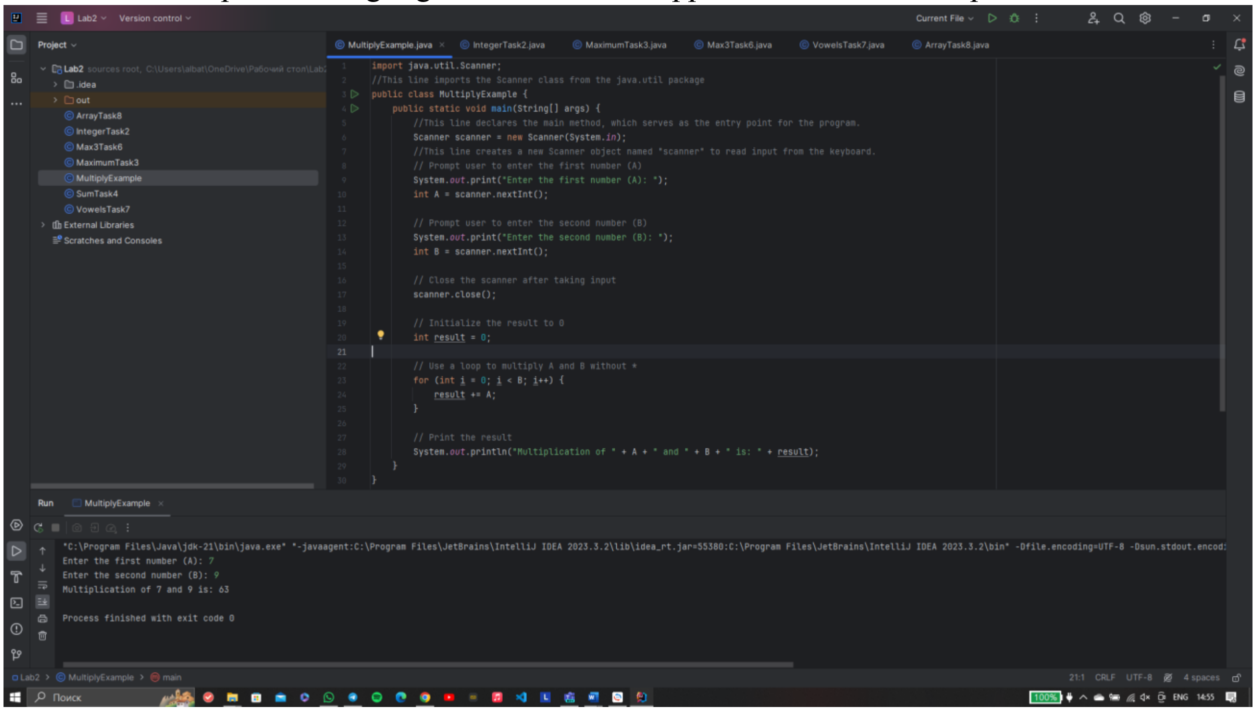
Task1) Explanation:

The provided Java code performs multiplication of two numbers without using the multiplication operator. It uses a Scanner to obtain input for two integers, initializes variables to store the numbers and the result, and prompts the user to input the first (A) and second (B) numbers. After closing the scanner, the program initializes the result to 0 and employs a for loop to simulate multiplication by repeatedly adding the value of A to the result for B iterations.

Finally, the program prints the result of the multiplication, incorporating the entered values for A and B into the output message.This code structure ensures user input, multiplication logic without using the multiplication operator, and clear output display of the result. The use of a loop to simulate multiplication highlights an alternative approach to arithmetic operations.



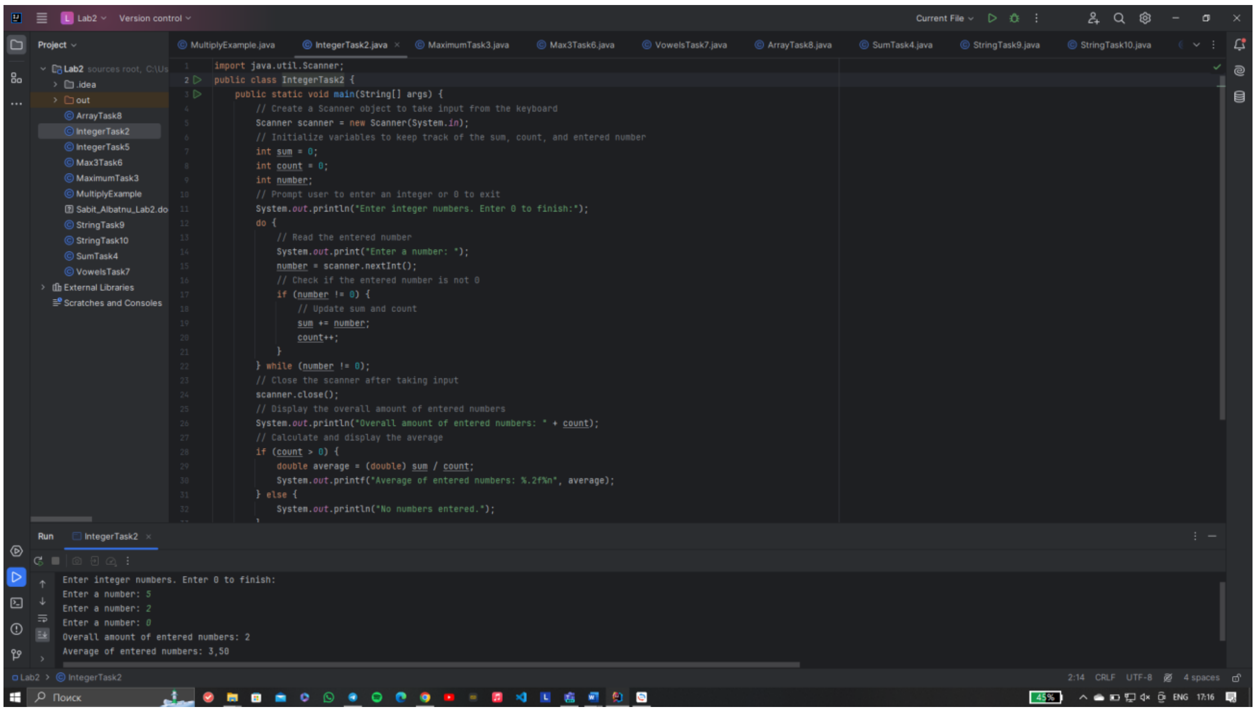
```java
import java.util.Scanner;
//This line imports the Scanner class from the java.util package
public class MultiplyExample {
    public static void main(String[] args) {
        //This line declares the main method, which serves as the entry point for the program.
        Scanner scanner = new Scanner(System.in);
        //This line creates a new Scanner object named "scanner" to read input from the keyboard.
        // Prompt user to enter the first number (A)
        System.out.print("Enter the first number (A): ");
        int A = scanner.nextInt();

        // Prompt user to enter the second number (B)
        System.out.print("Enter the second number (B): ");
        int B = scanner.nextInt();

        // Close the scanner after taking input
        scanner.close();

        // Initialize the result to 0
        int result = 0;

        // Use a loop to multiply A and B without *
        for (int i = 0; i < B; i++) {
            result += A;
        }

        // Print the result
        System.out.println("Multiplication of " + A + " and " + B + " is: " + result);
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=55380:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter the first number (A): 7
Enter the second number (B): 9
Multiplication of 7 and 9 is: 63

Process finished with exit code 0
```

Task 2) Explanation:

Code calculates the sum and average of a series of user-entered integer numbers until 0 is input. It uses a Scanner to take input, initializes variables to keep track of the sum, count, and the entered number, and prompts the user to input integers until 0 is entered. The do-while loop continuously reads and processes input, updating the sum and count if the entered number is not 0.

After the loop, the Scanner is closed, and the program displays the overall amount of entered numbers. It then calculates and displays the average if at least one number was entered. If no numbers were entered, it outputs a message indicating so.This code structure ensures continuous input processing and provides clear information about the overall amount of entered numbers and their average. The handling of the count variable helps avoid division by zero when calculating the average.



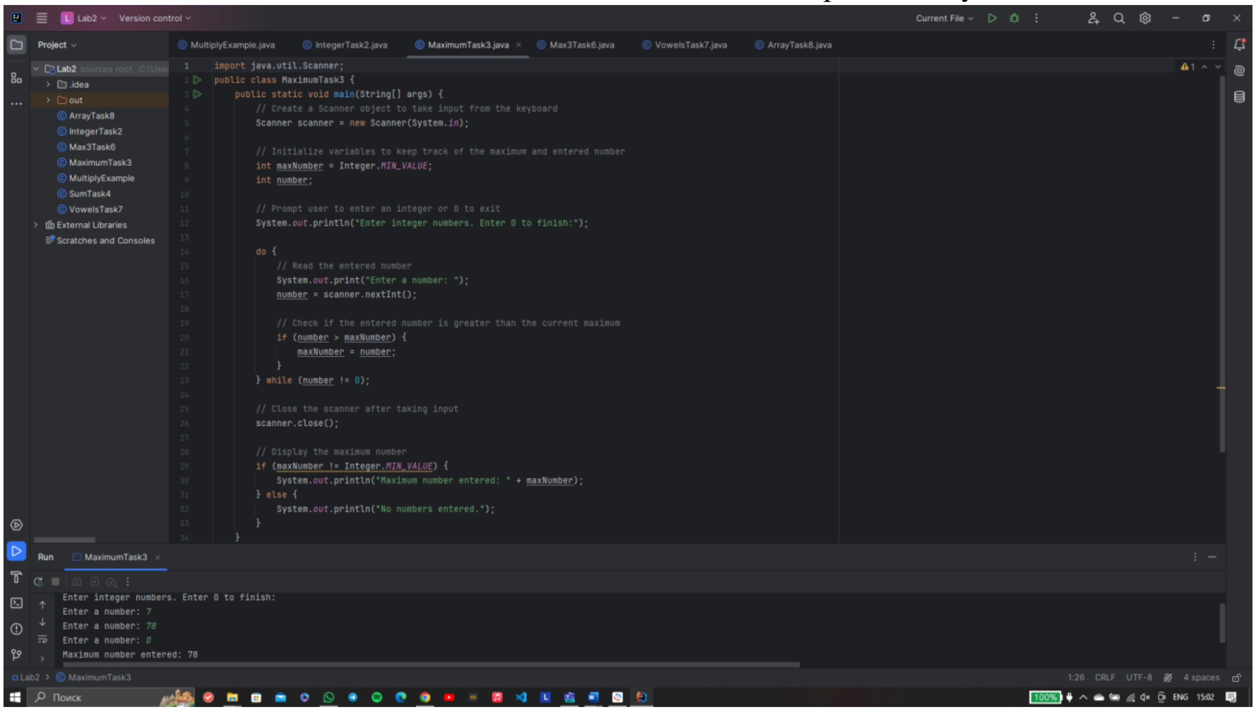
```java
import java.util.Scanner;
public class IntegerTask2 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Initialize variables to keep track of the sum, count, and entered number
        int sum = 0;
        int count = 0;
        int number;
        // Prompt user to enter an integer or 0 to exit
        System.out.println("Enter integer numbers. Enter 0 to finish:");
        do {
            // Read the entered number
            System.out.print("Enter a number: ");
            number = scanner.nextInt();
            // Check if the entered number is not 0
            if (number != 0) {
                // Update sum and count
                sum += number;
                count++;
            }
        } while (number != 0);
        // Close the scanner after taking input
        scanner.close();
        // Display the overall amount of entered numbers
        System.out.println("Overall amount of entered numbers: " + count);
        // Calculate and display the average
        if (count > 0) {
            double average = (double) sum / count;
            System.out.printf("Average of entered numbers: %.2f%n", average);
        } else {
            System.out.println("No numbers entered.");
```

```
Enter integer numbers. Enter 0 to finish:
Enter a number: 5
Enter a number: 2
Enter a number: 0
Overall amount of entered numbers: 2
Average of entered numbers: 3,50
```

Task 3) Explanation:

Task determines the maximum among a series of user-entered integer numbers until 0 is input. It uses a Scanner to take input, initializes variables to keep track of the maximum and the entered number, and prompts the user to input integers until 0 is entered. The do-while loop continuously reads and processes input, updating the maximum if the entered number is greater.

After the loop, the Scanner is closed, and the program displays the maximum number entered. If no numbers were entered (indicated by the initialized value of `maxNumber`), it outputs a message indicating so.This code structure ensures continuous input processing and accurately tracks the maximum number entered by the user. The handling of the initialized value of `maxNumber` accounts for the case when no valid numbers are provided by the user.



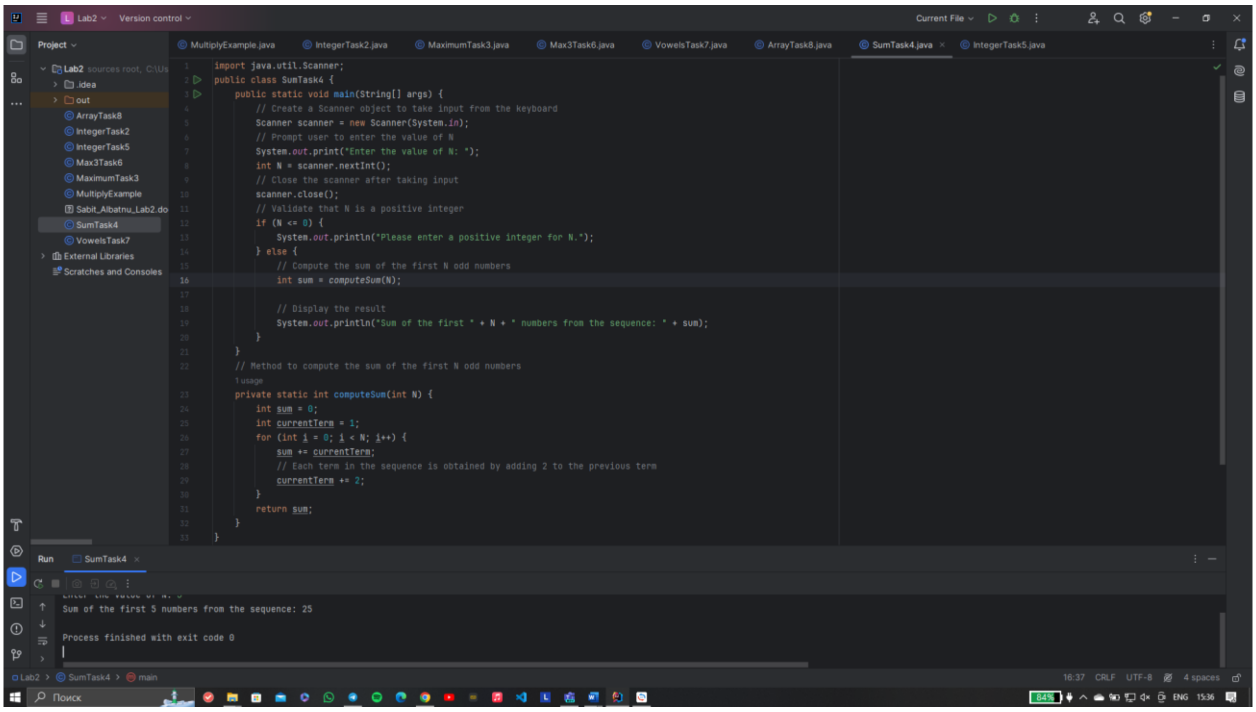
```java
import java.util.Scanner;
public class MaximumTask3 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);

        // Initialize variables to keep track of the maximum and entered number
        int maxNumber = Integer.MIN_VALUE;
        int number;

        // Prompt user to enter an integer or 0 to exit
        System.out.println("Enter integer numbers. Enter 0 to finish:");

        do {
            // Read the entered number
            System.out.print("Enter a number: ");
            number = scanner.nextInt();

            // Check if the entered number is greater than the current maximum
            if (number > maxNumber) {
                maxNumber = number;
            }
        } while (number != 0);

        // Close the scanner after taking input
        scanner.close();

        // Display the maximum number
        if (maxNumber != Integer.MIN_VALUE) {
            System.out.println("Maximum number entered: " + maxNumber);
        } else {
            System.out.println("No numbers entered.");
        }
    }
```

```
Enter integer numbers. Enter 0 to finish:
Enter a number: 7
Enter a number: 78
Enter a number: 0
Maximum number entered: 78
```

Task 4) (**homework**)Explanation:

The code calculates the sum of the first N odd numbers in a sequence. It begins by prompting the user to input a positive integer N, using a Scanner object for input. After closing the scanner, it checks if N is a positive integer; if not, it prompts the user to enter a positive integer. If N is positive, the program proceeds to calculate the sum of the first N odd numbers using a method called `computeSum`.

In the `computeSum` method, a variable `sum` is initialized to 0, and a variable `currentTerm` is set to 1. It then iterates N times, adding the current term to the sum and updating the current term by adding 2 in each iteration, simulating the generation of odd numbers. Finally, the program displays the computed sum for the user.The code structure ensures input validation and provides a clear method (`computeSum`) for the core logic of summing the first N odd numbers.



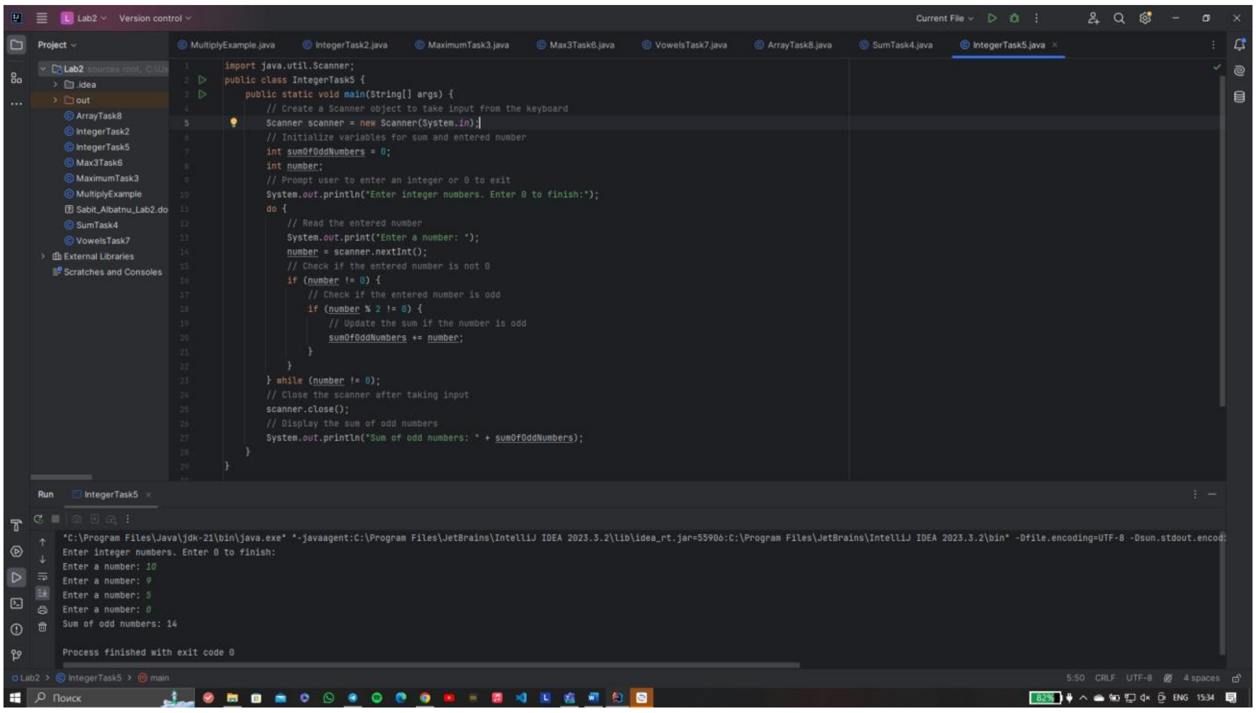
```java
import java.util.Scanner;
public class SumTask4 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Prompt user to enter the value of N
        System.out.print("Enter the value of N: ");
        int N = scanner.nextInt();
        // Close the scanner after taking input
        scanner.close();
        // Validate that N is a positive integer
        if (N <= 0) {
            System.out.println("Please enter a positive integer for N.");
        } else {
            // Compute the sum of the first N odd numbers
            int sum = computeSum(N);

            // Display the result
            System.out.println("Sum of the first " + N + " numbers from the sequence: " + sum);
        }
    }
    // Method to compute the sum of the first N odd numbers
    private static int computeSum(int N) {
        int sum = 0;
        int currentTerm = 1;
        for (int i = 0; i < N; i++) {
            sum += currentTerm;
            // Each term in the sequence is obtained by adding 2 to the previous term
            currentTerm += 2;
        }
        return sum;
    }
}
```

```
Sum of the first 5 numbers from the sequence: 25

Process finished with exit code 0
```

Task 5) (**homework**) Justification:

In this task, code calculates the sum of odd numbers entered by the user until 0 is input. It uses a Scanner to take input, initializes variables for the sum and the entered number, and prompts the user to input integers until 0 is entered. The do-while loop continuously reads and processes input, checking if the entered number is odd and updating the sum accordingly.

If the entered number is not 0 and is odd, it gets added to the `sumOfOddNumbers`. The loop continues until the user enters 0. After the loop, the Scanner is closed, and the program displays the sum of the entered odd numbers.This code ensures continuous input processing, updates the sum for odd numbers, and provides a clear user interface for entering integers and calculating their sum.



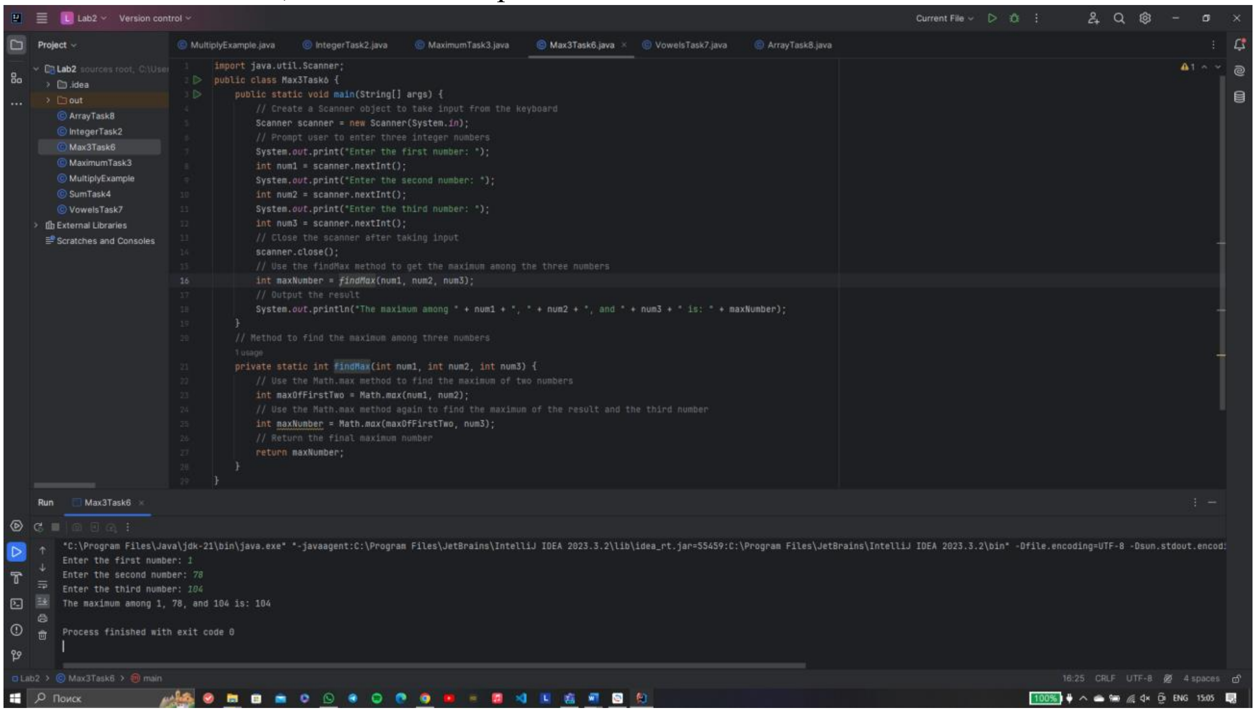
```java
import java.util.Scanner;
public class IntegerTask5 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Initialize variables for sum and entered number
        int sumOfOddNumbers = 0;
        int number;
        // Prompt user to enter an integer or 0 to exit
        System.out.println("Enter integer numbers. Enter 0 to finish:");
        do {
            // Read the entered number
            System.out.print("Enter a number: ");
            number = scanner.nextInt();
            // Check if the entered number is not 0
            if (number != 0) {
                // Check if the entered number is odd
                if (number % 2 != 0) {
                    // Update the sum if the number is odd
                    sumOfOddNumbers += number;
                }
            }
        } while (number != 0);
        // Close the scanner after taking input
        scanner.close();
        // Display the sum of odd numbers
        System.out.println("Sum of odd numbers: " + sumOfOddNumbers);
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=55986:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter integer numbers. Enter 0 to finish:
Enter a number: 10
Enter a number: 9
Enter a number: 5
Enter a number: 0
Sum of odd numbers: 14

Process finished with exit code 0
```

Task 6) Explanation:

It determines the maximum among three user-entered integer numbers. It uses a Scanner to obtain input for three integers, prompting the user accordingly. After closing the scanner, the program employs a method named `findMax` to calculate the maximum among the three input numbers.

The `findMax` method utilizes the `Math.max` function to find the maximum of two numbers. It first calculates the maximum of the first two input numbers and then uses the same method to find the maximum of that result and the third input number. The final maximum number is returned.The main method then outputs the result, displaying the maximum among the three input numbers. The code provides a clear structure for user input, method abstraction for maximum calculation, and concise output of the result.



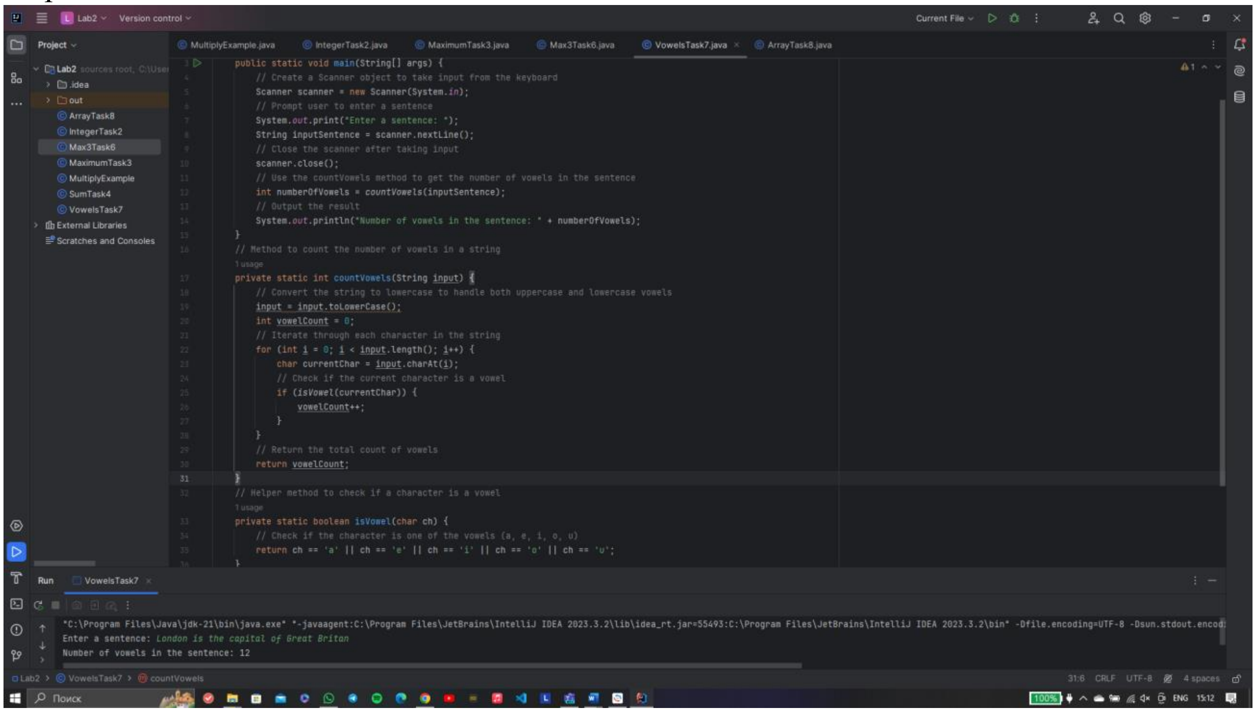
```java
import java.util.Scanner;
public class Max3Task6 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Prompt user to enter three integer numbers
        System.out.print("Enter the first number: ");
        int num1 = scanner.nextInt();
        System.out.print("Enter the second number: ");
        int num2 = scanner.nextInt();
        System.out.print("Enter the third number: ");
        int num3 = scanner.nextInt();
        // Close the scanner after taking input
        scanner.close();
        // Use the findMax method to get the maximum among the three numbers
        int maxNumber = findMax(num1, num2, num3);
        // Output the result
        System.out.println("The maximum among " + num1 + ", " + num2 + ", and " + num3 + " is: " + maxNumber);
    }
    // Method to find the maximum among three numbers
    private static int findMax(int num1, int num2, int num3) {
        // Use the Math.max method to find the maximum of two numbers
        int maxOfFirstTwo = Math.max(num1, num2);
        // Use the Math.max method again to find the maximum of the result and the third number
        int maxNumber = Math.max(maxOfFirstTwo, num3);
        // Return the final maximum number
        return maxNumber;
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=55459:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter the first number: 1
Enter the second number: 78
Enter the third number: 104
The maximum among 1, 78, and 104 is: 104

Process finished with exit code 0
```

Task 7) Explanation:

The code determines the number of vowels in a user-entered sentence. It utilizes a Scanner to obtain input for the sentence, prompting the user accordingly. After closing the scanner, the program employs a method named `countVowels` to calculate the number of vowels in the input sentence.

The `countVowels` method first converts the input sentence to lowercase to handle both uppercase and lowercase vowels consistently. It then iterates through each character in the string, utilizing a helper method `isVowel` to check if the current character is a vowel (a, e, i, o, or u). The count of vowels is incremented accordingly, and the total count is returned.The main method then outputs the result, displaying the number of vowels in the entered sentence. This code structure ensures a clear separation of concerns, with distinct methods for input, processing, and output.



```java
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Prompt user to enter a sentence
        System.out.print("Enter a sentence: ");
        String inputSentence = scanner.nextLine();
        // Close the scanner after taking input
        scanner.close();
        // Use the countVowels method to get the number of vowels in the sentence
        int numberOfVowels = countVowels(inputSentence);
        // Output the result
        System.out.println("Number of vowels in the sentence: " + numberOfVowels);
    }
    // Method to count the number of vowels in a string
    private static int countVowels(String input) {
        // Convert the string to lowercase to handle both uppercase and lowercase vowels
        input = input.toLowerCase();
        int vowelCount = 0;
        // Iterate through each character in the string
        for (int i = 0; i < input.length(); i++) {
            char currentChar = input.charAt(i);
            // Check if the current character is a vowel
            if (isVowel(currentChar)) {
                vowelCount++;
            }
        }
        // Return the total count of vowels
        return vowelCount;
    }
    // Helper method to check if a character is a vowel
    private static boolean isVowel(char ch) {
        // Check if the character is one of the vowels (a, e, i, o, u)
        return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';
    }
```

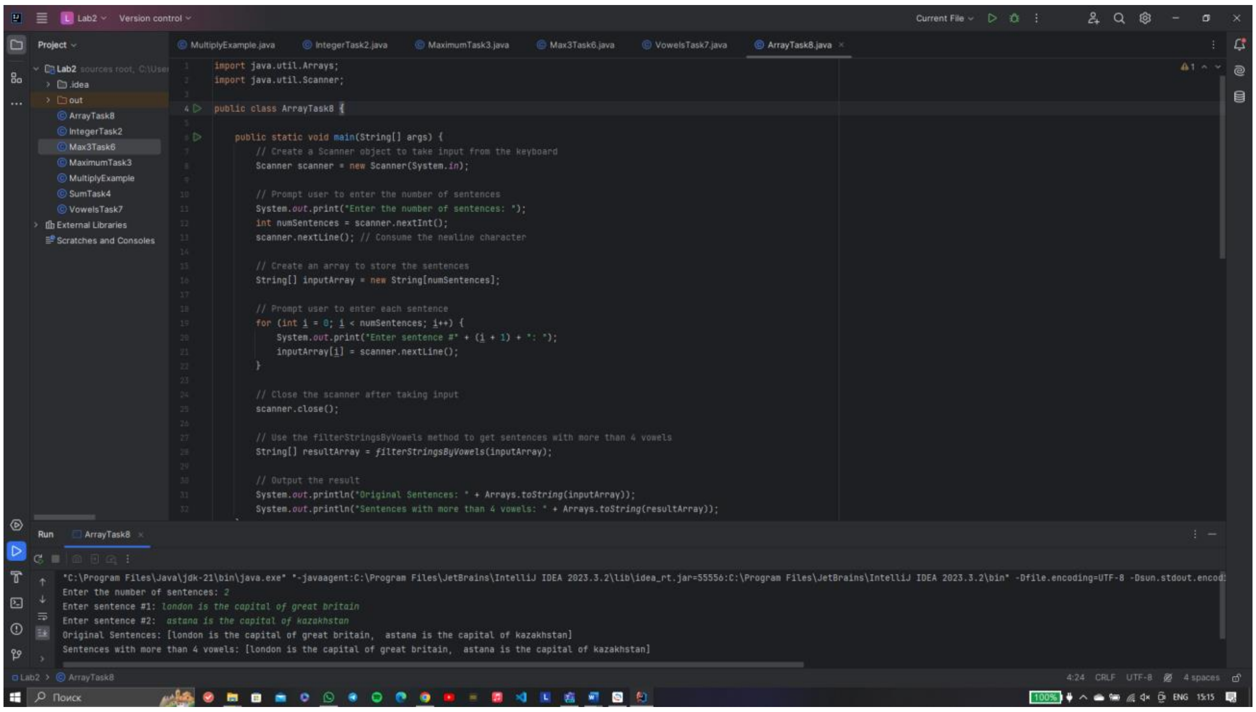
```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=55493:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter a sentence: London is the capital of Great Britan
Number of vowels in the sentence: 12
```

Task 8) Justification:

Task processes an array of sentences entered by the user. It uses a Scanner to obtain the number of sentences and each sentence itself. After closing the scanner, it creates an array to store the sentences, prompts the user to input each sentence, and then uses a method named `filterStringsByVowels` to extract sentences with more than 4 vowels.

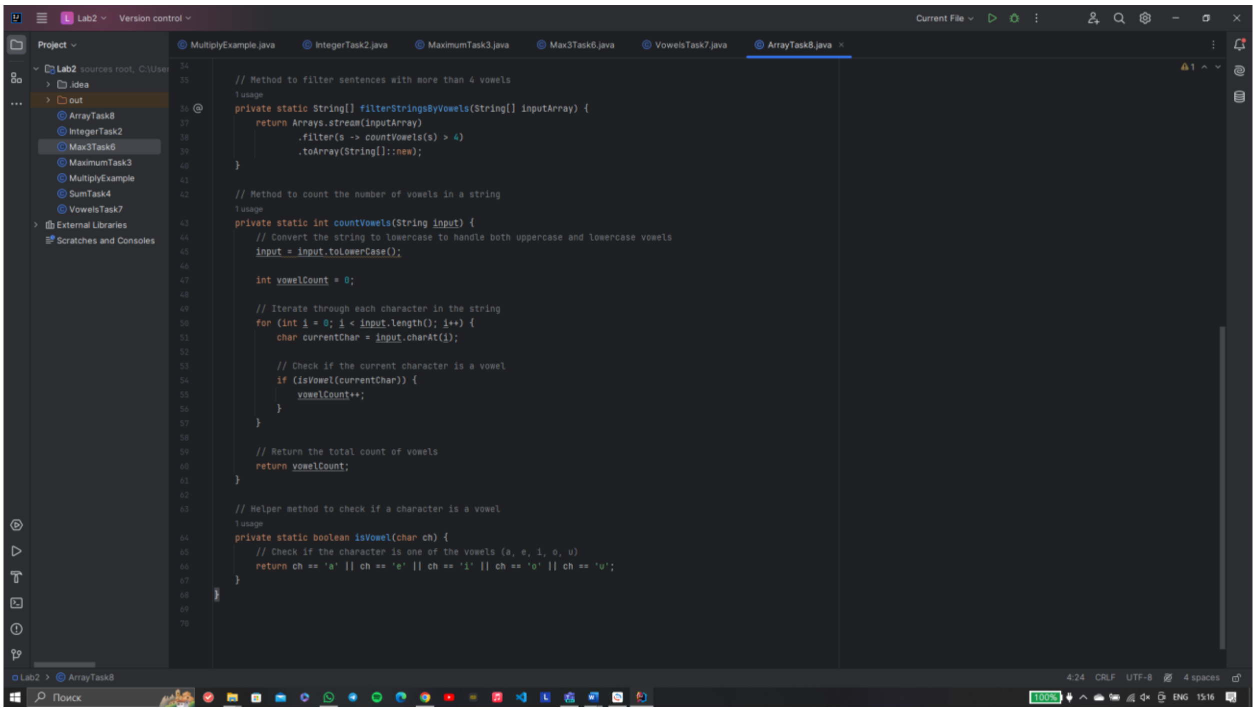
The `filterStringsByVowels` method utilizes the Java Stream API to filter the input array based on the count of vowels in each sentence. The result array is then displayed, showing both the original sentences and the ones with more than 4 vowels.

The code structure ensures modularization with separate methods for input, processing, and output. It leverages the Stream API for efficient filtering and utilizes helper methods to handle vowel counting and checking.



```java
import java.util.Arrays;
import java.util.Scanner;

public class ArrayTask8 {

    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);

        // Prompt user to enter the number of sentences
        System.out.print("Enter the number of sentences: ");
        int numSentences = scanner.nextInt();
        scanner.nextLine(); // Consume the newline character

        // Create an array to store the sentences
        String[] inputArray = new String[numSentences];

        // Prompt user to enter each sentence
        for (int i = 0; i < numSentences; i++) {
            System.out.print("Enter sentence #" + (i + 1) + ": ");
            inputArray[i] = scanner.nextLine();
        }

        // Close the scanner after taking input
        scanner.close();

        // Use the filterStringsByVowels method to get sentences with more than 4 vowels
        String[] resultArray = filterStringsByVowels(inputArray);

        // Output the result
        System.out.println("Original Sentences: " + Arrays.toString(inputArray));
        System.out.println("Sentences with more than 4 vowels: " + Arrays.toString(resultArray));
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=55556:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter the number of sentences: 2
Enter sentence #1: london is the capital of great britain
Enter sentence #2:  astana is the capital of kazakhstan
Original Sentences: [london is the capital of great britain,  astana is the capital of kazakhstan]
Sentences with more than 4 vowels: [london is the capital of great britain,  astana is the capital of kazakhstan]
```

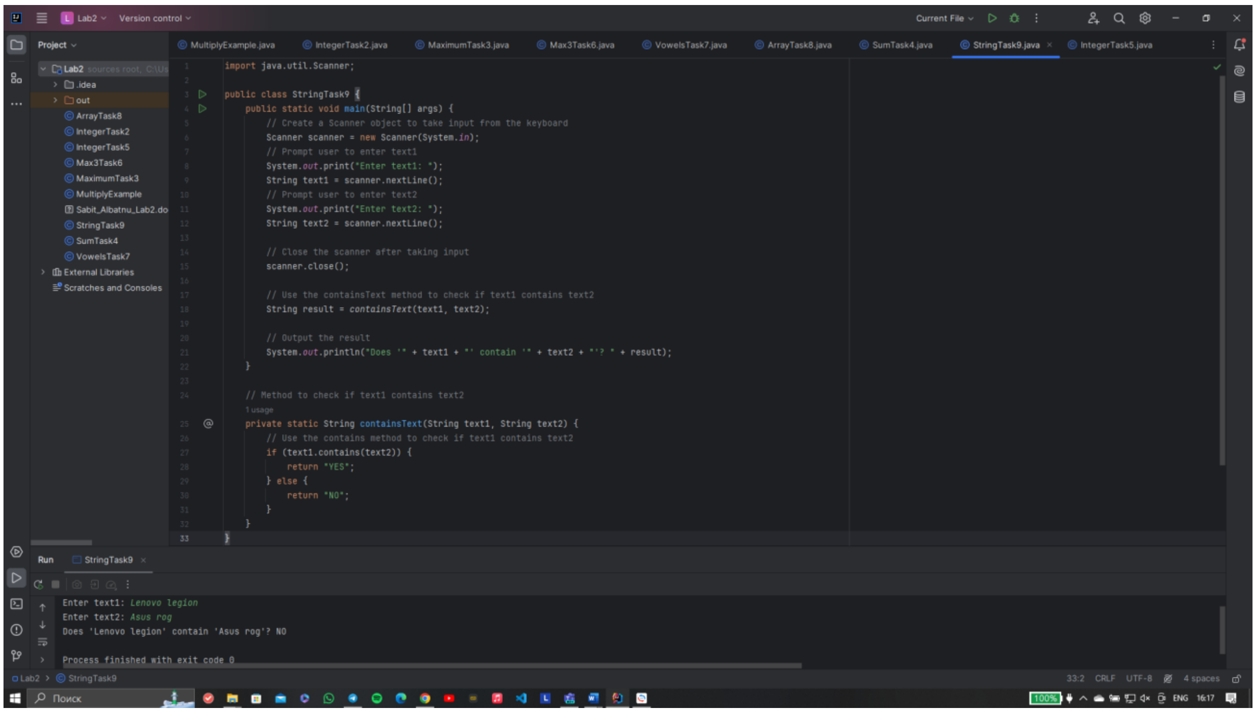


```java
    // Method to filter sentences with more than 4 vowels
    1 usage
    private static String[] filterStringsByVowels(String[] inputArray) {
        return Arrays.stream(inputArray)
                .filter(s -> countVowels(s) > 4)
                .toArray(String[]::new);
    }

    // Method to count the number of vowels in a string
    1 usage
    private static int countVowels(String input) {
        // Convert the string to lowercase to handle both uppercase and lowercase vowels
        input = input.toLowerCase();

        int vowelCount = 0;

        // Iterate through each character in the string
        for (int i = 0; i < input.length(); i++) {
            char currentChar = input.charAt(i);

            // Check if the current character is a vowel
            if (isVowel(currentChar)) {
                vowelCount++;
            }
        }

        // Return the total count of vowels
        return vowelCount;
    }

    // Helper method to check if a character is a vowel
    1 usage
    private static boolean isVowel(char ch) {
        // Check if the character is one of the vowels (a, e, i, o, u)
        return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';
    }
}
```

Task 9) (**homework**)Explanation:

The provided Java code checks whether the first entered text (`text1`) contains the second entered text (`text2`). It utilizes a Scanner to obtain input for both texts, prompts the user accordingly, and then uses a method named `containsText` to perform the containment check.

The `containsText` method uses the `contains` method of the String class to check if `text1` contains `text2`. The result is then displayed, indicating whether `text1` contains `text2` with a "YES" or "NO" response.This code structure ensures a clear separation of concerns, with distinct methods for input, processing, and output. The `containsText` method abstracts the containment check, making the main method concise and easy to understand.



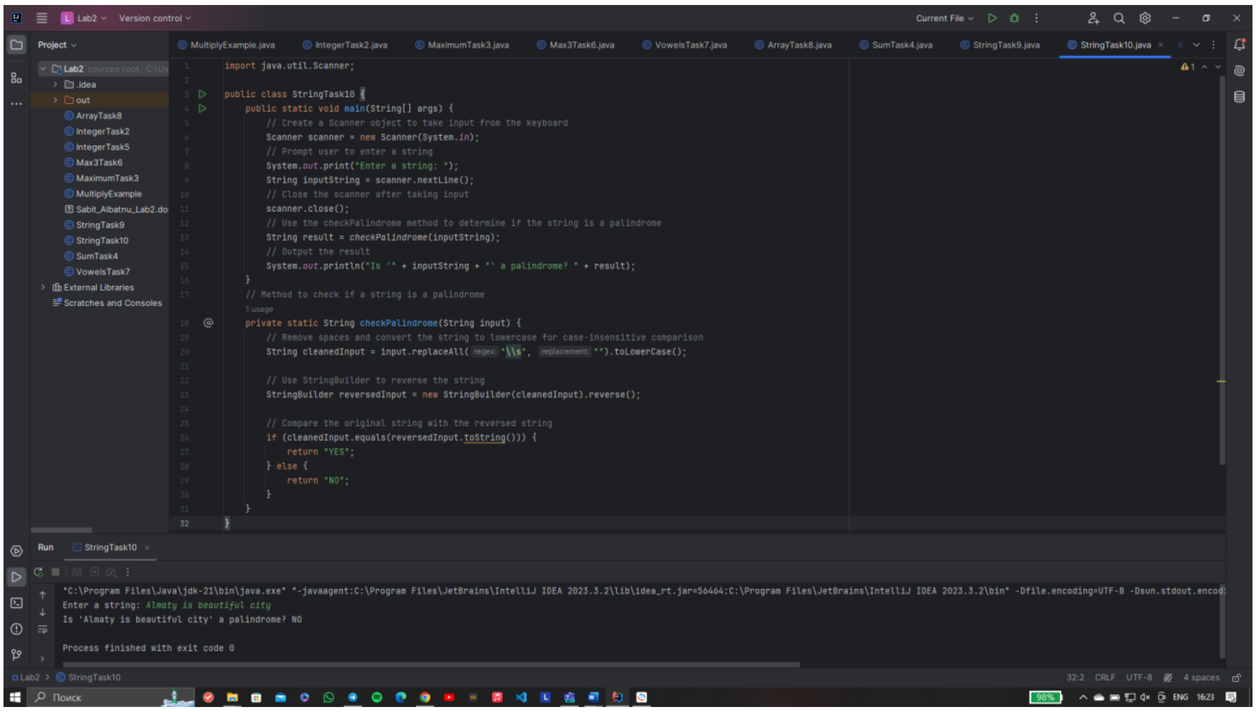
```java
import java.util.Scanner;

public class StringTask9 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Prompt user to enter text1
        System.out.print("Enter text1: ");
        String text1 = scanner.nextLine();
        // Prompt user to enter text2
        System.out.print("Enter text2: ");
        String text2 = scanner.nextLine();

        // Close the scanner after taking input
        scanner.close();

        // Use the containsText method to check if text1 contains text2
        String result = containsText(text1, text2);

        // Output the result
        System.out.println("Does '" + text1 + "' contain '" + text2 + "'? " + result);
    }

    // Method to check if text1 contains text2
    private static String containsText(String text1, String text2) {
        // Use the contains method to check if text1 contains text2
        if (text1.contains(text2)) {
            return "YES";
        } else {
            return "NO";
        }
    }
}
```

```
Enter text1: Lenovo legion
Enter text2: Asus rog
Does 'Lenovo legion' contain 'Asus rog'? NO

Process finished with exit code 0
```

Task 10) (**homework**)Justification:

Code checks whether the entered string is a palindrome. It uses a Scanner to obtain input for the string, prompts the user accordingly, and then uses a method named `checkPalindrome` to perform the palindrome check.

The `checkPalindrome` method first removes spaces and converts the input string to lowercase to ensure a case-insensitive comparison. It then uses a `StringBuilder` to reverse the string and compares the original string with the reversed one. The result is displayed, indicating whether the entered string is a palindrome with a "YES" or "NO" response.This code structure ensures a clear separation of concerns, with distinct methods for input, processing, and output. The `checkPalindrome` method abstracts the palindrome check logic, making the main method concise and easy to understand. The code efficiently handles case-insensitivity and space removal for accurate palindrome detection.



```java
import java.util.Scanner;

public class StringTask10 {
    public static void main(String[] args) {
        // Create a Scanner object to take input from the keyboard
        Scanner scanner = new Scanner(System.in);
        // Prompt user to enter a string
        System.out.print("Enter a string: ");
        String inputString = scanner.nextLine();
        // Close the scanner after taking input
        scanner.close();
        // Use the checkPalindrome method to determine if the string is a palindrome
        String result = checkPalindrome(inputString);
        // Output the result
        System.out.println("Is '" + inputString + "' a palindrome? " + result);
    }
    // Method to check if a string is a palindrome
    private static String checkPalindrome(String input) {
        // Remove spaces and convert the string to lowercase for case-insensitive comparison
        String cleanedInput = input.replaceAll("\\s", "").toLowerCase();

        // Use StringBuilder to reverse the string
        StringBuilder reversedInput = new StringBuilder(cleanedInput).reverse();

        // Compare the original string with the reversed string
        if (cleanedInput.equals(reversedInput.toString())) {
            return "YES";
        } else {
            return "NO";
        }
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=56464:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter a string: Almaty is beautiful city
Is 'Almaty is beautiful city' a palindrome? NO

Process finished with exit code 0
```