

Домашнее Задание 6

Акильбеков Алар

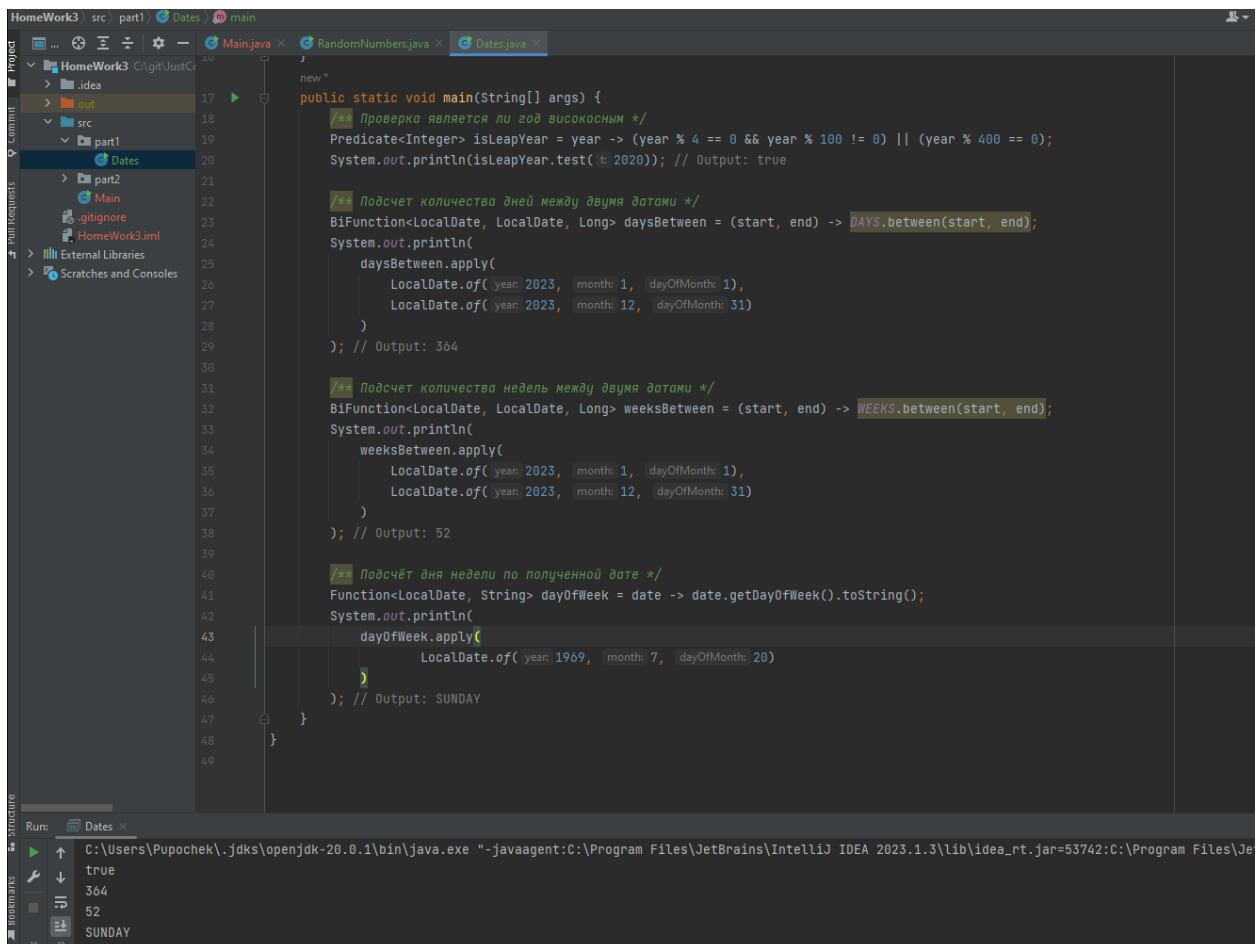
Github: (https://github.com/Alar-q/JustCode_HomeWorks)

Part 1

Задание 1

Создайте и вызовите следующие лямбда-выражения:

- Проверка является ли год високосным;
- Подсчет количества дней между двумя датами;
- Подсчёт количества полных недель между двумя датами;
- Подсчёт дня недели по полученной дате. Например, 20 июля 1969 года — воскресенье.



```
new "
public static void main(String[] args) {
    /** Проверка является ли год високосным */
    Predicate<Integer> isLeapYear = year -> (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    System.out.println(isLeapYear.test(2020)); // Output: true

    /** Подсчет количества дней между двумя датами */
    BiFunction<LocalDate, LocalDate, Long> daysBetween = (start, end) -> DAYS.between(start, end);
    System.out.println(
        daysBetween.apply(
            LocalDate.of(year: 2023, month: 1, dayOfMonth: 1),
            LocalDate.of(year: 2023, month: 12, dayOfMonth: 31)
        )
    ); // Output: 364

    /** Подсчет количества недель между двумя датами */
    BiFunction<LocalDate, LocalDate, Long> weeksBetween = (start, end) -> WEEKS.between(start, end);
    System.out.println(
        weeksBetween.apply(
            LocalDate.of(year: 2023, month: 1, dayOfMonth: 1),
            LocalDate.of(year: 2023, month: 12, dayOfMonth: 31)
        )
    ); // Output: 52

    /** Подсчёт дня недели по полученной дате */
    Function<LocalDate, String> dayOfWeek = date -> date.getDayOfWeek().toString();
    System.out.println(
        dayOfWeek.apply(
            LocalDate.of(year: 1969, month: 7, dayOfMonth: 20)
        )
    ); // Output: SUNDAY
}
```

Run: Dates

C:\Users\Pupochek\jdk\openjdk-20.0.1\bin\java.exe --javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.3\lib\idea_rt.jar=53742:C:\Program Files\Je

true
364
52
SUNDAY

Задание 2

Создайте и вызовите следующие лямбда-выражения:

- Сумма двух дробей;
- Разница двух дробей;
- Произведение двух дробей;
- Деление двух дробей.

The screenshot shows the IntelliJ IDEA IDE with a project named 'HomeWork3'. The 'src' directory contains a subdirectory 'part1' which includes files 'Dates.java', 'Fractions.java', and 'Main.java'. The 'Fractions.java' file is open in the editor, showing the following code:

```
1 package part1;
2
3 import java.util.function.BiFunction;
4
5 new *
6 public class Fractions {
7     new *
8     public static void main(String[] args) {
9         /** Сумма двух дробей */
10        BiFunction<Double, Double, Double> add = (a, b) -> a + b;
11        System.out.println(add.apply(1.2, 3.4)); // Output: 4.6
12
13        /** Разница двух дробей */
14        BiFunction<Double, Double, Double> subtract = (a, b) -> a - b;
15        System.out.println(subtract.apply(1.2, 3.4)); // Output: -2.2
16
17        /** Произведение двух дробей */
18        BiFunction<Double, Double, Double> multiply = (a, b) -> a * b;
19        System.out.println(multiply.apply(1.2, 3.4)); // Output: 4.08
20
21        /** Деление двух дробей */
22        BiFunction<Double, Double, Double> divide = (a, b) -> a / b;
23        System.out.println(divide.apply(1.2, 3.4)); // Output: 0.35294117647058826
24    }
25 }
```

The 'Run' tab at the bottom shows the execution output:

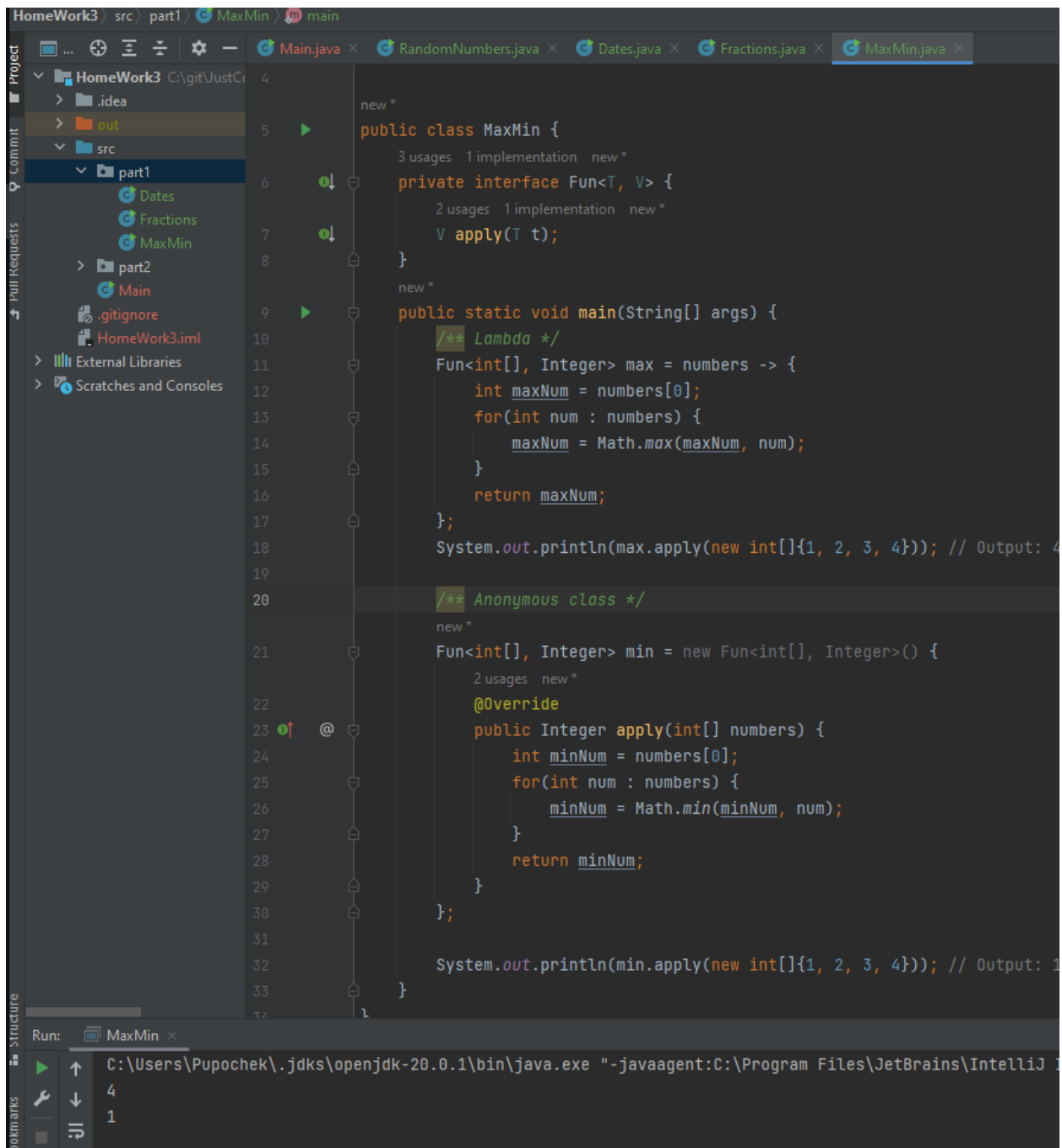
```
Run: Fractions
C:\Users\Pupochek\.jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.3\
4.6
-2.2
4.08
0.35294117647058826
```

Задание 3

Создайте и вызовите следующие лямбда-выражения.

Обязательно использовать шаблоны:

- Максимум из четырёх;
- Минимум из четырёх.



```
HomeWork3 > src > part1 > MaxMin > main
new *
public class MaxMin {
    3 usages 1 implementation new *
    private interface Fun<T, V> {
        2 usages 1 implementation new *
        V apply(T t);
    }
    new *
    public static void main(String[] args) {
        /** Lambda */
        Fun<int[], Integer> max = numbers -> {
            int maxNum = numbers[0];
            for(int num : numbers) {
                maxNum = Math.max(maxNum, num);
            }
            return maxNum;
        };
        System.out.println(max.apply(new int[]{1, 2, 3, 4})); // Output: 4

        /** Anonymous class */
        new *
        Fun<int[], Integer> min = new Fun<int[], Integer>() {
            2 usages new *
            @Override
            public Integer apply(int[] numbers) {
                int minNum = numbers[0];
                for(int num : numbers) {
                    minNum = Math.min(minNum, num);
                }
                return minNum;
            }
        };
        System.out.println(min.apply(new int[]{1, 2, 3, 4})); // Output: 1
    }
}
```

Run: MaxMin

C:\Users\Pupocek\.jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ" 4 1

Задание 4

Создайте и вызовите следующие лямбда-выражения.

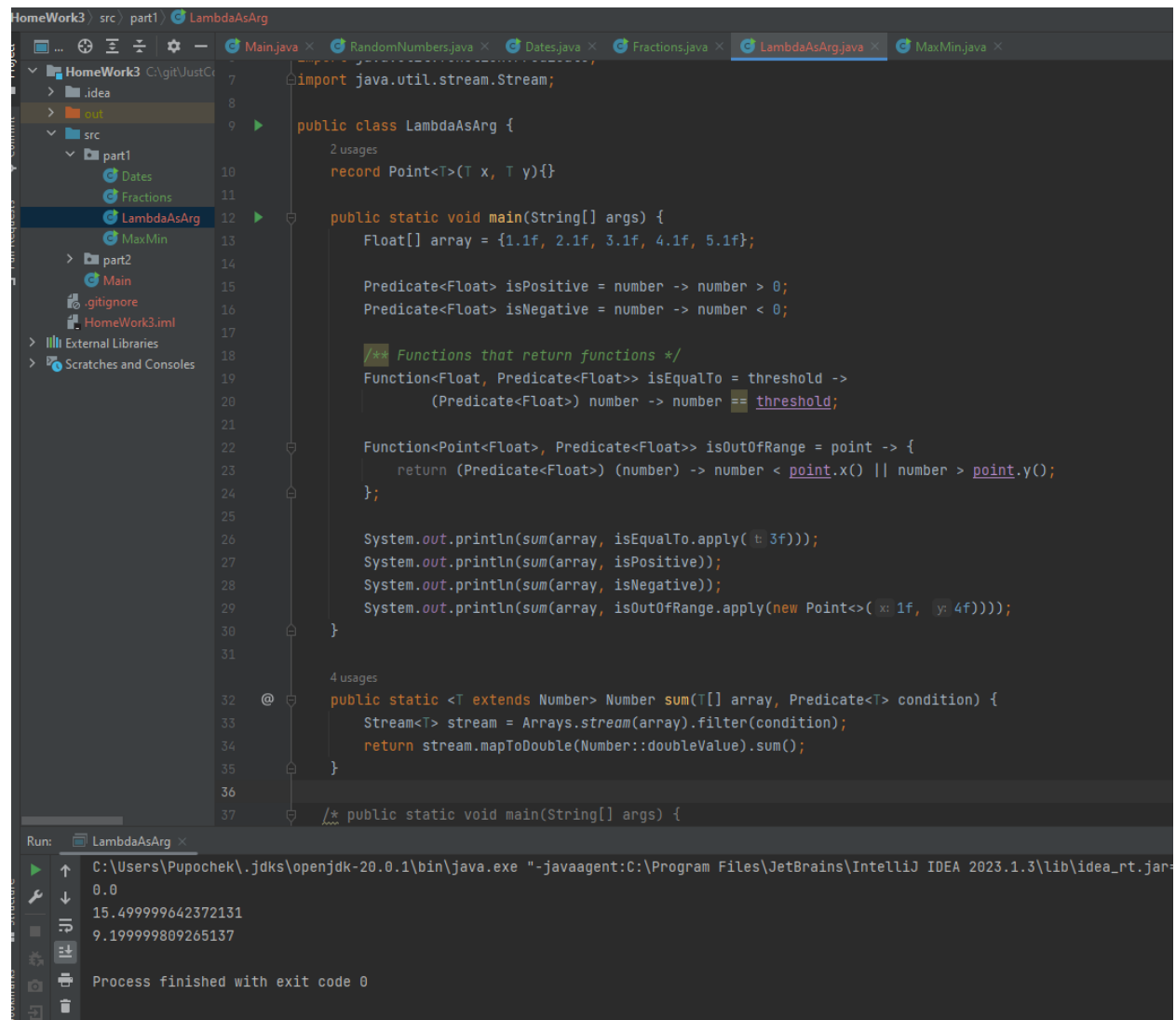
Обязательно использовать лямбду, как параметр метода.

Метод находит сумму элементов массива, которые соответствуют условию лямбда-выражения.

Варианты лямбда-выражений:

- Проверка на равенство конкретному числу;
- Число не находится в диапазоне от А до В;
- Проверка на положительность числа;
- Проверка на отрицательность числа

Изначально написал слишком сложный, но рабочий вариант с лямбдами, возвращающими лямбды и обобщенным статическим методом. Есть ошибки двоичной арифметики, которые свойственны числам с плавающей точкой.



```
import java.util.stream.Stream;

public class LambdaAsArg {
    record Point<T> (T x, T y){}

    public static void main(String[] args) {
        Float[] array = {1.1f, 2.1f, 3.1f, 4.1f, 5.1f};

        Predicate<Float> isPositive = number -> number > 0;
        Predicate<Float> isNegative = number -> number < 0;

        /** Functions that return functions */
        Function<Float, Predicate<Float>> isEqualTo = threshold ->
            (Predicate<Float>) number -> number == threshold;

        Function<Point<Float>, Predicate<Float>> isOutOfRange = point -> {
            return (Predicate<Float>) (number -> number < point.x() || number > point.y());
        };

        System.out.println(sum(array, isEqualTo.apply(3.3f)));
        System.out.println(sum(array, isPositive));
        System.out.println(sum(array, isNegative));
        System.out.println(sum(array, isOutOfRange.apply(new Point<> (x: 1f, y: 4f))));
    }

    @ 4 usages
    public static <T extends Number> Number sum(T[] array, Predicate<T> condition) {
        Stream<T> stream = Arrays.stream(array).filter(condition);
        return stream.mapToDouble(Number::doubleValue).sum();
    }

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

Run: LambdaAsArg x

```
C:\Users\Pupochek\.jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.3\lib\idea_rt.jar:
0.0
15.499999642372131
9.199999809265137
Process finished with exit code 0
```

Простой и рабочий вариант.

```
30 }
31
32 public static <T extends Number> Number sum(T[] array, Predicate<T> condition) {
33     Stream<T> stream = Arrays.stream(array).filter(condition);
34     return stream.mapToDouble(Number::doubleValue).sum();
35 }*/
36
37 public static void main(String[] args) {
38     Integer[] array = {1, 2, 3, 4, 5};
39
40     Predicate<Integer> isEqualToThree = number -> number == 3;
41     Predicate<Integer> isPositive = number -> number > 0;
42     Predicate<Integer> isOutOfRange = number -> number < 2 || number > 4;
43
44     System.out.println(sum(array, isEqualToThree)); // Output: 3
45     System.out.println(sum(array, isPositive)); // Output: 15
46     System.out.println(sum(array, isOutOfRange)); // Output: 6
47 }
48
49 public static int sum(Integer[] array, Predicate<Integer> condition) {
50     return Arrays.stream(array).filter(condition)
51         .mapToInt(Integer::valueOf)
52         .sum();
53 }
54
55 }
56
```

Run: LambdaAsArg

C:\Users\Pupochek\.jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\lib\idea_rt.jar=6093:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\bin" -Didea.config.path=C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\config -Didea.copyright.path=C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\copyright -Didea.home.path=C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\bin -Didea.platform.prefix=Java -Didea.vendor.id=JetBrains -Didea.version=2023.1 -jar C:\Program Files\JetBrains\IntelliJ IDEA 2023.1\bin\idea.jar

3
15
6

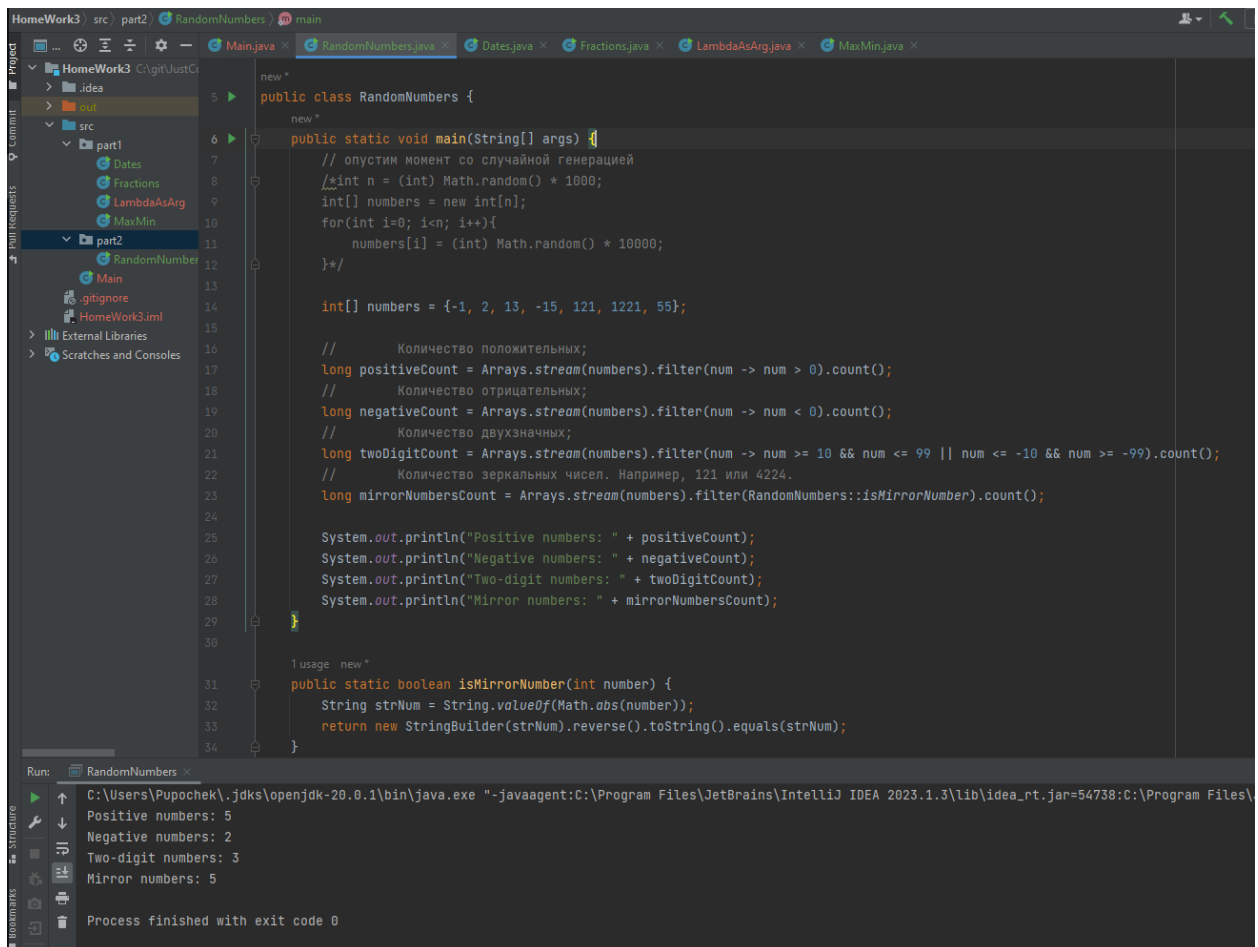
Process finished with exit code 0

Part 2

Задание 1

Для набора случайно сгенерированных целых чисел нужно:

- Количество положительных;
- Количество отрицательных;
- Количество двухзначных;
- Количество зеркальных чисел. Например, 121 или 4224.



```
public class RandomNumbers {  
    public static void main(String[] args) {  
        // опустим момент со случайной генерацией  
        /*int n = (int) Math.random() * 1000;  
        int[] numbers = new int[n];  
        for(int i=0; i<n; i++){  
            numbers[i] = (int) Math.random() * 10000;  
        }*/  
  
        int[] numbers = {-1, 2, 13, -15, 121, 1221, 55};  
  
        // Количество положительных;  
        long positiveCount = Arrays.stream(numbers).filter(num -> num > 0).count();  
        // Количество отрицательных;  
        long negativeCount = Arrays.stream(numbers).filter(num -> num < 0).count();  
        // Количество двухзначных;  
        long twoDigitCount = Arrays.stream(numbers).filter(num -> num >= 10 && num <= 99 || num <= -10 && num >= -99).count();  
        // Количество зеркальных чисел. Например, 121 или 4224.  
        long mirrorNumbersCount = Arrays.stream(numbers).filter(RandomNumbers::isMirrorNumber).count();  
  
        System.out.println("Positive numbers: " + positiveCount);  
        System.out.println("Negative numbers: " + negativeCount);  
        System.out.println("Two-digit numbers: " + twoDigitCount);  
        System.out.println("Mirror numbers: " + mirrorNumbersCount);  
    }  
  
    public static boolean isMirrorNumber(int number) {  
        String strNum = String.valueOf(Math.abs(number));  
        return new StringBuilder(strNum).reverse().toString().equals(strNum);  
    }  
}
```

Run: RandomNumbers ×

C:\Users\Pupohek\jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.3\lib\idea_rt.jar=54738:C:\Program Files\

Positive numbers: 5
Negative numbers: 2
Two-digit numbers: 3
Mirror numbers: 5

Process finished with exit code 0

Задание 2

Для набора названий продуктов (продукты могут повторяться) нужно:

- Показать все продукты;
- Показать все продукты с названием меньше пяти символов;
- Посчитать сколько раз встречается продукт, чье название ввёл пользователь;
- Показать все продукты, которые начинаются на заданную букву;
- Показать все продукты из категории «Молоко».

The screenshot shows the IntelliJ IDEA IDE with the `Products.java` file open. The code defines a `main` method that creates a list of products. The `Run` console shows the output of the program, which includes the list of products and the results of the five tasks specified in the assignment.

```
1 usage: new *
47 void main(){
48     List<Product> productList = new ArrayList<>(Arrays.asList(
49         new Product(Title.Milk, Category.MilkProducts),
50         new Product(Title.Milk, Category.MilkProducts),
51         new Product(Title.Cheese, Category.MilkProducts),
52         new Product(Title.Butter, Category.MilkProducts),
53         new Product(Title.Pasta, Category.FlourProducts),
54         new Product(Title.Bread, Category.FlourProducts),
```

Run: Products ×

C:\Users\Pupochek\.jdk\openjdk-20.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ I

Показать все продукты

MilkProducts: Milk (id:66375)

MilkProducts: Milk (id:33951)

MilkProducts: Cheese (id:94899)

MilkProducts: Butter (id:7010)

FlourProducts: Pasta (id:66950)

FlourProducts: Bread (id:21210)

FlourProducts: Cake (id:27668)

Показать все продукты с названием меньше пяти символов

MilkProducts: Milk (id:66375)

MilkProducts: Milk (id:33951)

FlourProducts: Cake (id:27668)

Посчитать сколько раз встречается продукт, чье название ввёл пользователь

Milk appears 2 times.

Показать все продукты, которые начинаются на заданную букву

MilkProducts: Milk (id:66375)

MilkProducts: Milk (id:33951)

Показать все продукты из категории «Молоко»

MilkProducts: Milk (id:66375)

MilkProducts: Milk (id:33951)

MilkProducts: Cheese (id:94899)

MilkProducts: Butter (id:7010)

Process finished with exit code 0

```

Scanner scanner = new Scanner(System.in);
while(true){
    System.out.println("\nПосчитать сколько раз встречается продукт, чье название ввёл пользователь");
    String name = scanner.next();
    count = productList.stream()
        .filter(product -> product.getTitle().toString().equalsIgnoreCase(name))
        .count();
    System.out.println(userInput + " appears " + count + " times.");
}

```

Посчитать сколько раз встречается продукт, чье название ввёл пользователь

bread

bread appears 1 times.

Посчитать сколько раз встречается продукт, чье название ввёл пользователь

milk

milk appears 2 times.

Посчитать сколько раз встречается продукт, чье название ввёл пользователь

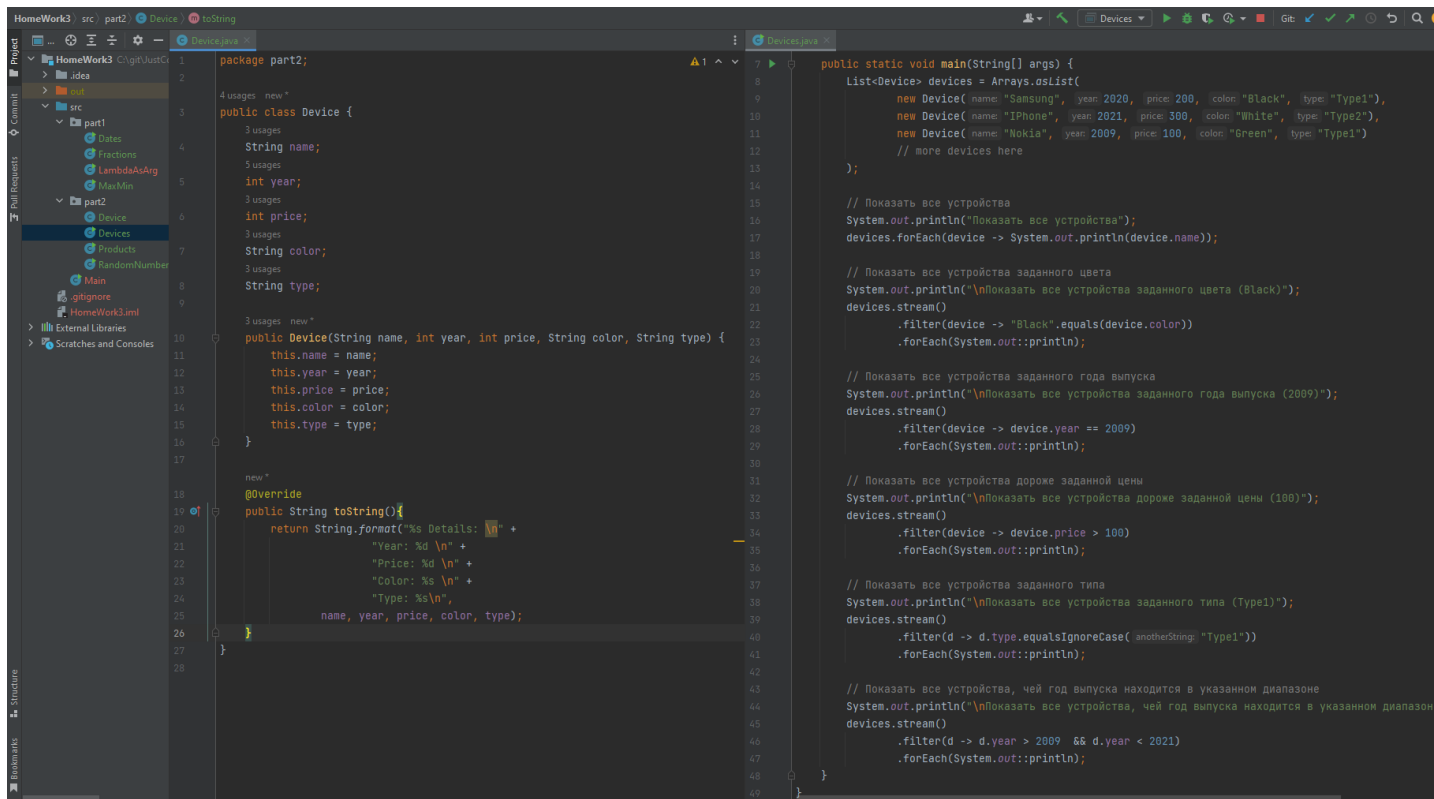
cake

cake appears 1 times.

Задание 3

Создайте класс «Устройство». Он должен хранить информацию о названии устройства, год выпуска, цена, цвет, тип устройства. Нужно создать набор устройств и выполнить следующие задачи:

- Показать все устройства;
- Показать все устройства заданного цвета;
- Показать все устройства заданного года выпуска;
- Показать все устройства дороже заданной цены;
- Показать все устройства заданного типа;
- Показать все устройства, чей год выпуска находится в указанном диапазоне.



```
package part2;

4 usages new
public class Device {
    3 usages new
    String name;
    4 usages new
    int year;
    5 usages new
    int price;
    6 usages new
    String color;
    7 usages new
    String type;

    8 usages new
    public Device(String name, int year, int price, String color, String type) {
        9 usages new
        this.name = name;
        10 usages new
        this.year = year;
        11 usages new
        this.price = price;
        12 usages new
        this.color = color;
        13 usages new
        this.type = type;
    }

    14 usages new
    @Override
    public String toString() {
        15 usages new
        return String.format("%s Details: %n" +
            "Year: %d \n" +
            "Price: %d \n" +
            "Color: %s \n" +
            "Type: %s\n",
            name, year, price, color, type);
    }
}

new
@Override
public static void main(String[] args) {
    List<Device> devices = Arrays.asList(
        new Device(name: "Samsung", year: 2020, price: 200, color: "Black", type: "Type1"),
        new Device(name: "IPhone", year: 2021, price: 300, color: "White", type: "Type2"),
        new Device(name: "Nokia", year: 2009, price: 100, color: "Green", type: "Type1")
        // more devices here
    );

    // Показать все устройства
    System.out.println("Показать все устройства");
    devices.forEach(device -> System.out.println(device.name));

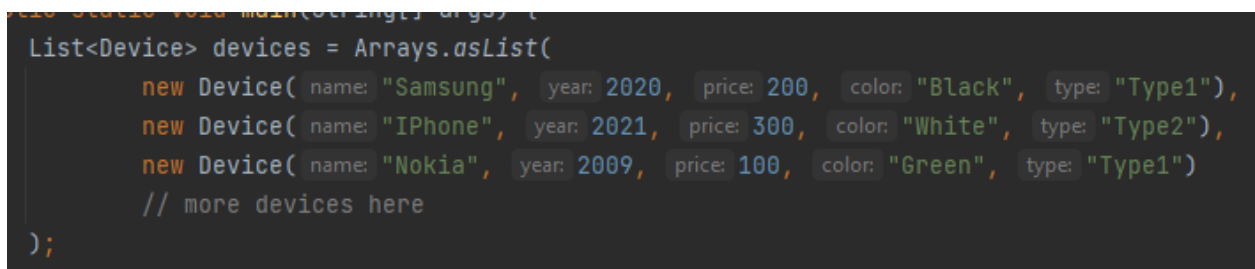
    // Показать все устройства заданного цвета
    System.out.println("\nПоказать все устройства заданного цвета (Black)");
    devices.stream()
        .filter(device -> "Black".equals(device.color))
        .forEach(System.out::println);

    // Показать все устройства заданного года выпуска
    System.out.println("\nПоказать все устройства заданного года выпуска (2009)");
    devices.stream()
        .filter(device -> device.year == 2009)
        .forEach(System.out::println);

    // Показать все устройства дороже заданной цены
    System.out.println("\nПоказать все устройства дороже заданной цены (100)");
    devices.stream()
        .filter(device -> device.price > 100)
        .forEach(System.out::println);

    // Показать все устройства заданного типа
    System.out.println("\nПоказать все устройства заданного типа (Type1)");
    devices.stream()
        .filter(d -> d.type.equalsIgnoreCase("Type1"))
        .forEach(System.out::println);

    // Показать все устройства, чей год выпуска находится в указанном диапазоне
    System.out.println("\nПоказать все устройства, чей год выпуска находится в указанном диапазоне");
    devices.stream()
        .filter(d -> d.year > 2009 && d.year < 2021)
        .forEach(System.out::println);
}
```



```
public static void main(String[] args) {
    List<Device> devices = Arrays.asList(
        new Device(name: "Samsung", year: 2020, price: 200, color: "Black", type: "Type1"),
        new Device(name: "IPhone", year: 2021, price: 300, color: "White", type: "Type2"),
        new Device(name: "Nokia", year: 2009, price: 100, color: "Green", type: "Type1")
        // more devices here
    );
}
```

Показать все устройства

Samsung

iPhone

Nokia

Показать все устройства заданного цвета (Black)

Samsung Details:

Year: 2020

Price: 200

Color: Black

Type: Type1

Показать все устройства заданного года выпуска (2009)

Nokia Details:

Year: 2009

Price: 100

Color: Green

Type: Type1

Показать все устройства дороже заданной цены (100)

Samsung Details:

Year: 2020

Price: 200

Color: Black

Type: Type1

iPhone Details:

Year: 2021

Price: 300

Color: White

Type: Type2

Показать все устройства заданного типа (Type1)

Samsung Details:

Year: 2020

Price: 200

Color: Black

Type: Type1

Nokia Details:

Year: 2009

Price: 100

Color: Green

Type: Type1

Показать все устройства, чей год выпуска находится в указанном диапазоне (2009-2021)

Samsung Details:

Year: 2020

Price: 200

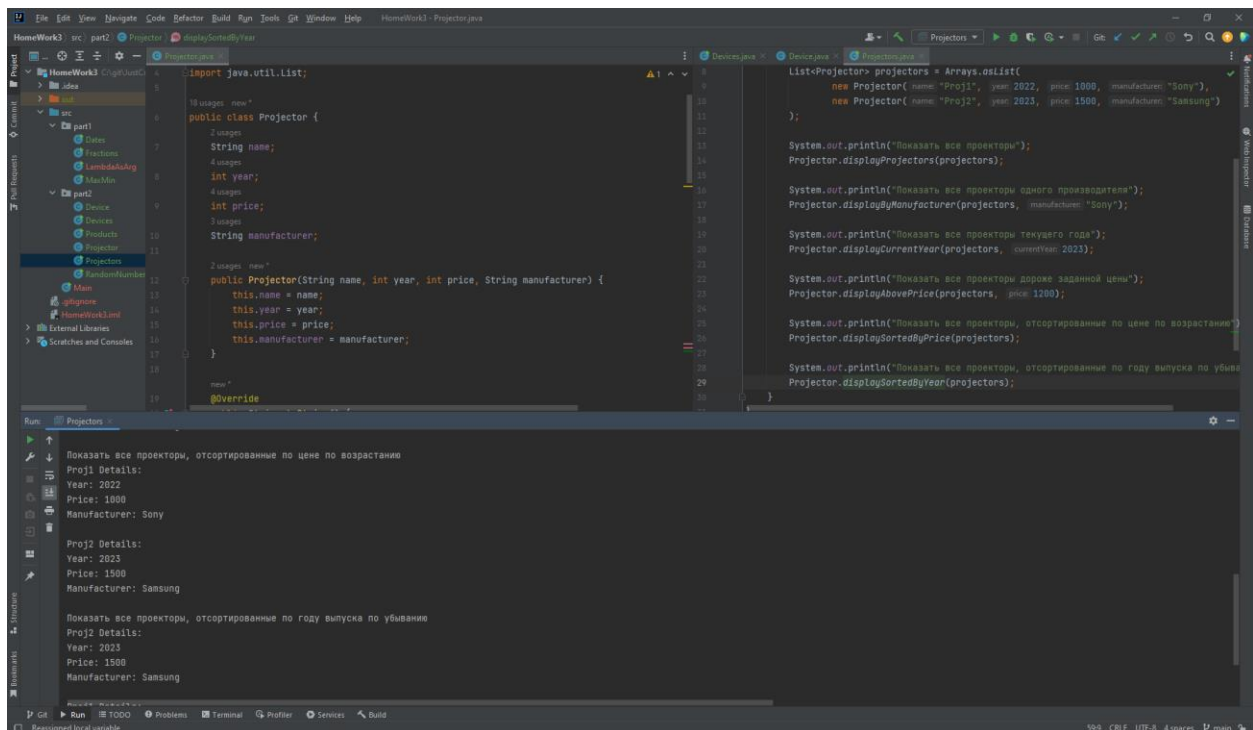
Color: Black

Type: Type1

Задание 4

Создайте класс «Проектор». Он должен хранить информацию о названии проектора, год выпуска, цена, производитель. Нужно создать набор проекторов и выполнить следующие задачи:

- Показать все проекторы;
- Показать все проекторы одного производителя;
- Показать все проекторы текущего года;
- Показать все проекторы дороже заданной цены;
- Показать все проекторы, отсортированные по цене по возрастанию;
- Показать все проекторы, отсортированные по цене по убыванию;
- Показать все проекторы, отсортированные по году выпуска по возрастанию;
- Показать все проекторы, отсортированные по году выпуска по убыванию.








```



public static void main(String[] args) {
    List<Projector> projectors = Arrays.asList(
        new Projector( name: "Proj1", year: 2022, price: 1000, manufacturer: "Sony"),
        new Projector( name: "Proj2", year: 2023, price: 1500, manufacturer: "Samsung")
    );
}



```



Run: Projectors x




 Price: 1500
 Manufacturer: Samsung




 Показать все проекторы одного производителя
 Proj1 Details:
 Year: 2022
 Price: 1000
 Manufacturer: Sony




 Показать все проекторы текущего года
 Proj2 Details:
 Year: 2023
 Price: 1500
 Manufacturer: Samsung


 Показать все проекторы дороже заданной цены
 Proj2 Details:
 Year: 2023
 Price: 1500
 Manufacturer: Samsung


 Показать все проекторы, отсортированные по цене по возрастанию
 Proj1 Details:
 Year: 2022
 Price: 1000
 Manufacturer: Sony


 Proj2 Details:
 Year: 2023
 Price: 1500
 Manufacturer: Samsung


 Показать все проекторы, отсортированные по году выпуска по убыванию
 Proj2 Details:
 Year: 2023
 Price: 1500
 Manufacturer: Samsung


 Proj1 Details:
 Year: 2022
 Price: 1000
 Manufacturer: Sony