

Lab_2

1. Write a program that takes a student's score as input and outputs the corresponding grade based on the following scale:

A: 90-100

B: 80-89

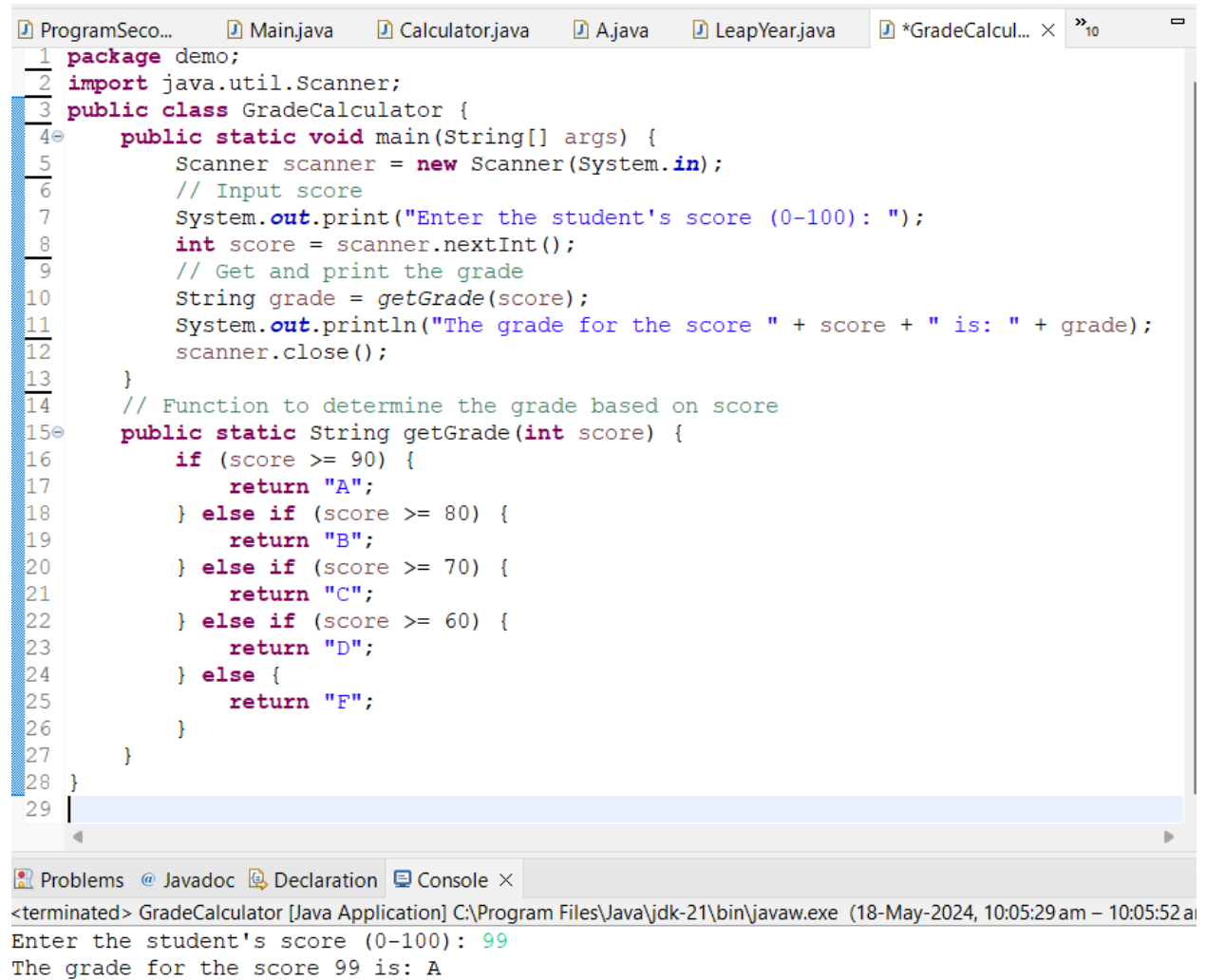
C: 70-79

D: 60-69

F: 0-59

Program:

```
package demo;
import java.util.Scanner;
public class GradeCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Input score
        System.out.print("Enter the student's score (0-100): ");
        int score = scanner.nextInt();
        // Get and print the grade
        String grade = getGrade(score);
        System.out.println("The grade for the score " + score + " is: " +
grade);
        scanner.close();
    }
    // Function to determine the grade based on score
    public static String getGrade(int score) {
        if (score >= 90) {
            return "A";
        } else if (score >= 80) {
            return "B";
        } else if (score >= 70) {
            return "C";
        } else if (score >= 60) {
            return "D";
        } else {
            return "F";
        }
    }
}
```

Output:

The screenshot displays an IDE with a Java file named `GradeCalculator.java`. The code defines a `GradeCalculator` class with a `main` method and a `getGrade` method. The `main` method prompts the user for a score (0-100), reads the input, and prints the corresponding grade. The `getGrade` method uses a series of `if-else` statements to determine the grade based on the score: A (90-100), B (80-89), C (70-79), D (60-69), and F (below 60).

```
1 package demo;
2 import java.util.Scanner;
3 public class GradeCalculator {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         // Input score
7         System.out.print("Enter the student's score (0-100): ");
8         int score = scanner.nextInt();
9         // Get and print the grade
10        String grade = getGrade(score);
11        System.out.println("The grade for the score " + score + " is: " + grade);
12        scanner.close();
13    }
14    // Function to determine the grade based on score
15    public static String getGrade(int score) {
16        if (score >= 90) {
17            return "A";
18        } else if (score >= 80) {
19            return "B";
20        } else if (score >= 70) {
21            return "C";
22        } else if (score >= 60) {
23            return "D";
24        } else {
25            return "F";
26        }
27    }
28 }
29
```

The console output shows the execution of the program. It starts with a terminated message, followed by the prompt "Enter the student's score (0-100):" and the user input "99". The final output is "The grade for the score 99 is: A".

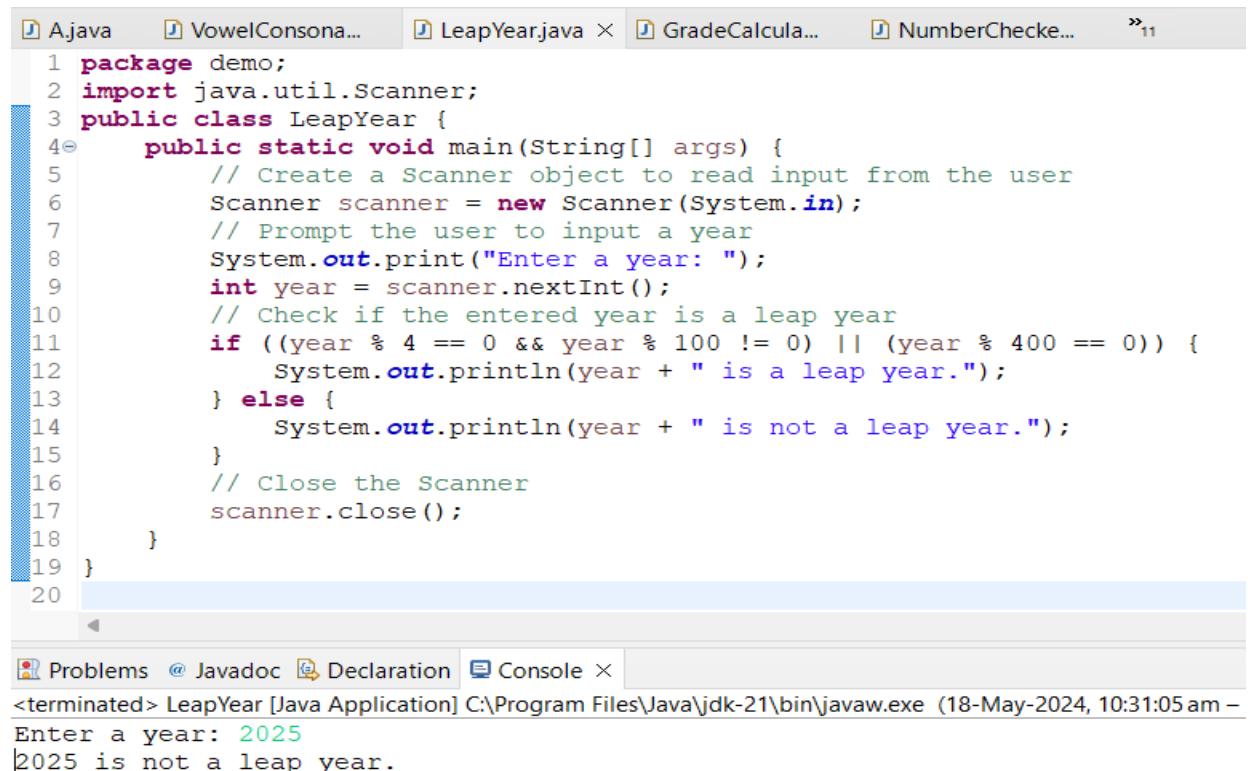
```
<terminated> GradeCalculator [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (18-May-2024, 10:05:29 am - 10:05:52 am)
Enter the student's score (0-100): 99
The grade for the score 99 is: A
```

2. Write a program to check if a given year is a leap year. (A year is a leap year if it is divisible by 4 but not by 100, or it is divisible by 400.)

Program:

```
package demo;
import java.util.Scanner;
public class LeapYear {
    public static void main(String[] args) {
        // Create a Scanner object to read input from the user
        Scanner scanner = new Scanner(System.in);
        // Prompt the user to input a year
        System.out.print("Enter a year: ");
        int year = scanner.nextInt();
        // Check if the entered year is a leap year
        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
            System.out.println(year + " is a leap year.");
        } else {
            System.out.println(year + " is not a leap year.");
        }
        // Close the Scanner
        scanner.close();
    }
}
```

Output:



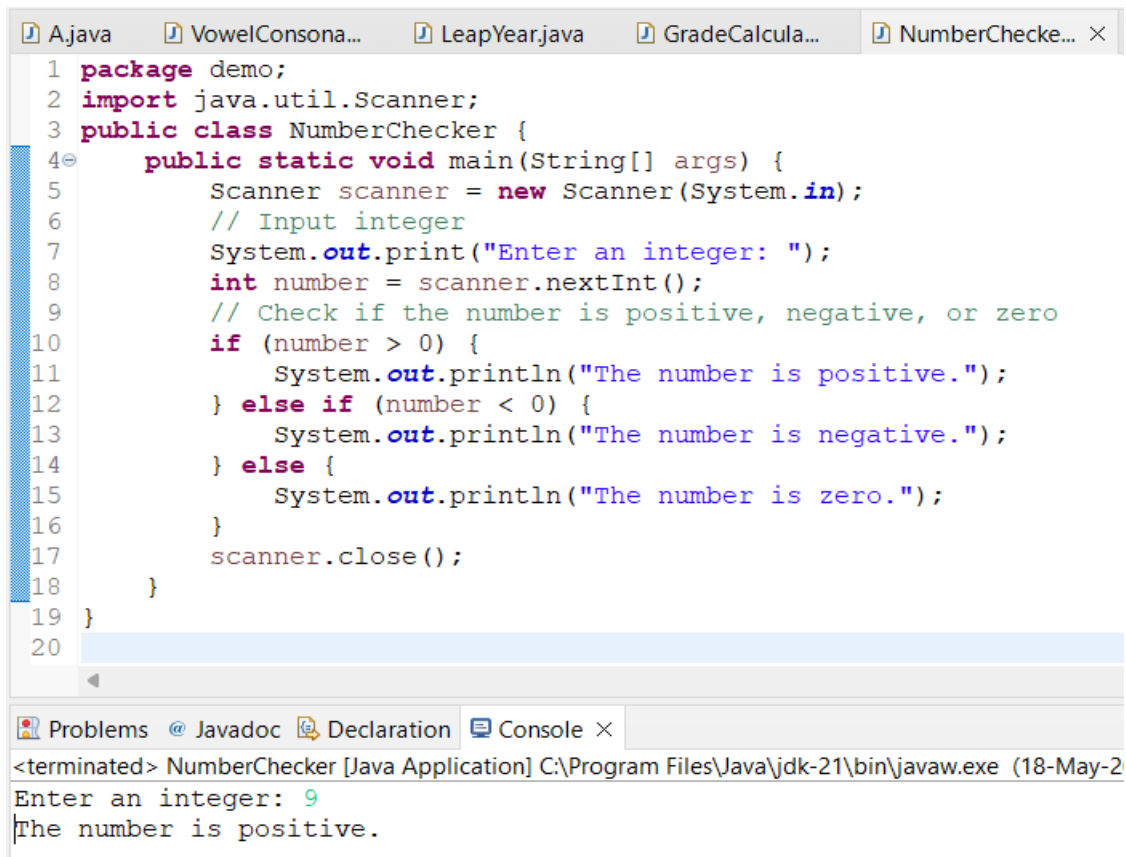
The screenshot shows an IDE with several tabs: A.java, VowelConsona..., LeapYear.java (active), GradeCalcula..., and NumberChecke... The code in LeapYear.java is displayed with line numbers 1 through 20. The console output at the bottom shows the program execution: <terminated> LeapYear [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (18-May-2024, 10:31:05 am - Enter a year: 2025 2025 is not a leap year.

3. Write a program that takes an integer as input and checks if it is positive, negative, or zero.

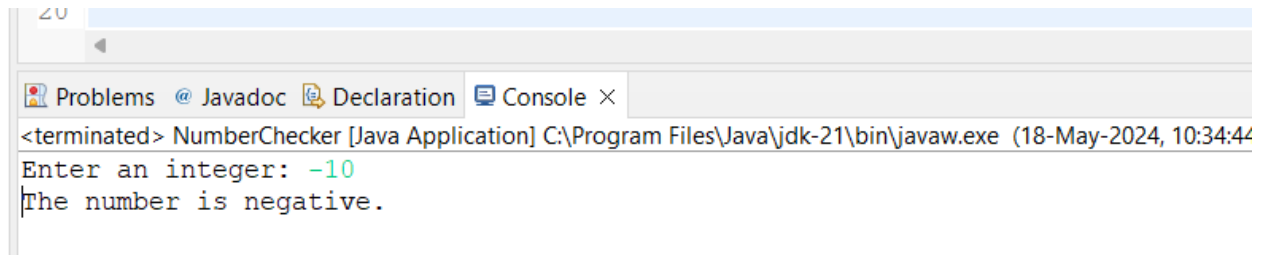
Program:

```
package demo;
import java.util.Scanner;
public class NumberChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Input integer
        System.out.print("Enter an integer: ");
        int number = scanner.nextInt();
        // Check if the number is positive, negative, or zero
        if (number > 0) {
            System.out.println("The number is positive.");
        } else if (number < 0) {
            System.out.println("The number is negative.");
        } else {
            System.out.println("The number is zero.");
        }
        scanner.close();
    }
}
```

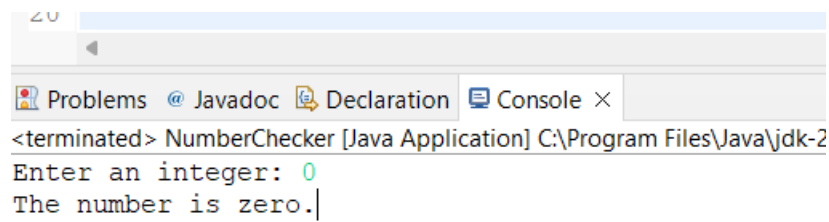
Output:



The screenshot shows an IDE with several tabs open: A.java, VowelConsona..., LeapYear.java, GradeCalcula..., and NumberChecke... (selected). The NumberChecker.java file is open, displaying the same code as in the 'Program' section. The console at the bottom shows the execution output: <terminated> NumberChecker [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (18-May-2024) Enter an integer: 9 The number is positive.



```
<terminated> NumberChecker [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (18-May-2024, 10:34:44)
Enter an integer: -10
The number is negative.
```



```
<terminated> NumberChecker [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (18-May-2024, 10:34:44)
Enter an integer: 0
The number is zero.
```

4. Write a program that prints numbers from 1 to 10 using a loop.

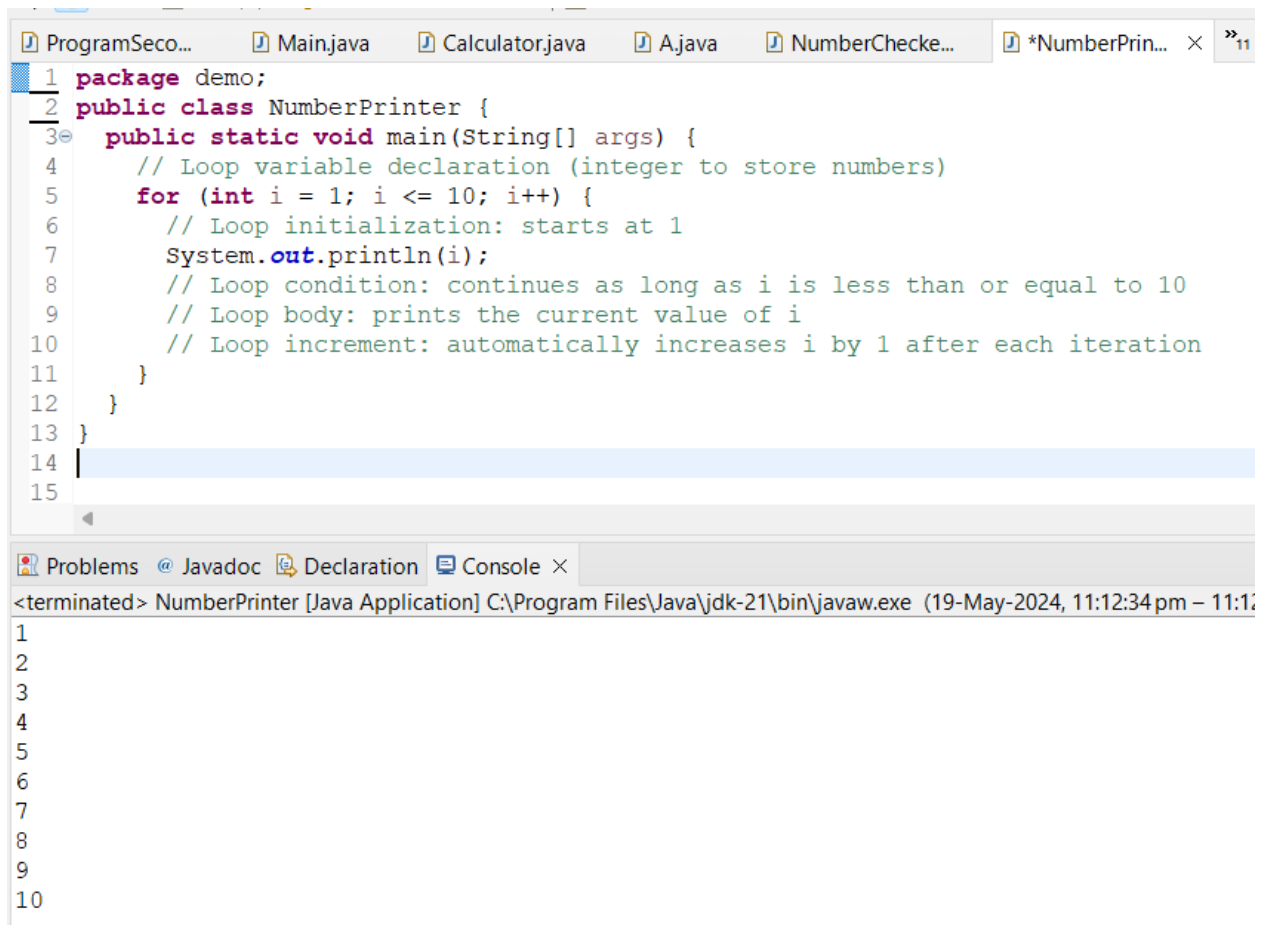
Program:

```
package demo;

public class NumberPrinter {

    public static void main(String[] args) {
        // Loop variable declaration (integer to store numbers)
        for (int i = 1; i <= 10; i++) {
            // Loop initialization: starts at 1
            System.out.println(i);
            // Loop condition: continues as long as i is less than or equal to 10
            // Loop body: prints the current value of i
            // Loop increment: automatically increases i by 1 after each iteration
        }
    }
}
```

Output:

The screenshot shows an IDE window with a tab for 'NumberPrinter.java'. The code is the same as shown in the previous block. Below the code editor, the 'Console' tab is active, displaying the output of the program: the numbers 1 through 10, each on a new line. The console title bar indicates the application is 'NumberPrinter [Java Application]' running on 'C:\Program Files\Java\jdk-21\bin\javaw.exe' at '19-May-2024, 11:12:34 pm'.

```
1 package demo;
2 public class NumberPrinter {
3     public static void main(String[] args) {
4         // Loop variable declaration (integer to store numbers)
5         for (int i = 1; i <= 10; i++) {
6             // Loop initialization: starts at 1
7             System.out.println(i);
8             // Loop condition: continues as long as i is less than or equal to 10
9             // Loop body: prints the current value of i
10            // Loop increment: automatically increases i by 1 after each iteration
11        }
12    }
13 }
14
15
```

<terminated> NumberPrinter [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (19-May-2024, 11:12:34 pm – 11:12:34 pm)

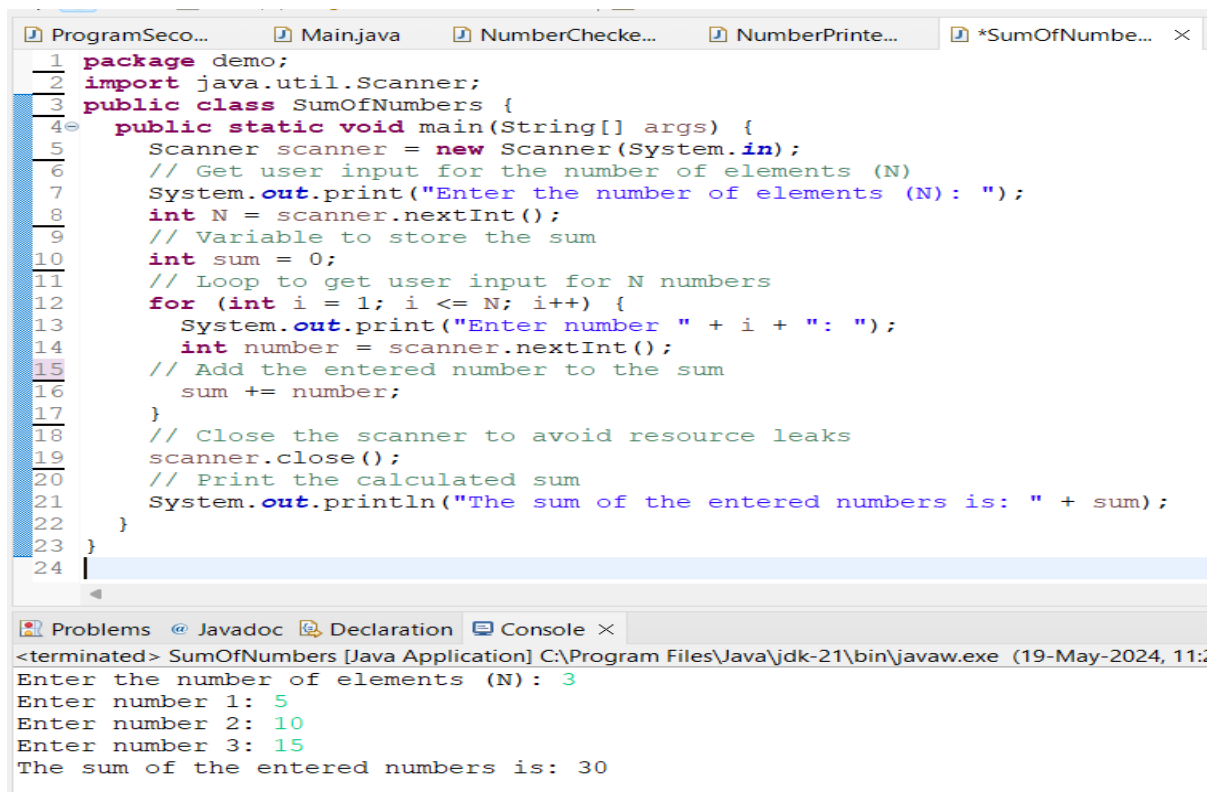
```
1
2
3
4
5
6
7
8
9
10
```

5. Write a program that takes an integer N as input and calculates the sum of entered numbers.

Program:

```
package demo;
import java.util.Scanner;
public class SumOfNumbers {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Get user input for the number of elements (N)
        System.out.print("Enter the number of elements (N): ");
        int N = scanner.nextInt();
        // Variable to store the sum
        int sum = 0;
        // Loop to get user input for N numbers
        for (int i = 1; i <= N; i++) {
            System.out.print("Enter number " + i + ": ");
            int number = scanner.nextInt();
            // Add the entered number to the sum
            sum += number;
        }
        // Close the scanner to avoid resource leaks
        scanner.close();
        // Print the calculated sum
        System.out.println("The sum of the entered numbers is: " + sum);
    }
}
```

Output:



The screenshot shows an IDE with a Java file named `*SumOfNumbe...` open. The code is identical to the one provided in the 'Program' section. The console output at the bottom shows the execution results:

```
<terminated> SumOfNumbers [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (19-May-2024, 11:
Enter the number of elements (N): 3
Enter number 1: 5
Enter number 2: 10
Enter number 3: 15
The sum of the entered numbers is: 30
```

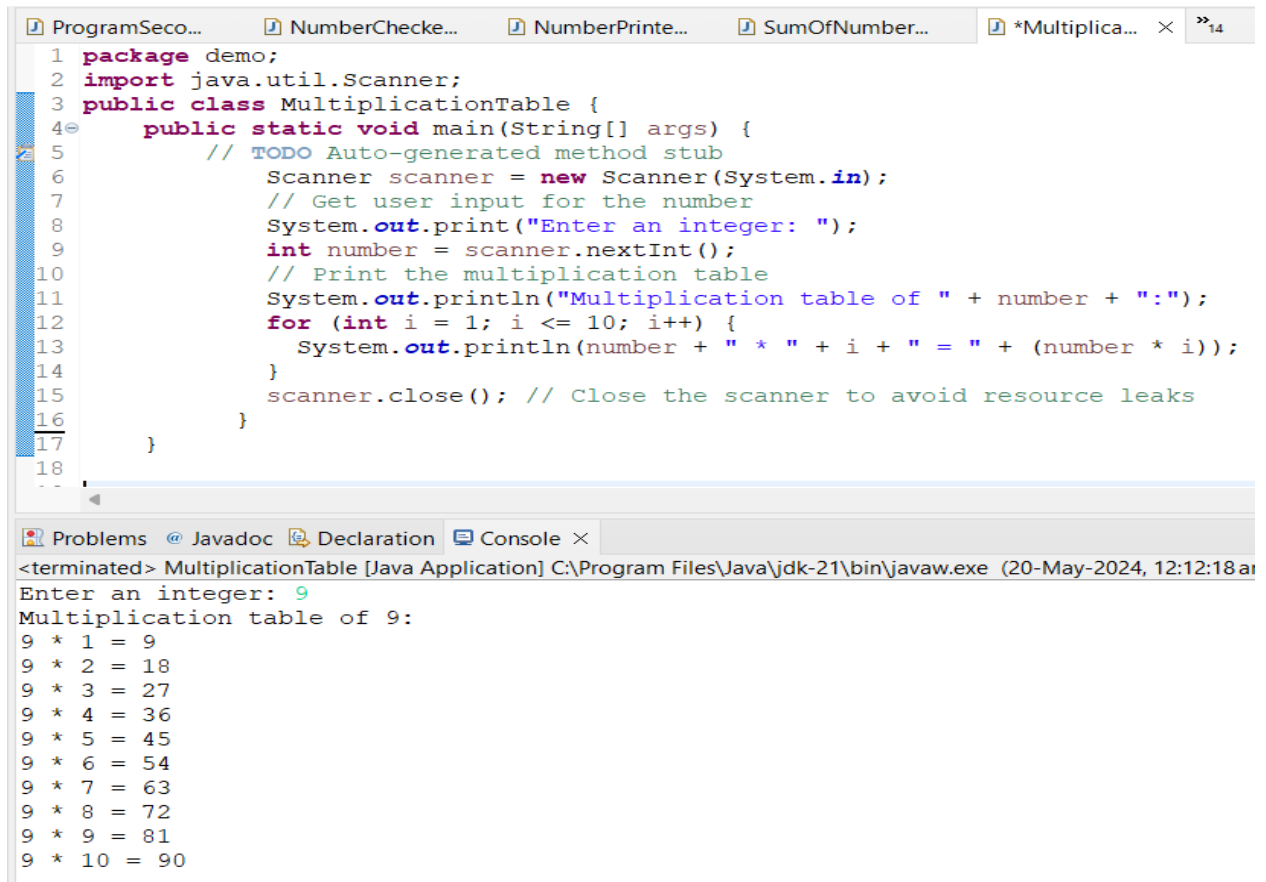
6. Write a program that takes an integer as input and prints its multiplication table up to 10.

Program:

```
package demo;
import java.util.Scanner;
public class MultiplicationTable {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner scanner = new Scanner(System.in);
        // Get user input for the number
        System.out.print("Enter an integer: ");
        int number = scanner.nextInt();
        // Print the multiplication table
        System.out.println("Multiplication table of " + number +
            ":");

        for (int i = 1; i <= 10; i++) {
            System.out.println(number + " * " + i + " = " + (number *
                i));
        }
        scanner.close(); // Close the scanner to avoid resource leaks
    }
}
```

Output:



```
1 package demo;
2 import java.util.Scanner;
3 public class MultiplicationTable {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner scanner = new Scanner(System.in);
7         // Get user input for the number
8         System.out.print("Enter an integer: ");
9         int number = scanner.nextInt();
10        // Print the multiplication table
11        System.out.println("Multiplication table of " + number + ":");
12        for (int i = 1; i <= 10; i++) {
13            System.out.println(number + " * " + i + " = " + (number * i));
14        }
15        scanner.close(); // Close the scanner to avoid resource leaks
16    }
17 }
18
```

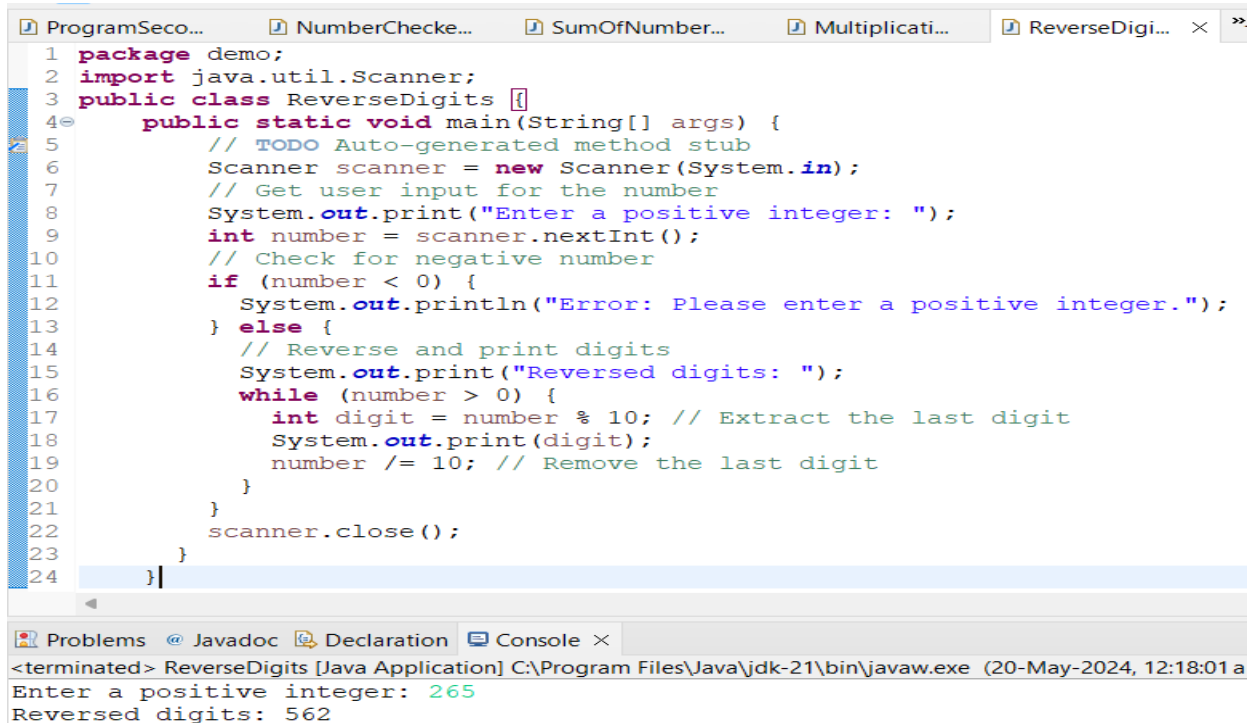
```
<terminated> MultiplicationTable [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (20-May-2024, 12:12:18 ar
Enter an integer: 9
Multiplication table of 9:
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
```


7. Write a program that takes a positive integer as input and prints its digits in reverse order.

Program:

```
package demo;
import java.util.Scanner;
public class ReverseDigits {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner scanner = new Scanner(System.in);
        // Get user input for the number
        System.out.print("Enter a positive integer: ");
        int number = scanner.nextInt();
        // Check for negative number
        if (number < 0) {
            System.out.println("Error: Please enter a positive integer.");
        } else {
            // Reverse and print digits
            System.out.print("Reversed digits: ");
            while (number > 0) {
                int digit = number % 10; // Extract the last digit
                System.out.print(digit);
                number /= 10; // Remove the last digit
            }
        }
        scanner.close();
    }
}
```

Output:



The screenshot shows an IDE with several open files: ProgramSeco..., NumberChecke..., SumOfNumber..., Multiplicati..., and ReverseDigi... The code for ReverseDigits.java is visible, matching the code provided in the previous block. The console output at the bottom shows the program execution:

```
<terminated> ReverseDigits [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (20-May-2024, 12:18:01 a
Enter a positive integer: 265
Reversed digits: 562
```

8. Create a class Animal with a method makeSound() that prints "Some generic animal sound". Create another class Dog that extends Animal and overrides the makeSound() method to print "Bark". Write a main method to demonstrate calling the makeSound() method on an Animal reference holding a Dog object.

Program:

```
package demo;
class Animal { //creating class animal
    public void makeSound() {
        System.out.println("Some generic animal sound");
    }
}

class Dog extends Animal { //Dog class extends Animal
    //Override function
    public void makeSound() {
        System.out.println("Bark");
    }
}

public class method {
    public static void main(String[] args) {
        Animal animal = new Dog();
        animal.makeSound(); // This will call the makeSound() method of the Dog
    }
}
```

Output:

```
Bark
```