**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) **Blank**, **tab**, 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; they do **not** cause the computer to print anything on the screen.

b) ✅ **True** – All variables must be declared with a type before they are used.

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

d) ❌ **False** – The remainder operator (%) can be used **with both integer and floating-point operands** (e.g., 5.5 % 2.0 is valid).

e) ❌ **False** – The operators \*, /, and % have **higher precedence** than + and -.

**✅ 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("%s%n%s", "This is a Java", "program");

f)

if (number != 7)

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

**✅ 2.4 Identify and correct the errors**

a) ❌ Error: Semicolon after the if statement ends it prematurely.  
✅ Corrected:

if (c < 7)

System.out.println("c is less than 7");

b) ❌ Error: Incorrect operator =>. Java uses >=.  
✅ Corrected:

if (c >= 7)

System.out.println("c is equal to or greater than 7");

**✅ 2.5 Write declarations/statements/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**

Here’s the full working Java program combining everything from **2.5**:

import java.util.Scanner; // Program uses class Scanner

public class Product {

public static void main(String[] args) {

// Create Scanner to obtain input from command window

Scanner input = new Scanner(System.in);

int x, y, z, result; // Variables to store integers

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

x = input.nextInt();

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

y = input.nextInt();

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

z = input.nextInt();

result = x \* y \* z; // Calculate product

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 **remainder (%)**.

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** – Java operators are **not always** evaluated strictly left to right. Operator **precedence** and **associativity** determine evaluation order.

b) ✅ **True** – All those are **valid variable names** because they follow Java’s naming rules.

c) ❌ **False** – The expression is evaluated according to **operator precedence**, not strictly left to right (e.g., \* and / before + and -).

d) ✅ **True** – Variable names cannot start with a digit, so 3g, 87, 67h2, h22, and 2h are invalid.

**✅ 2.10 Assuming x = 2 and y = 3, what is displayed?**

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 =

(cursor stays on the same line)

d)

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

**Output:**

5 = 5

**✅ 2.11 Which statements modify variable values?**

a) ✅ p = i + j + k + 7; → modifies p.  
b) ❌ System.out.println("variables whose values are modified"); → prints text only.  
c) ❌ System.out.println("a = 5"); → prints text only.  
d) ✅ value = input.nextInt(); → modifies value.

**✅ 2.12 Given y = a\*x³ + 7, which are correct Java statements?**

Correct statements that match y = a\*x\*x\*x + 7 are:  
✅ (a), (d), and (e).

* (a) y = a \* x \* x \* x + 7; ✅ correct
* (d) y = (a \* x) \* x \* x + 7; ✅ correct
* (e) y = a \* (x \* x \* x) + 7; ✅ correct

Others are ❌ incorrect because they change the intended expression.

**✅ 2.13 Order of evaluation**

a)

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

**Order:** \*, /, then +, -  
→ 3 \* 6 = 18  
→ 18 / 2 = 9  
→ 7 + 9 = 16  
→ 16 - 1 = 15  
✅ **x = 15**

b)

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

**Order:** %, \*, /, +, -  
→ 2 % 2 = 0  
→ 2 \* 2 = 4  
→ 2 / 2 = 1  
→ 0 + 4 = 4  
→ 4 - 1 = 3  
✅ **x = 3**

c)

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

Innermost parentheses first:  
→ (9 \* 3 / 3) = 27 / 3 = 9  
→ (3 + 9) = 12  
→ (3 \* 9 \* 12) = 27 \* 12 = 324  
✅ **x = 324**

**✅ 2.14 Display numbers 1–4 on the same line, separated by spaces**

a) **Using one println:**

System.out.println("1 2 3 4");

b) **Using four print statements:**

System.out.print("1 ");

System.out.print("2 ");

System.out.print("3 ");

System.out.print("4\n");

c) **Using one printf:**

System.out.printf("%d %d %d %d%n", 1, 2, 3, 4);

**✅ 2.15 (Arithmetic)**

**Program:**

import java.util.Scanner;

public class Arithmetic {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int number1, number2, sum, product, difference, quotient;

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

number1 = input.nextInt();

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

number2 = input.nextInt();

sum = number1 + number2;

product = number1 \* number2;

difference = number1 - number2;

quotient = number1 / number2;

System.out.printf("Sum is %d%n", sum);

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

System.out.printf("Difference is %d%n", difference);

System.out.printf("Quotient is %d%n", quotient);

}

}

**✅ 2.16 (Comparing Integers)**

import java.util.Scanner;

public class Comparison {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int number1, number2;

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

number1 = input.nextInt();

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

number2 = input.nextInt();

if (number1 > number2)

System.out.printf("%d is larger%n", number1);

else if (number2 > number1)

System.out.printf("%d is larger%n", number2);

else

System.out.println("These numbers are equal");

}

}

**✅ 2.17 (Sum, Average, Product, Smallest, Largest)**

import java.util.Scanner;

public class ThreeIntegers {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int x, y, z, sum, average, product, smallest, largest;

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

x = input.nextInt();

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

y = input.nextInt();

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

z = input.nextInt();

sum = x + y + z;

average = sum / 3;

product = x \* y \* z;

// Determine smallest

smallest = x;

if (y < smallest) smallest = y;

if (z < smallest) smallest = z;

// Determine largest

largest = x;

if (y > largest) largest = y;

if (z > largest) largest = z;

System.out.printf("Sum is %d%nAverage is %d%nProduct is %d%nSmallest is %d%nLargest is %d%n",

sum, average, product, smallest, largest);

}

}

**✅ 2.18 Displaying Shapes with Asterisks**

public class Shapes {

public static void main(String[] args) {

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

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

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

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

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

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

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

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

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

}

}

**✅ 2.19–2.22 Output Questions**

**2.19**

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

**Output:**

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

**2.20**

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

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

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

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

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

**Output:**

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*

**2.21**

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

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

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

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

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

**Output:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\* (all on one line)

Actually displayed as:

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

(1 + 3 + 5 + 4 + 2 = 15 asterisks, all on the same line)

**2.22**

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

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

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

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

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

**Output:**

\*\*\*\*

\*\*\*\*\*

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

(Explanation: print("\*") + next line \*\*\* gives \*\*\*\* on first line)