1. **Arithmetic & Assignment Operators**
2. **Q1:** Write a program to swap two numbers **without using a third variable** and without using arithmetic operators like + or - .

**Hint** : Use bitwise XOR ^ operator.

public class swapping {

public static void main(String[] args) {

int a = 5;

int b = 10;

System.out.println("Before swapping: a = " + a + ", b = " + b);

a = a ^ b;

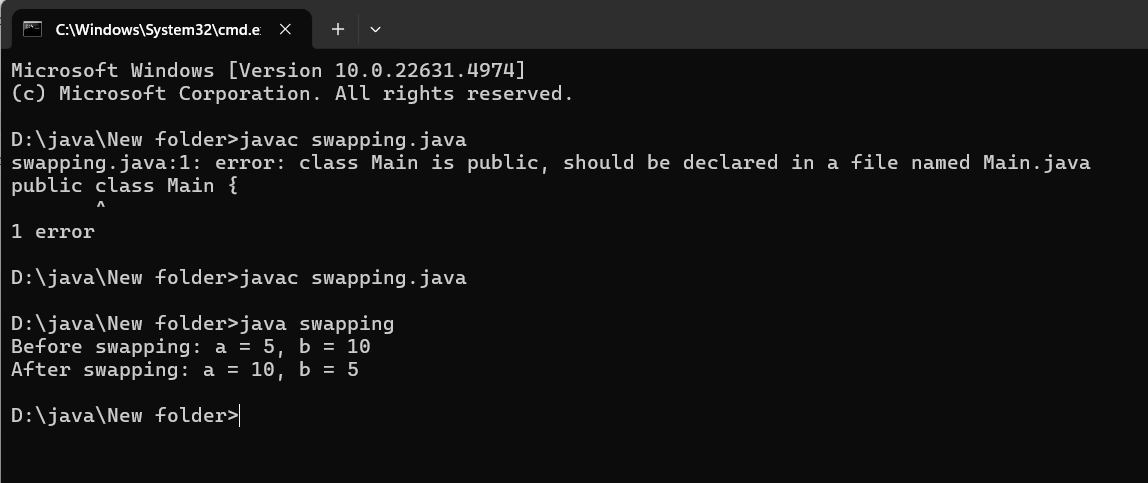
b = a ^ b;

a = a ^ b;

System.out.println("After swapping: a = " + a + ", b = " + b);

}

}



**Q2:** Write a program to check whether a given number is **even or odd** using only **bitwise**

**operators** .

**Hint** : Use n & 1 to check.

public class EvenOddCheck {

public static void main(String[] args) {

int number = 7;

if ((number & 1) == 0) {

System.out.println(number + " is Even.");

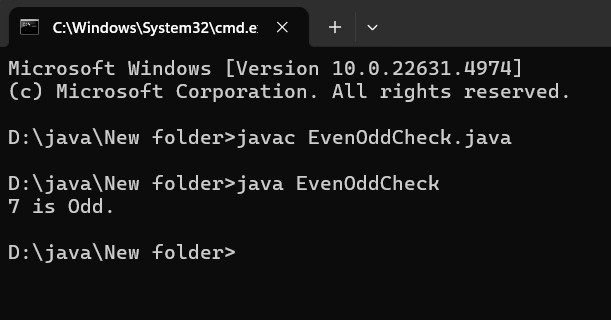
} else {

System.out.println(number + " is Odd.");

}

}

}



**Q3:** Implement a program that calculates the **sum of digits** of an integer using **modulus**

**( % ) and division ( / ) operators** .

public class SumOfDigits {

public static void main(String[] args) {

int number = 12345;

int sum = 0;

while (number != 0) {

sum += number % 10;

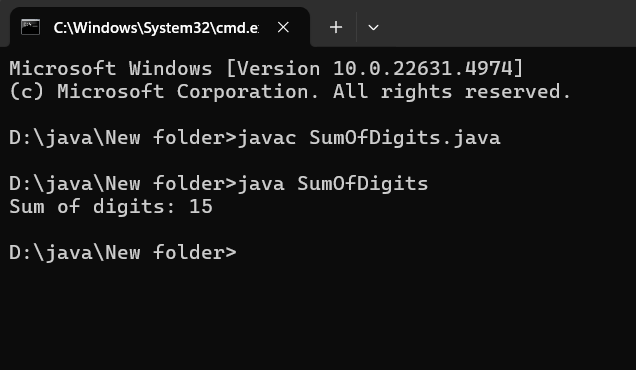
number = number / 10;

}

System.out.println("Sum of digits: " + sum);

}

}



**Q4:** Write a program to find whether a given number is **divisible by 3** without using the

modulus ( % ) or division ( / ) operators.

**Hint** : Use **subtraction and bitwise shifts** .

public class DivisibleByThree {

public static void main(String[] args) {

int number = 9;

if (isDivisibleByThree(number)) {

System.out.println(number + " is divisible by 3.");

} else {

System.out.println(number + " is not divisible by 3.");

}

}

public static boolean isDivisibleByThree(int number) {

while (number >= 3) {

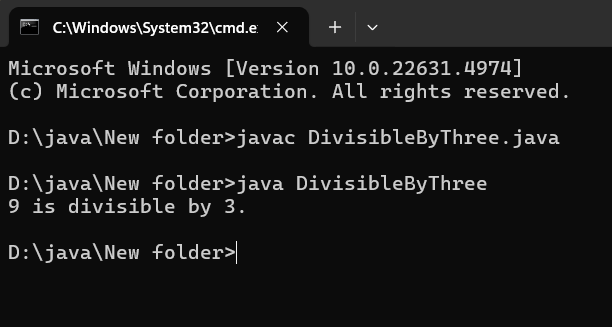
number -= 3;

}

return number == 0;

}

}



**Q5:** Write a Java program to **swap two numbers** using the += and -= operators only

public class SwapNumbers {

public static void main(String[] args) {

int a = 5;

int b = 10;

System.out.println("Before swapping:");

System.out.println("a = " + a + ", b = " + b);

a += b;

b = a - b;

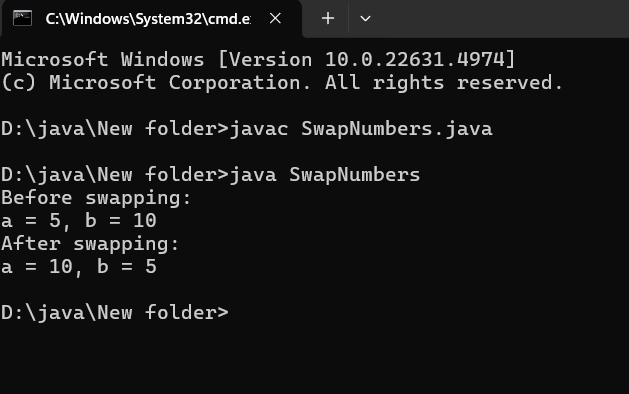
a -= b;

System.out.println("After swapping:");

System.out.println("a = " + a + ", b = " + b);

}

}



**2.Relational & Logical Operator:**

import java.util.Scanner;

public class LargestNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter three numbers: ");

int num1 = scanner.nextInt();

int num2 = scanner.nextInt();

int num3 = scanner.nextInt();

int largest = (num1 >= num2 && num1 >= num3) ? num1 :

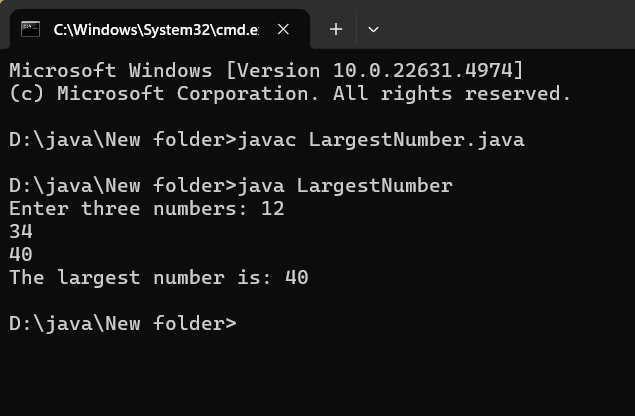
(num2 >= num1 && num2 >= num3) ? num2 : num3;

System.out.println("The largest number is: " + largest);

scanner.close();

}

}



Q7: Implement a Java program that checks whether a given year is a leap year or not using logical ( && , || ) operators in java

import java.util.Scanner;

public class LeapYearCheck {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a year: ");

int year = scanner.nextInt();

boolean isLeapYear = (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0));

if (isLeapYear) {

System.out.println(year + " is a leap year.");

} else {

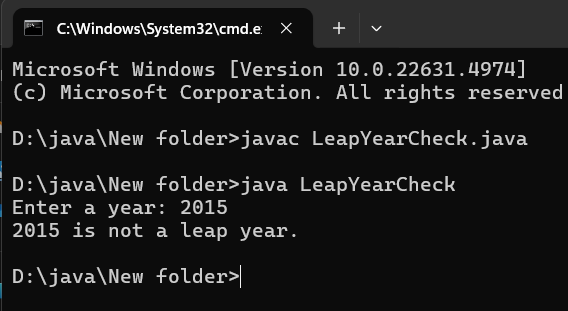
System.out.println(year + " is not a leap year.");

}

scanner.close();

}

}



Q8: Write a program that takes three boolean inputs and prints true if at least two of them are true . Hint : Use logical operators ( && , || ). in java

import java.util.Scanner;

public class BooleanCheck {

public static void main(String[] args) {

// Create a scanner object to take input from the user

Scanner scanner = new Scanner(System.in);

// Input three boolean values

System.out.print("Enter first boolean value (true/false): ");

boolean bool1 = scanner.nextBoolean();

System.out.print("Enter second boolean value (true/false): ");

boolean bool2 = scanner.nextBoolean();

System.out.print("Enter third boolean value (true/false): ");

boolean bool3 = scanner.nextBoolean();

// Check if at least two of the boolean values are true

boolean result = (bool1 && bool2) || (bool1 && bool3) || (bool2 && bool3);

// Output the result

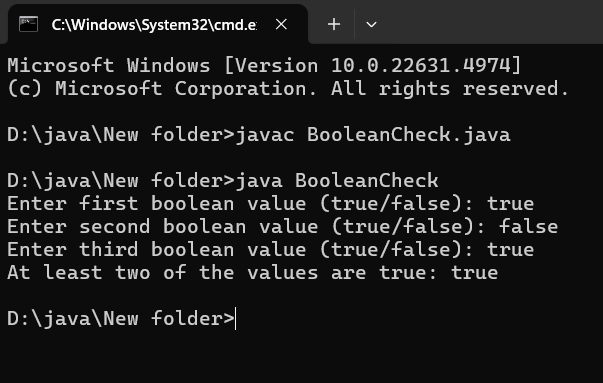
System.out.println("At least two of the values are true: " + result);

// Close the scanner

scanner.close();

}

}



Q10: Write a program to determine if a character is a vowel or a consonant using the ternary operatorin

import java.util.Scanner;

public class VowelOrConsonant {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a character: ");

char ch = scanner.next().charAt(0);

ch = Character.toLowerCase(ch);

String result = (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u')

? "Vowel"

: (ch >= 'a' && ch <= 'z')

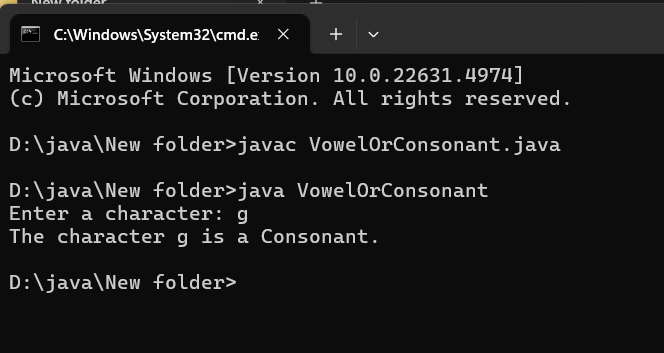
? "Consonant"

: "Invalid Input";

System.out.println("The character " + ch + " is a " + result + ".");

}

}



1. **Bitwise Operators:**

Q11: Write a program to check if a given number is a power of 2 using bitwise operators. Hint : n & (n - 1) == 0 for positive numbers. in java

import java.util.Scanner;

public class PowerOfTwo {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number: ");

int n = scanner.nextInt();

boolean isPowerOfTwo = (n > 0) && (n & (n - 1)) == 0;

if (isPowerOfTwo) {

System.out.println(n + " is a power of 2.");

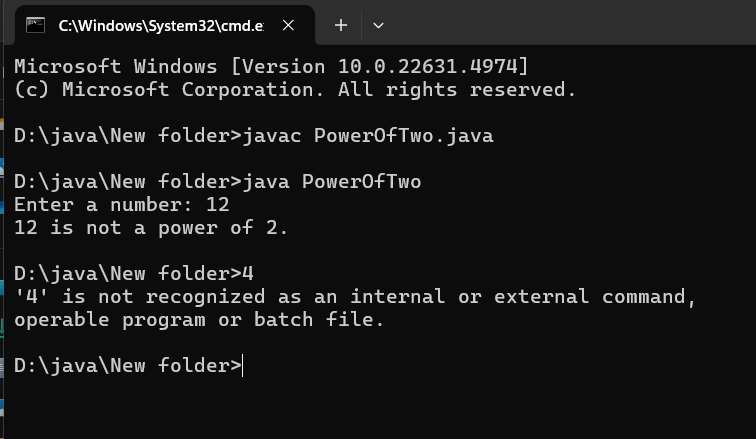
} else {

System.out.println(n + " is not a power of 2.");

}

}

}



**Q12:** Write a Java program to **multiply a number by 8** without using \* or / operators.

**Hint** : Use bitwise left shift ( << ).

import java.util.Scanner;

public class MultiplyByEight {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number: ");

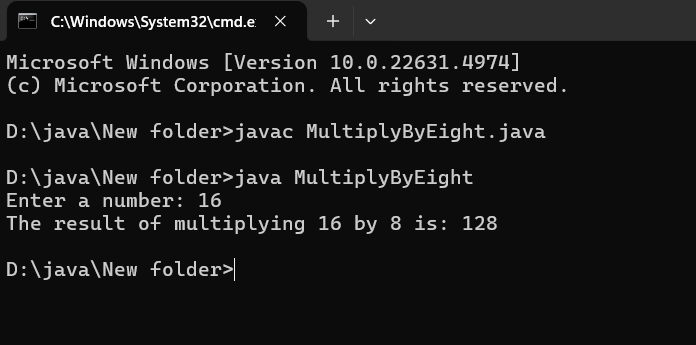
int num = scanner.nextInt();

int result = num << 3;

System.out.println("The result of multiplying " + num + " by 8 is: " + result);

}

}



**Q13:** Implement a Java program to find the **absolute value** of an integer using bitwise

operators.

**Hint** : mask = num >> 31; abs = (num + mask) ^ mask;

import java.util.Scanner;

public class AbsoluteValue {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int num = scanner.nextInt();

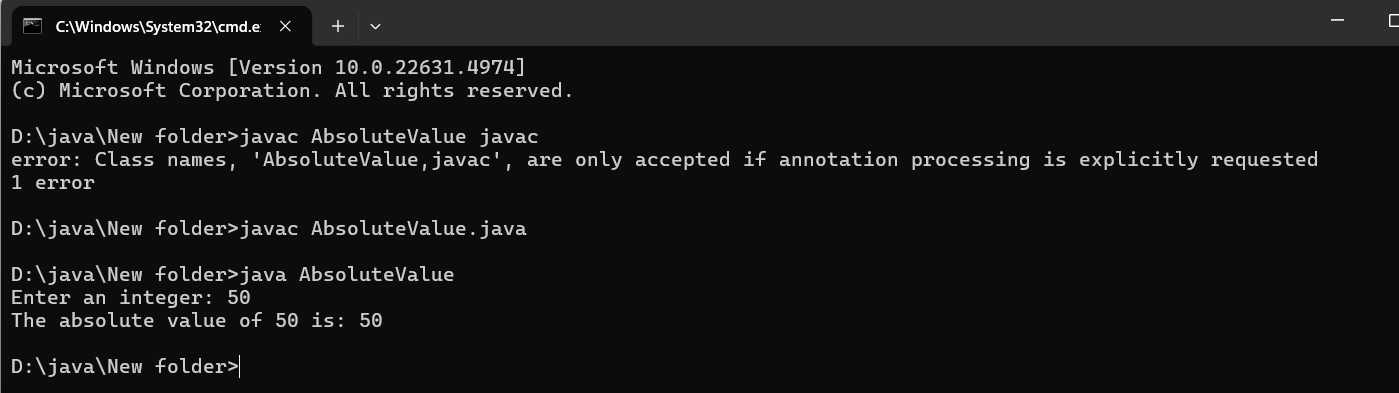
int mask = num >> 31;

int abs = (num + mask) ^ mask;

System.out.println("The absolute value of " + num + " is: " + abs);

}

}



**Q14:** Write a program to count the **number of 1s (set bits)** in a binary representation of a

number using bitwise operations.

**Hint** : Use n & (n - 1) .

public class SetBitCounter {

public static int countSetBits(int n) {

int count = 0;

while (n > 0) {

count++;

}

return count;

}

public static void main(String[] args) {

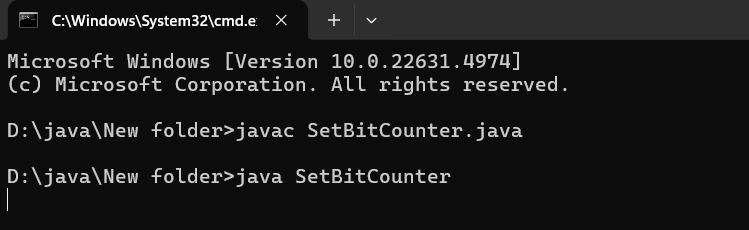
int number = 29;

int result = countSetBits(number);

System.out.println("Number of set bits in " + number + " is: " + result);

}

}



**Q15:** Implement a program to swap **odd and even bits** of a number using bitwise

operators.

**Hint** : Use masks: (x & 0xAAAAAAAA) >> 1 | (x & 0x55555555) << 1 .

public class BitSwapper {

public static int swapBits(int x) {

int EVEN\_MASK = 0xAAAAAAAA; // Binary: 10101010...

int ODD\_MASK = 0x55555555; // Binary: 01010101...

int evenBits = x & EVEN\_MASK;

int oddBits = x & ODD\_MASK;

evenBits >>= 1;

oddBits <<= 1;

return evenBits | oddBits;

}

public static void main(String[] args) {

int number = 29; // Example number

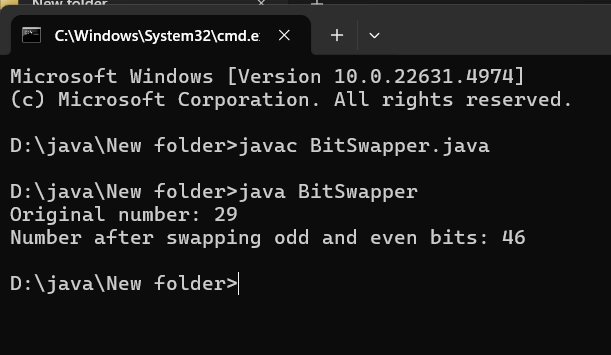
int swappedNumber = swapBits(number);

System.out.println("Original number: " + number);

System.out.println("Number after swapping odd and even bits: " + swappedNumber);

}

}



1. **Ternary Operator Challenges:**
2. **Q16:** Write a program that determines whether a given number is **positive, negative, or**
3. **zero** using only the **ternary operator**

**public class NumberClassifier {**

**public static void main(String[] args) {**

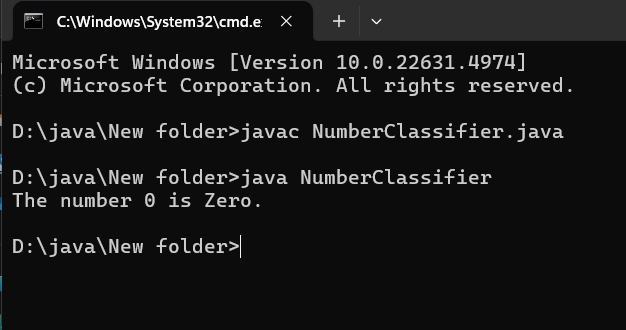
**int number = 0;**

**String result = (number > 0) ? "Positive" : ((number < 0) ? "Negative" : "Zero");**

**System.out.println("The number " + number + " is " + result + ".");**

**}**

**}**



**Q17:** Implement a Java program that finds the **minimum of four numbers** using nested

ternary operators

public class MinOfFour {

public static void main(String[] args) {

int a = 5, b = 3, c = 8, d = 1;

int min = (a < b)

? ((a < c)

? ((a < d) ? a : d)

: ((c < d) ? c : d))

: ((b < c)

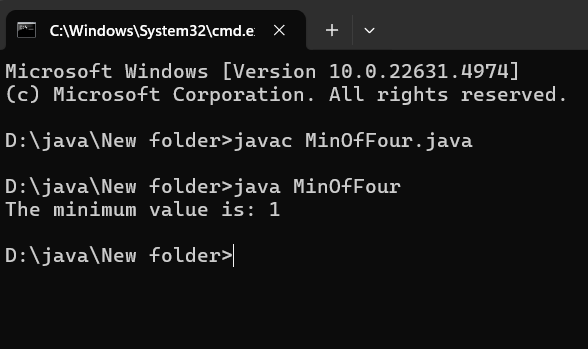
? ((b < d) ? b : d)

: ((c < d) ? c : d));

System.out.println("The minimum value is: " + min);

}

}



**Q18:** Given a student’s percentage, print **“Pass”** if the percentage is 40 or above;

otherwise, print **“Fail”** , using only the ternary operator.

public class StudentResult {

public static void main(String[] args) {

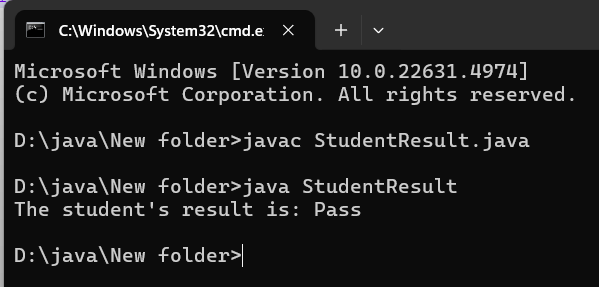
double percentage = 45.0;

String result = (percentage >= 40) ? "Pass" : "Fail";

System.out.println("The student's result is: " + result);

}

}



**Q19:** Write a Java program that checks whether a character is **uppercase, lowercase, or**

**not a letter** using only the ternary operator.

public class CharacterType {

public static void main(String[] args) {

char ch = 'A'; // Example character; modify as needed

// Determine character type using nested ternary operators

String result = (ch >= 'A' && ch <= 'Z') ? "Uppercase"

: (ch >= 'a' && ch <= 'z') ? "Lowercase"

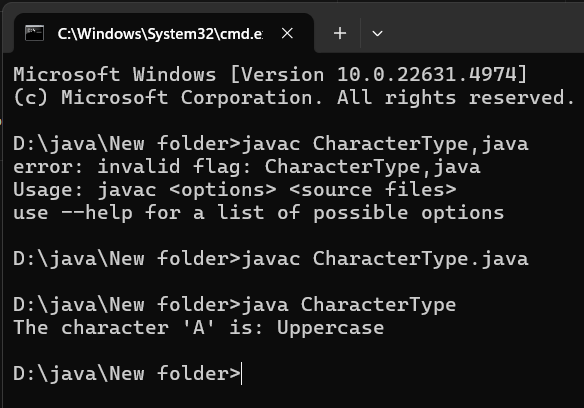
: "Not a letter";

// Output the result

System.out.println("The character '" + ch + "' is: " + result);

}

}



Q20: Implement a Java program that returns the absolute value of a given number using the ternary operator (without using Math.abs() ).

public class AbsoluteValue1 {

public static void main(String[] args) {

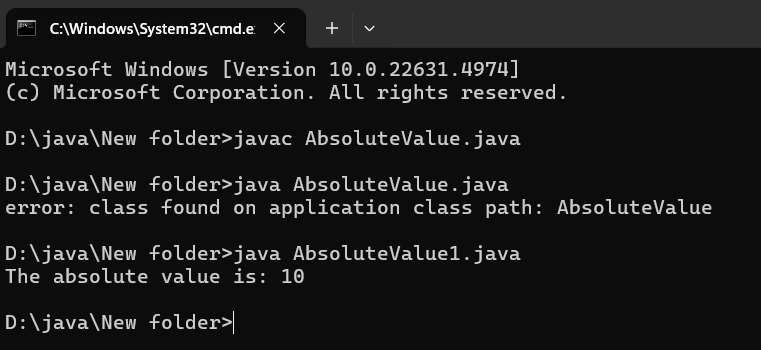
int number = -10;

int absoluteValue = (number >= 0) ? number : -number;

System.out.println("The absolute value is: " + absoluteValue);

}

}



**5. Miscellaneous Operator Questions:**

Q21: Write a program that increments a number without using + or ++ operators. Hint : Use bitwise - (~x) .

**public class BitwiseIncrement {**

**public static void main(String[] args) {**

**int number = 5;**

**int incrementedNumber = addOne(number);**

**System.out.println("Original number: " + number);**

**System.out.println("Incremented number: " + incrementedNumber);**

**}**

**public static int addOne(int num) {**

**int mask = 1;**

**int carry;**

**while ((num & mask) == mask) {**

**num = num ^ mask;**

**carry = mask << 1;**

**mask = carry;**

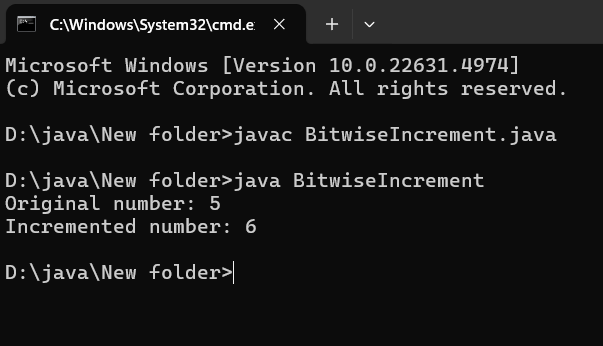
**}**

**num = num ^ mask; // Flip the last 0 bit to 1**

**return num;**

**}**

**}**

****

**Q22:** Implement a **calculator** that takes two numbers and an operator ( + , - , \* , / ) as input

and prints the result using only **switch-case** .

**import java.util.Scanner;**

**public class Calculator {**

**public static void main(String[] args) {**

**// Create a Scanner object for user input**

**Scanner scanner = new Scanner(System.in);**

**// Prompt the user to enter the first number**

**System.out.print("Enter first number: ");**

**double num1 = scanner.nextDouble();**

**// Prompt the user to enter the operator**

**System.out.print("Enter operator (+, -, \*, /): ");**

**char operator = scanner.next().charAt(0);**

**// Prompt the user to enter the second number**

**System.out.print("Enter second number: ");**

**double num2 = scanner.nextDouble();**

**// Declare a variable to store the result**

**double result;**

**// Use a switch-case statement to perform the operation based on the operator**

**switch (operator) {**

**case '+':**

**result = num1 + num2;**

**System.out.println("Result: " + result);**

**break;**

**case '-':**

**result = num1 - num2;**

**System.out.println("Result: " + result);**

**break;**

**case '\*':**

**result = num1 \* num2;**

**System.out.println("Result: " + result);**

**break;**

**case '/':**

**// Check for division by zero**

**if (num2 != 0) {**

**result = num1 / num2;**

**System.out.println("Result: " + result);**

**} else {**

**System.out.println("Error: Division by zero is not allowed.");**

**}**

**break;**

**default:**

**System.out.println("Error: Invalid operator.");**

**break;**

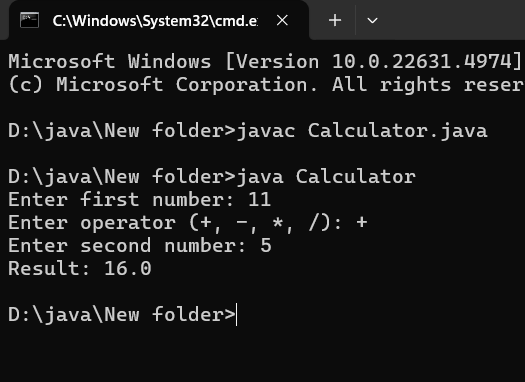
**}**

**// Close the scanner**

**scanner.close();**

**}**

**}**

****

**Q23:** Given a number, find whether it is **odd or even** using the & bitwise operator and print

the result without using if-else .

**public class EvenOddChecker1{**

**public static void main(String[] args) {**

**int number = 7; // Example number; modify as needed**

**// Determine if the number is even or odd using the bitwise AND operator**

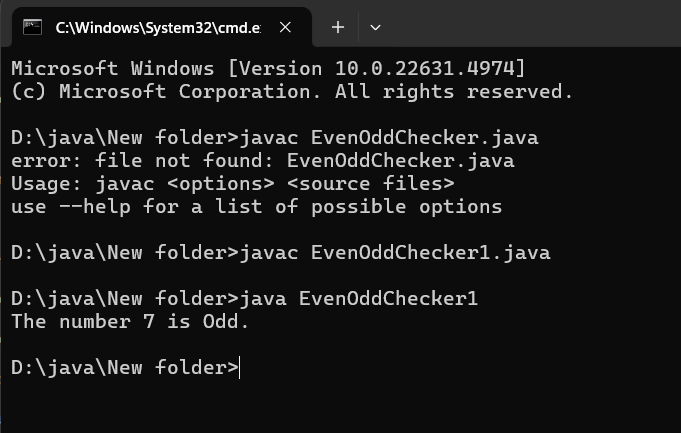
**String result = (number & 1) == 0 ? "Even" : "Odd";**

**// Output the result**

**System.out.println("The number " + number + " is " + result + ".");**

**}**

**}**

****

**Q24:** Write a program that prints **all even numbers from 1 to 100** using only **bitwise AND**

**( & )** and for loop.

**public class EvenNumbers1 {**

**public static void main(String[] args) {**

**for (int i = 1; i <= 100; i++) {**

**if ((i & 1) == 0) {**

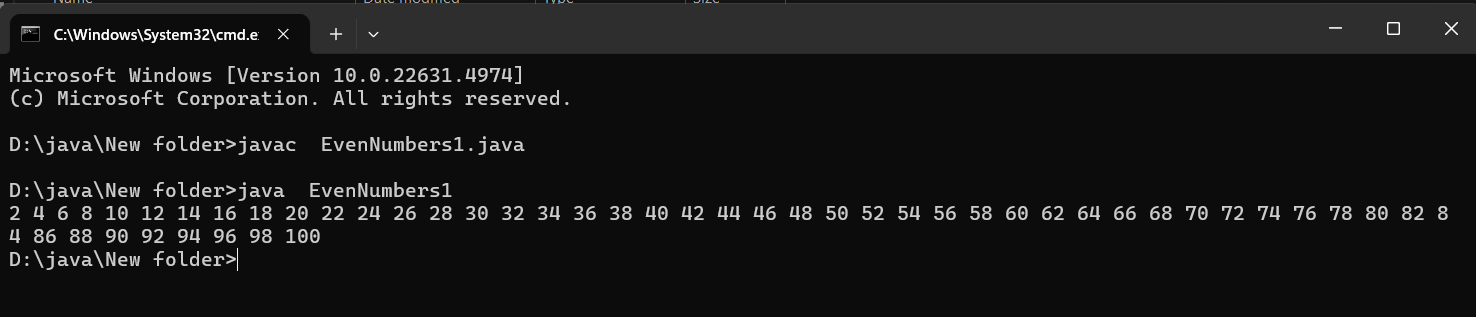
**System.out.print(i + " ");**

**}**

**}**

**}**

**}**

****

**Q25:** Implement a program that reverses an **integer number** without using string

conversion ( StringBuilder or toCharArray ).

**Hint** : Use while(n!=0) { rev = rev \* 10 + n % 10; n /= 10; }

public class ReverseInteger {

public static void main(String[] args) {

int number = 12345; // Example number; modify as needed

int reversedNumber = reverse(number);

System.out.println("Original number: " + number);

System.out.println("Reversed number: " + reversedNumber);

}

// Method to reverse the digits of an integer

public static int reverse(int n) {

int rev = 0;

while (n != 0) {

rev = rev \* 10 + n % 10; // Append the last digit to rev

n /= 10; // Remove the last digit from n

}

return rev;

}

}

