Laboratory work 3

Task 1)

1.Import Scanner Class: The program imports the Scanner class from the java.util package, allowing it to read input from the user.

2.Main Method: The main method serves as the entry point for the program.

3.Variable Initialization: The program initializes variables for the hourly rate (`rate`) and the calculated salary (`salary`). The `double` data type is used for these variables to accommodate decimal values. The rate is set to 0.2, representing the base hourly rate.

4.Input Prompt: The program prompts the user to enter the number of work hours.

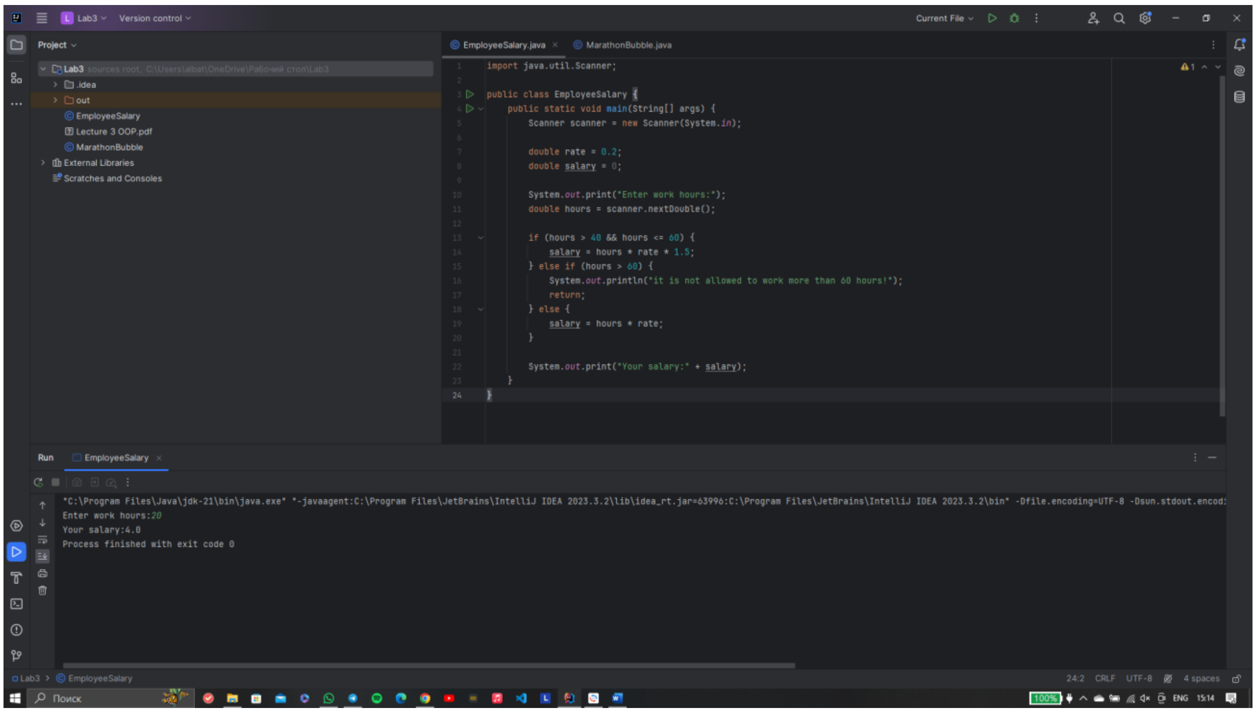
5.Read Input: The input provided by the user (number of work hours) is stored in the `hours` variable using the Scanner object.

6.Salary Calculation:

- If the number of hours is greater than 40 and less than or equal to 60, the program calculates the salary for overtime hours. It multiplies the number of hours by the rate and then by 1.5 to apply a 50% overtime bonus.
- If the number of hours exceeds 60, the program prints a message stating that it's not allowed to work more than 60 hours and terminates.
- If the number of hours is less than or equal to 40, the program calculates the regular salary by multiplying the hours by the rate.

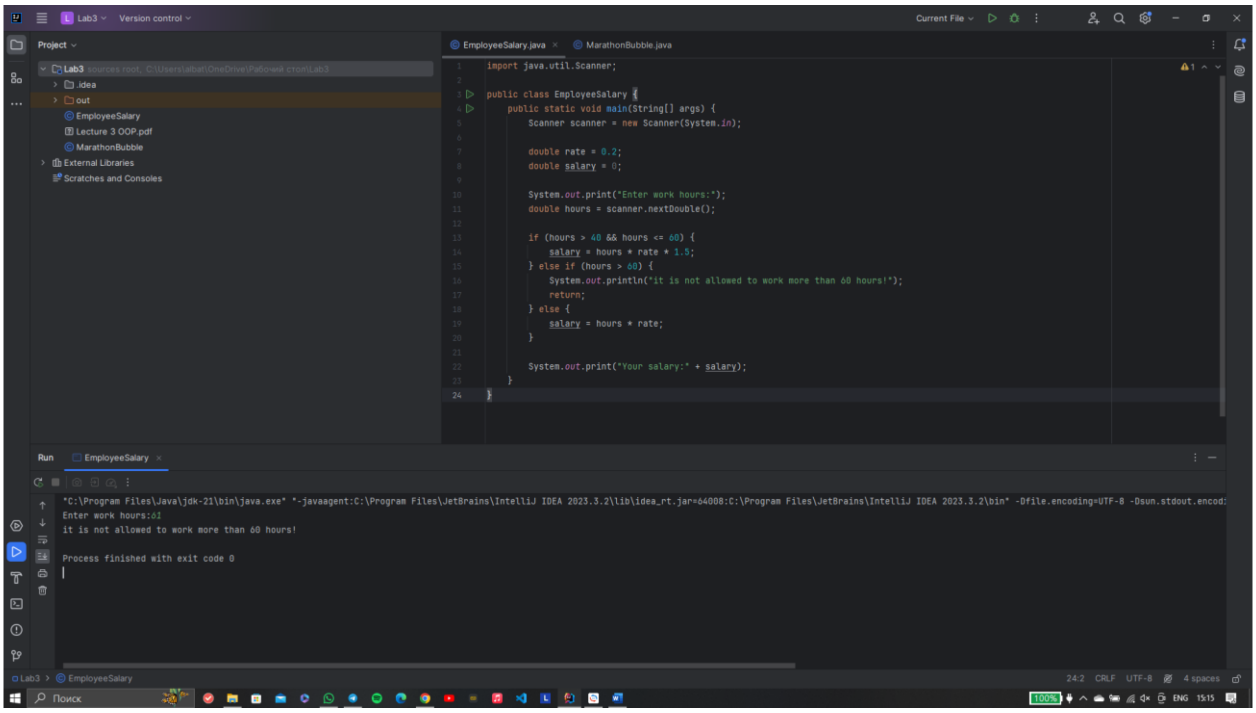
7.Output Display: The code prints the calculated salary to the user.

This code efficiently handles different scenarios related to work hours, such as regular hours, overtime, and exceeding the maximum allowable hours. It provides clear feedback on the salary calculation and adheres to the specified rules regarding work hours and overtime. The use of the `double` data type allows for precise calculation of salaries involving fractional hours or rates. Additionally, the `if`, `else if`, and `else` statements control the flow of execution based on the conditions specified.



```java
import java.util.Scanner;

public class EmployeeSalary {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double rate = 0.2;
        double salary = 0;

        System.out.print("Enter work hours:");
        double hours = scanner.nextDouble();

        if (hours > 40 && hours <= 60) {
            salary = hours * rate * 1.5;
        } else if (hours > 60) {
            System.out.println("it is not allowed to work more than 60 hours!");
            return;
        } else {
            salary = hours * rate;
        }

        System.out.print("Your salary:" + salary);
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=63996:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter work hours:20
Your salary:4.0
Process finished with exit code 0
```



```java
import java.util.Scanner;

public class EmployeeSalary {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double rate = 0.2;
        double salary = 0;

        System.out.print("Enter work hours:");
        double hours = scanner.nextDouble();

        if (hours > 40 && hours <= 60) {
            salary = hours * rate * 1.5;
        } else if (hours > 60) {
            System.out.println("it is not allowed to work more than 60 hours!");
            return;
        } else {
            salary = hours * rate;
        }

        System.out.print("Your salary:" + salary);
    }
}
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\lib\idea_rt.jar=64008:C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encod
Enter work hours:61
it is not allowed to work more than 60 hours!

Process finished with exit code 0
```

Task 2)

1.Main Method:

- The main method initializes two arrays: `names`, which holds the names of marathon participants, and `times`, which stores their corresponding finishing times in minutes.

- It then calls the `bubbleSort` method to sort the arrays `names` and `times` based on the participants' finishing times.

2.Bubble Sort Algorithm (`bubbleSort` method):

- The `bubbleSort` method implements the Bubble Sort algorithm to sort the arrays `names` and `times` simultaneously.

- It iterates over the array `times` using nested loops, comparing adjacent elements.

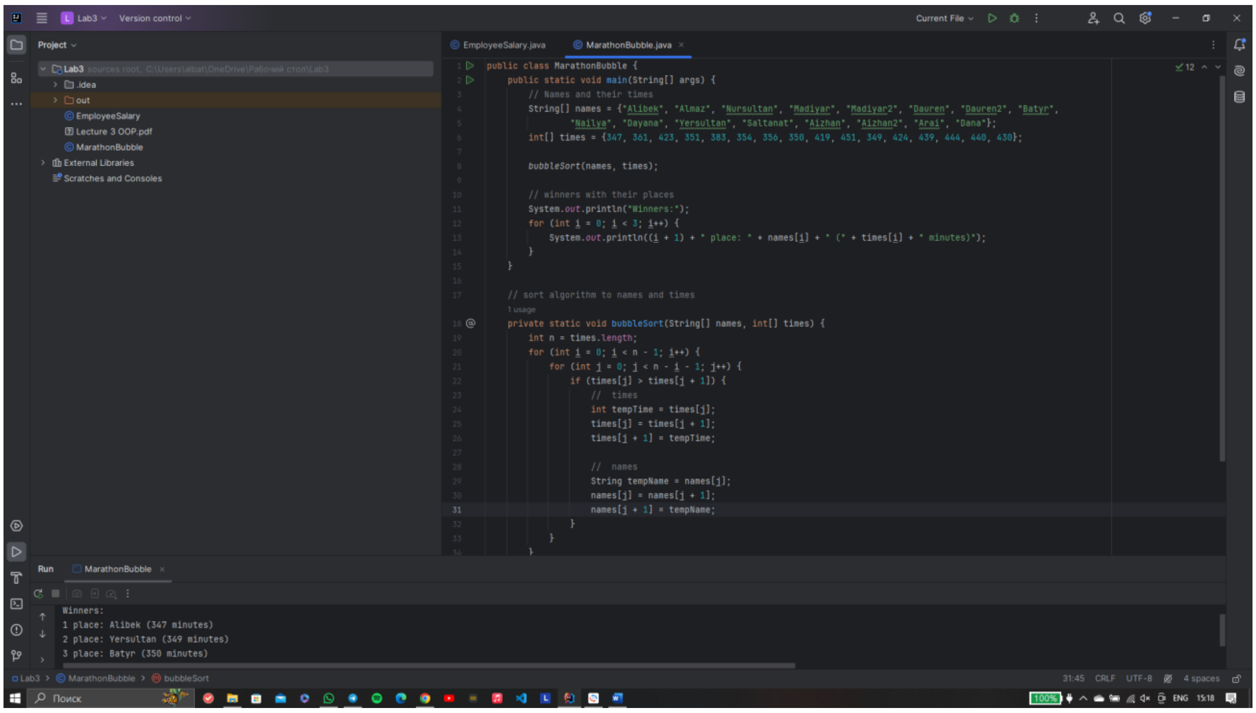
- If the finishing time of a participant (`times[j]`) is greater than the finishing time of the next participant (`times[j + 1]`), their positions are swapped along with their corresponding names in the `names` array.

- This process continues until the array is fully sorted.

3.Output Display:

- After sorting, the program prints the top three winners along with their respective finishing places and times.

- It iterates over the sorted `names` and `times` arrays, displaying the name, finishing time, and place (1st, 2nd, or 3rd) for each of the top three participants.



```java
public class MarathonBubble {
    public static void main(String[] args) {
        // Names and their times
        String[] names = {"Alibek", "Almaz", "Nursultan", "Madiyar", "Madiyar2", "Dauren", "Dauren2", "Batyr",
                "Nailya", "Dayana", "Yersultan", "Saltanat", "Aizhan", "Aizhan2", "Arai", "Dana"};
        int[] times = {347, 361, 423, 351, 383, 354, 356, 350, 419, 451, 349, 424, 439, 444, 440, 430};

        bubbleSort(names, times);

        // winners with their places
        System.out.println("Winners:");
        for (int i = 0; i < 3; i++) {
            System.out.println((i + 1) + " place: " + names[i] + " (" + times[i] + " minutes)");
        }
    }

    // sort algorithm to names and times
    private static void bubbleSort(String[] names, int[] times) {
        int n = times.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (times[j] > times[j + 1]) {
                    //  times
                    int tempTime = times[j];
                    times[j] = times[j + 1];
                    times[j + 1] = tempTime;

                    //  names
                    String tempName = names[j];
                    names[j] = names[j + 1];
                    names[j + 1] = tempName;
                }
            }
        }
```

```
Winners:
1 place: Alibek (347 minutes)
2 place: Yersultan (349 minutes)
3 place: Batyr (350 minutes)
```