and a(n) right brace (}) ends the body of every method.

b) You can use the if statement to make decisions.

c) // begins an end-of-line comment.

d) Spaces, tabs, and newline characters are called white space.

e) Keywords are reserved for use by Java.

f) Java applications begin execution at method main.

g) Methods System.out.print, System.out.println, and System.out.printf display information in a command window.

---

2.2 True or False:

a) False – Comments are ignored by the compiler and do not affect program execution.

b) True – All variables must have a data type when declared in Java.

c) False – Java is case-sensitive, so number and NuMbEr are different variables.

d) False – The remainder operator (%) can be used with both integer and floating-point operands.

e) False – Multiplication (\*), division (/), and modulus (%) have higher precedence than addition (+) and subtraction (-).

---

2.3 Write statements:

a) int c, thisIsAVariable, q76354, number;

b) System.out.print("Enter an integer: ");

c) int value = input.nextInt();

d) System.out.println("This is a Java program");

e) System.out.printf("This is a %s%n program", "Java");

f) if (number != 7) System.out.println("The variable number is not equal to 7");

---

2.4 Identify and correct errors:

a) Incorrect: if (c < 7); System.out.println("c is less than 7");

Corrected: if (c < 7) System.out.println("c is less than 7"); (remove the semicolon after if)

b) Incorrect: if (c => 7) System.out.println("c is equal to or greater than 7");

Corrected: if (c >= 7) System.out.println("c is equal to or greater than 7"); (use >= instead of =>)

---

2.5 Write declarations, statements, or comments:

a) // This program calculates the product of three integers

b) Scanner input = new Scanner(System.in);

c) int x, y, z, result;

d) System.out.print("Enter first integer: ");

e) x = input.nextInt();

f) System.out.print("Enter second integer: ");

g) y = input.nextInt();

h) System.out.print("Enter third integer: ");

i) z = input.nextInt();

j) result = x \* y \* z;

k) System.out.printf("Product is %d%n", result);

---

2.6 Complete program to calculate product of three integers:

import java.util.Scanner;

public class ProductCalculator {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter first integer: ");

int x = input.nextInt();

System.out.print("Enter second integer: ");

int y = input.nextInt();

System.out.print("Enter third integer: ");

int z = input.nextInt();

int result = x \* y \* z;

System.out.printf("Product is %d%n", result);

}

}

---

2.7 Fill in the blanks:

a) Comments are used to document a program and improve its readability.

b) A decision can be made in a Java program with a(n) if statement.

c) Calculations are normally performed by assignment statements.

d) The arithmetic operators with the same precedence as multiplication are division (/) and modulus (%).

e) When parentheses in an arithmetic expression are nested, the innermost set of parentheses is evaluated first.

f) A location in the computer’s memory that may contain different values at various times throughout the execution of a program is called a(n) variable.

---

2.8 Write Java statements:

a) System.out.print("Enter an integer: ");

b) a = b \* c;

c) // This program performs a sample payroll calculation

---

2.9 True or False:

a) False – Operator precedence determines evaluation order, not just left-to-right order.

b) True – All listed variable names are valid.

c) False – Operators are evaluated based on precedence, not strictly from left to right.

d) True – Variable names cannot start with a number.

---

2.10 Output of Java statements:

Assuming x = 2 and y = 3:

a) System.out.printf("x = %d%n", x); → Output: x = 2

b) System.out.printf("Value of %d + %d is %d%n", x, x, (x + x)); → Output: Value of 2 + 2 is 4

c) System.out.printf("x ="); → Output: x = (no newline)

d) System.out.printf("%d = %d%n", (x + y), (y + x)); → Output: 5 = 5

---

2.11 Statements that modify variables:

a) Yes, p = i + j + k + 7; modifies p.

b) No, System.out.println("variables whose values are modified"); does not modify a variable.

c) No, System.out.println("a = 5"); does not modify a variable.

d) Yes, value = input.nextInt(); modifies value.

---

2.12 Correct Java statements for y = ax³ + 7:

a) Correct → y = a \* x \* x \* x + 7;

b) Incorrect → Adds 7 inside multiplication.

c) Incorrect → Alters order of operations.

d) Correct → y = (a \* x) \* x \* x + 7;

e) Correct → y = a \* (x \* x \* x) + 7;

f) Incorrect → Adds 7 inside multiplication.

---

2.13 Operator precedence and final value of x:

a) x = 7 + 3 \* 6 / 2 - 1;

3 \* 6 = 18

18 / 2 = 9

7 + 9 = 16

16 - 1 = 15

Final value: x = 15

b) x = 2 % 2 + 2 \* 2 - 2 / 2;

2 % 2 = 0

2 \* 2 = 4

2 / 2 = 1

0 + 4 = 4

4 - 1 = 3

Final value: x = 3

c) x = (3 \* 9 \* (3 + (9 \* 3 / (3))));

9 \* 3 = 27

27 / 3 = 9

3 + 9 = 12

3 \* 9 = 27

27 \* 12 = 324

Final value: x = 324

// 2.19 What does the following code print?

public class PrintPattern1 {

public static void main(String[] args) {

System.out.printf("%n%n%n\*%n%n");

}

}

Output:

\*

\*\*

\*\*\*

\*\*\*\*

// 2.20 What does the following code print?

public class PrintPattern2 {

public static void main(String[] args) {

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

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

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

System.out.println("");

System.out.println("");

}

}

Output:

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*

// 2.21 What does the following code print?

public class PrintPattern3 {

public static void main(String[] args) {

System.out.print("\*");

System.out.print("\*");

System.out.print("\*");

System.out.print("");

System.out.println("");

}

}

Output:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 2.22 What does the following code print?

public class PrintPattern4 {

public static void main(String[] args) {

System.out.print("\*");

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

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

System.out.print("");

System.out.println("");

}

}

Output:

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*

// 2.23 What does the following code print?

public class PrintPattern5 {

public static void main(String[] args) {

System.out.printf("%s%n%s%n%s%n", "", "", "\*");

}

}

Output:

\*

\*\*\*

\*\*\*\*\*