CLASS -10 (2025-26)

**INPUT IN JAVA**

**CHAPTER 5**

**Assignments:-**

**1. Identify and explain the problem with the following code fragment:**

int cts;

char answer;

cts = 10;

answer cts; // Error

**Problem:**

answer cts; is invalid syntax. It looks like an assignment is attempted, but the assignment operator = is missing.

**Correct version:**

answer = (char) cts; // if type conversion is intended

**2. In an expression, what type are byte and short promoted to?**

**Answer:**  
In expressions, **byte and short are promoted to int** before the operation is performed.

**3. Are the following statements legal? Why or why not?**

short s1 = 10;

short s2 = 10;

short result = s1 + s2; // Illegal

**Answer:**  
**Illegal**. s1 + s2 is promoted to int, and assigning it directly to a short causes a type mismatch.

**Fix:**

short result = (short)(s1 + s2);

**4. What is arithmetic promotion? What is coercion?**

* **Arithmetic Promotion:** Automatic conversion of smaller data types (byte, short, char) to int (or larger types like float, double) in arithmetic operations.
* **Coercion:** Implicit or explicit conversion of one data type to another (e.g., int to double).

**5. What types can you assign a short to without explicit casting?**

**Answer:**  
You can assign a short to:

* int
* long
* float
* double

These are **widening conversions** and do **not require casting**.

**6. What is casting and how do you do it?**

**Answer:**  
**Casting** is explicitly converting a value from one type to another.

**Syntax:**

int i = (int) 3.14; // Cast double to int

**When needed:**

* When converting from a larger to smaller type (double to int, int to byte)
* When assigning between incompatible types

**7. Why would you want to use an object wrapper rather than a primitive type?**

**Answer:**

* **Wrapper classes** allow primitives to be used in **collections** (like ArrayList<Integer>)
* They provide **utility methods**
* Needed for **nullability**, **generics**, and **object manipulation**

**8. What are wrapper classes? How is Integer different from int?**

**Answer:**

* **Wrapper classes** wrap primitive types into objects (Integer, Double, etc.)
* int is a primitive; Integer is an object class with extra features.

**9. What are Wrapper classes? Give any two examples.**

**Answer:**  
Wrapper classes wrap primitive data types into objects.

Examples:

* Integer for int
* Double for double

**10. What is autoboxing?**

**Answer:**

**Autoboxing** is the **automatic conversion** of a primitive to its corresponding wrapper class.

Example:

int x = 5;

Integer obj = x; // Autoboxing

**11. What is unboxing?**

**Answer:**

**Unboxing** is the **automatic conversion** of a wrapper class object to its corresponding primitive type.

Example:

Integer obj = 5;

int x = obj; // Unboxing

**12. When to prefer primitive types vs wrapper classes?**

**Answer:**

* **Prefer primitives**: When performance and memory efficiency matter.
* **Use wrappers**: When you need to work with collections, generics, or null values.

**13. Program: Converter.java (Kilometers to Feet and Light Years)**

**Answer:**

public class Converter {

public static void main(String[] args) {

System.out.println("This program converts kilometers into feet and light years.");

double kilometers = Double.parseDouble(args[0]);

double feet = kilometers \* 3280.839895013;

double lightYears = kilometers / 9460730472580.8;

System.out.println("The number of kilometers: " + kilometers);

System.out.println("This is equal to " + feet + " feet and " + lightYears + " light years.");

}

}

**Sample run:**  
If you enter 145 as a command-line argument, the output will be:

This program converts kilometers into feet and light years.

The number of kilometers: 145.0

This is equal to 475721.784776885 feet and 1.5326512093356922E-11 light years.

**14. Output of the Given Code**

**Answer:**

int number;

number = 10; //1

System.out.println("1= "+ number); // Output: 1= 10

number = 10+6; //2

System.out.println("2="+ number); // Output: 2=16

number = 10+6\*7; //3 => 10 + (6\*7) = 10 + 42 = 52

System.out.println("3="+ number); // Output: 3=52

number = 10 + 6 \* 7 / 2; //4 => 6\*7=42, then 42/2=21, then 10+21=31

System.out.println("4 = "+number); // Output: 4 = 31

number = 10 + 7 / 2 \* 6 - 2; //5

// 7/2 = 3 (int division), 3\*6 = 18, 10+18=28, 28-2=26

System.out.println("5"+ number); // Output: 526

**15. Program: Change.java**

**Answer:**

import java.util.Scanner;

public class Change {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter an amount between Rs 100 and Rs 1000: ");

int amount = sc.nextInt();

int[] denominations = {50, 20, 10, 5, 2, 1};

System.out.println("Minimum number of notes/coins:");

for (int note : denominations) {

int count = amount / note;

amount %= note;

if (count > 0)

System.out.println(note + " x " + count);

}

}

}

**16. Output of Given Code**

**Answer:**

**(a)**

byte x = 64, y;

y = (byte) (x << 2); // 64 << 2 = 256 => overflow => (byte)256 = 0

System.out.println(y); // Output: 0

**(b)** (Binary Logic Operations)

Let’s convert them:

* 00110011 = 0x33 = 51
* 11110000 = 0xF0 = 240

(i) 00110011 & 11110000 = 00110000 = **48**  
(ii) 00110011 ^ 11110000 = 11000011 = **195**  
(iii) 00110011 | 11110000 = 11110011 = **243**

**17. Output Prediction**

**Answer:**

**(i)**

byte b;

double d = 417.35;

b = (byte) d;

System.out.println(b); // Output: 417 % 256 = 161 (because byte range is -128 to 127), so Output: \*\*-95\*\*

**(ii)**

int x = 10;

int y = 20;

if ((x < y) || (x = 5) > 10)

System.out.println(x); // x < y is true, so second condition not evaluated due to short-circuit

else

System.out.println(y);

**Output: 10**

**18. Understanding the Output**

**Answer:**

int i = 5;

System.out.println(++i); // Pre-increment: i becomes 6, prints 6

System.out.println(i++); // Post-increment: prints 6, then i becomes 7

**Output:**

6

6

**Explanation:**

* First ++i: increments **before** use → i = 6, prints **6**
* Then i++: uses value **before** increment → prints **6**, then i becomes 7