Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации
Бюджетное образовательное учреждение высшего образования
Московский технический университет связи и информатики

Кафедра «Математической кибернетики и информационных технологий»

Дисциплина «Структуры и алгоритмы обработки данных»

Отчёт по курсовой работе

Выполнил студент
Группы БФИ1901
Курбатов А. О.

Проверил
Кутейников И. А.

Москва 2021

Задание

Задача 1. «Треугольник с максимальным периметром»

Массив `A` состоит из целых положительных чисел - длин отрезков. Составьте из трех отрезков такой треугольник, чтобы его периметр был максимально возможным. Если невозможно составить треугольник с положительной площадью - функция возвращает `0`.

Пример 1.1:

Ввод: `[2,1,2]`

Вывод: `5`

Пример 1.2:

Ввод: `[1,2,1]`

Вывод: `0`

Пример 1.3:

Ввод: `[3,2,3,4]`

Вывод: `10`

Пример 1.4:

Ввод: `[3,6,2,3]`

Вывод: `8`

Ограничения:

- $3 \leqslant len(A) \leqslant 10000$
- $1 \leqslant A[i] \leqslant 10^6$

Задача 2. «Максимальное число»

Дан массив неотрицательных целых чисел `nums`. Расположите их в таком порядке, чтобы вместе они образовали максимально возможное число.

Замечание: Результат может быть очень большим числом, поэтому представьте его как `string`, а не `integer`.

Пример 2.1:

Ввод: `nums = [10,2]`

Вывод: `"210"`

Пример 2.2:

Ввод: `nums = [3,30,34,5,9]`

Вывод: `"9534330"`

Пример 2.3:

Ввод: `nums = [1]`

Вывод: `"1"`

Пример 2.4:

Ввод: `nums = [10]`

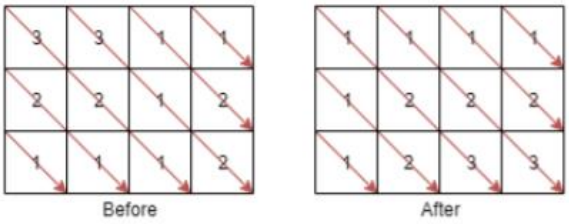
Вывод: `"10"`

Ограничения:

- $1 \leqslant len(nums) \leqslant 100$
- $0 \leqslant nums[i] \leqslant 10^9$

Задача 3. «Сортировка диагоналей в матрице»

Дана матрица `mat` размером `m * n`, значения - целочисленные. Напишите функцию, сортирующую каждую диагональ матрицы по возрастанию и возвращающую получившуюся матрицу.



Пример 3.1:

Ввод: `mat = [[3, 3, 1, 1], [2, 2, 1, 2], [1, 1, 1, 2]]`

Вывод: `[[1, 1, 1, 1], [1, 2, 2, 2], [1, 2, 3, 3]]`

Пример 3.2:

Ввод: `mat = [[11, 25, 66, 1, 69, 7], [23, 55, 17, 45, 15, 52], [75, 31, 36, 44, 58, 8], [22, 27, 33, 25, 68, 4], [84, 28, 14, 11, 5, 50]]`

Вывод: `[[5, 17, 4, 1, 52, 7], [11, 11, 25, 45, 8, 69], [14, 23, 25, 44, 58, 15], [22, 27, 31, 36, 50, 66], [84, 28, 75, 33, 55, 68]]`

Ограничения:

- $m == len(mat)$
- $n == len(mat[i])$
- $1 \leqslant m, n \leqslant 100$
- $1 \leqslant mat[i][j] \leqslant 100$

Задача 1. «Шарики и стрелы»

Некоторые сферические шарики распределены по двухмерному пространству. Для каждого шарика даны x-координаты начала и конца его горизонтального диаметра. Так как пространство двумерно, то y-координаты не имеют значения в данной задаче. Координата x_{start} всегда меньше x_{end}.

Стрелу можно выстрелить строго вертикально (вдоль y-оси) из разных точек x-оси. Шарик с координатами x_{start} и x_{end} уничтожается стрелой, если она была выпущена из такой позиции x, что $x_{start} \leqslant x \leqslant x_{end}$. Когда стрела выпущена, она летит в пространстве бесконечное время (уничтожая все шарики на пути).

Дан массив `points`, где `points[i] = [`x_{start}`,` x_{end}`]`. Напишите функцию, возвращающую минимальное количество стрел, которые нужно выпустить, чтобы уничтожить все шарики.

Пример 1.1:

Ввод: `points = [[10,16],[2,8],[1,6],[7,12]]`

Вывод: `2`

Пример 1.2:

Ввод: `points = [[1,2],[3,4],[5,6],[7,8]]`

Вывод: `4`

Пример 1.3:

Ввод: `points = [[1,2],[2,3],[3,4],[4,5]]`

Вывод: `2`

Пример 1.4:

Ввод: `points = [[1,2]]`

Вывод: `1`

Пример 1.5:

Ввод: `points = [[2,3],[2,3]]`

Вывод: `1`

Ограничения:

- $0 \leqslant len(points) \leqslant 10^4$
- $len(points[i]) == 2$
- $-2^{31} \leqslant x_{start} < x_{end} \leqslant 2^{31} - 1$

ЗАДАЧА 1

Даны две строки: s1 и s2 с одинаковым размером, проверьте, может ли некоторая перестановка строки s1 “победить” некоторую перестановку строки s2 или наоборот.

Строка x может “победить” строку y (обе имеют размер n), если x[i]> = y [i] (в алфавитном порядке) для всех i от 0 до n-1.

Примеры:

```
Input: s1 = "abc", s2 = "xya"
Output: true
```

```
Объяснение: «ayx» - это перестановка строки s2 = «xya», которая “побеждает” строку s1 = «abc».
```

```
Input: s1 = "abe", s2 = "acd"
```

```
Output: false
```

```
Объяснение: Все перестановки для s1 = "abe": "abe", "aeb", "bae", "bea", "eab" и "eba", а все перестановки для s2 = "acd": "acd", «adc», «cad», «cda», «dac» и «ca». Однако нет никакой перестановки строки s1, которая может нарушить некоторую перестановку строки s2 и наоборот.
```

```
s1.length == n
```

```
s2.length == n
```

```
1 <= n <= 10^5
```

ЗАДАЧА 2

Дана строка s, вернуть самую длинную полиндромную подстроку в s.

Примеры:

```
Input: s = "babad"
Output: "bab"
Note: "aba" is also a valid answer.
```

```
Input: s = "cbbd"
Output: "bb"
```

ЗАДАЧА 3

Вернуть количество отдельных непустых подстрок текста, которые могут быть записаны как конкатенация некоторой строки с самой собой (т.е. она может быть записана, как a + a, где a - некоторая строка).

Примеры:

```
Input: text = "abcabcabc"
```

```
Output: 3
```

```
Explanation: The 3 substrings are "abcabc", "bcabca"
and "cabcab".
```

Задача 1. «Стопки монет»

На столе стоят $3n$ стопок монет. Вы и ваши друзья Алиса и Боб забираете стопки монет по следующему алгоритму:

1. Вы выбираете 3 стопки монет из оставшихся на столе.
2. Алиса забирает себе стопку с максимальным количеством монет.
3. Вы забираете одну из двух оставшихся стопок.
4. Боб забирает последнюю стопку.
5. Если еще остались стопки, то действия повторяются с первого шага.

Дан массив целых положительных чисел `piles`. Напишите функцию, возвращающую максимальное число монет, которое вы можете получить.

Пример 1.1:

Ввод: `piles = [2,4,1,2,7,8]`

Вывод: `9`

Пример 1.2:

Ввод: `piles = [2,4,5]`

Вывод: `4`

Пример 1.3:

Ввод: `piles = [9,8,7,6,5,1,2,3,4]`

Вывод: `18`

Ограничения:

- $3 \leqslant len(piles) \leqslant 10^5$
- $len(piles)\, mod\, 3 == 0$
- $1 \leqslant piles[i] \leqslant 10^4$

Листинг

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Random;

public class Tasks {

    public static void main(String[] args) {

        // Первый блок задач
```

```
        System.out.println("Задача 1:");

        int[] mas = {3, 6, 2, 3};

        System.out.println("Периметр = " + zadacha1(mas));


        System.out.println("\nЗадача 2:");

        int len = 10, min = 0, max = 999;
        int[] mass = new int[len];
        Random rand = new Random();
        for (int i = 0; i < len; i++) {
            mass[i] = min + rand.nextInt(max - min + 1);
        }

        System.out.println("Максимально число = " + zadacha2(mass, len));


        System.out.println("\nЗадача 3:\n");

        int m = 10, n = 10, min_limit = 0, max_limit = 100;
        int[][] array = new int[m][n];
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                array[i][j] = min_limit + rand.nextInt(max_limit - min_limit +
1);
                System.out.print(array[i][j] + "\t");
            }
            System.out.println();
        }

        zadacha3(array, m, n);

        // Задача про шарики

        int [][] points = {{-10,6},{-2,8},{-6,-1},{7,12}};
        System.out.println("\nЗадача про шарики и стрелы:");
        System.out.println(balloons(points));

        // Второй блок задач

        System.out.println("\nВторой блок задач. Задачи со строками");
        System.out.println("\nЗадача 1:");
        System.out.println(task1("abe", "acd"));

        System.out.println("\nЗадача 2:");
        System.out.println(task2("babad"));

        System.out.println("\nЗадача 3:");
        System.out.println(task3("abcabcabc"));


        // Задача про монеты

        System.out.println("\nЗадача про стопки монет:");
        int [] piles = {9,8,7,6,5,1,2,3,4};
        System.out.println(coins(piles));
```

```
    }


    public static int zadacha1(int[] mas) {
        int a, b, c;
        int per = 0;
        for (int i = 0; i < mas.length - 2; i++) {
            a = mas[i];
            for (int j = i + 1; j < mas.length - 1; j++) {
                b = mas[j];
                for (int k = j + 1; k < mas.length; k++) {
                    c = mas[k];
                    if ((a + b > c) && (a + c > b) && (b + c > a) && (a + b + c
> per)) {
                        per = a + b + c;
                    }
                }
            }
        }
        return per;
    }

    public static String zadacha2(int[]mass, int len) {
        boolean check = true;
        int swap;
        if(len < 1){
            return "В массиве нет чисел";
        }
        String str1, str2, str3, ans = "";
        int num1, num2;
        while(check){
            check=false;
            for (int i = mass.length - 1; i > 0; i--) {
                str1 = Integer.toString(mass[i]) + Integer.toString(mass[i -
1]);
                str2 = Integer.toString(mass[i - 1]) +
Integer.toString(mass[i]);
                num1 = Integer.parseInt(str1);
                num2 = Integer.parseInt(str2);
                if (num1 > num2) {
                    swap = mass[i];
                    mass[i] = mass[i - 1];
                    mass[i - 1] = swap;
                    check=true;
                }
            }
        }
        for (int num : mass) {
            str3 = Integer.toString(num);
            ans += str3;
        }
       return ans;
    }


    public static void zadacha3(int [][] array, int m, int n) {
        int swap;
        boolean check = true;
```

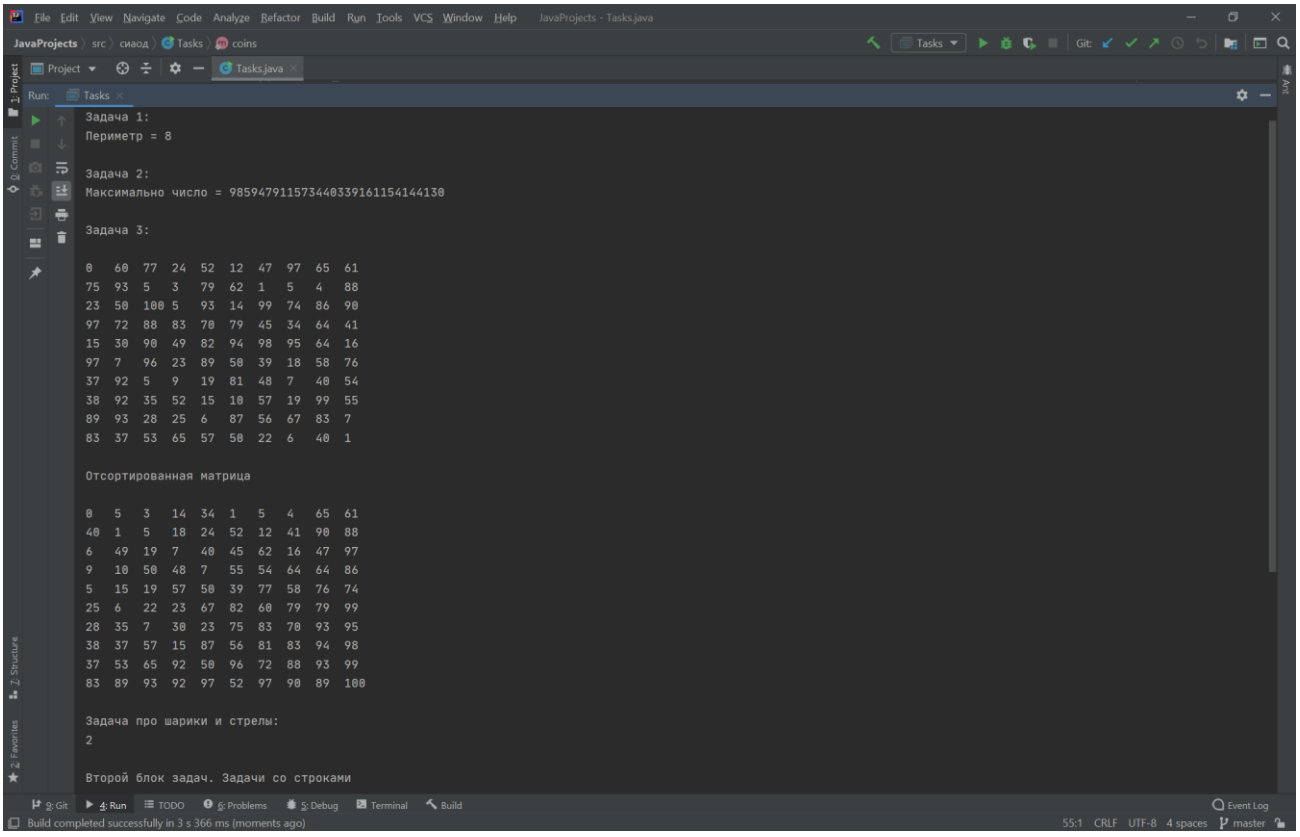
```
        // сортировка главной диагонали

        while (check){
            check = false;
            int i = 0;
            while (i < n - 1 && i < m - 1){
                if (array[i][i] > array[i + 1][i + 1]) {
                    swap = array[i][i];
                    array[i][i] = array[i + 1][i + 1];
                    array[i + 1][i + 1] = swap;
                    check = true;
                }
                i++;
            }
        }

        // сортировка под главной диагональю

        check = true;
        while (check){
            check = false;
            for(int i = m - 2; i > 0; i--) {
                int j = 0;
                while (j < n - 1 && j < m - 1){
                    if (array[i][j] > array[i + 1][j + 1]) {
                        swap = array[i][j];
                        array[i][j] = array[i + 1][j + 1];
                        array[i + 1][j + 1] = swap;
                        check = true;
                    }
                    j++;
                }
            }
        }

        // сортировка над главной диагональю

        check = true;
        while (check){
            check = false;
            for(int i = n - 2; i > 0; i--) {
                int j = 0;
                while (j < n - 1 && j < m - 1){
                    if (array[j][i] > array[j + 1][i + 1]) {
                        swap = array[j][i];
                        array[j][i] = array[j + 1][i + 1];
                        array[j + 1][i + 1] = swap;
                        check = true;
                    }
                    j++;
                }
            }
        }

        System.out.println("\nОтсортированная матрица\n");

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(array[i][j] + "\t");
```

```
            }
            System.out.println();
        }
    }


    public static int balloons (int [][] points){
        ArrayList<String> array = new ArrayList<>();
        for (int[] point : points) {
            array.add(point[0] + "," + (point[1]));
        }
            for (int i = 0; i < array.size() - 1; i++) {
                for (int j = i + 1; j < array.size();) {
                    int index1 = array.get(i).indexOf(",");
                    int x1 = Integer.parseInt(array.get(i).substring(0,
index1));
                    int x2 = Integer.parseInt(array.get(i).substring(index1 +
1));
                    int index2 = array.get(j).indexOf(",");
                    int x3 = Integer.parseInt(array.get(j).substring(0,
index2));
                    int x4 = Integer.parseInt(array.get(j).substring(index2 +
1));
                    if ((x1 <= x4 && x4 <= x2) || (x1 <= x3 && x3 <= x2)) {
                        array.add((Math.max(x1, x3)) + "," + Math.min(x2, x4));
                        array.remove(i);
                        array.remove(j - 1);
                        i = 0;
                        j = i + 1;
                    } else {
                        j++;
                    }
                }
            }
        return array.size();
    }

    public static boolean task1(String s1, String s2){
        if(s1.length() != s2.length()){
            System.out.println("Длины строк должны быть равны");
        }
        char [] str1 = s1.toCharArray();
        char [] str2 = s2.toCharArray();
        Arrays.sort(str1);
        Arrays.sort(str2);
        System.out.println(str1);
        System.out.println(str2);
        return checkStrings(str1, str2) || checkStrings(str2, str1);
    }

    public static boolean checkStrings(char [] str1, char [] str2){
        int counter = 0;
        for(int i = 0; i < str1.length; i++){
            if(str1[i] >= str2[i]){
                counter++;
            }
        }
        return counter == str1.length;
    }
```

```java
public static String task2(String s){
    String substring = "";
    int maxLen = 0;
    if(s.contains(" ")){
        return "В строке не должно быть пробелов";
    }
    for(int i = 0; i < s.length(); i++) {
        char letter = s.charAt(i);
        int index = s.indexOf(letter, i + 1);
        if (index == -1) {
            continue;
        }
        while (index != -1){
            String check = s.substring(i, index + 1);
            String reversed = "";

            for(int j = check.length() - 1; j >= 0; j--){
                reversed += check.charAt(j);
            }

            if(check.equals(reversed) && check.length() > maxLen){
                maxLen = check.length();
                substring = check;
            }
            index = s.indexOf(letter, index + 1);
        }
    }
    return substring;
}

public static int task3(String text){
    ArrayList<String> substrings = new ArrayList<>();
    if(text.contains(" ")){
        System.out.println("В строке не должно быть пробелов");
        return 0;
    }
    for(int i = 0; i < text.length(); i++) {
        char letter = text.charAt(i);
        int index = text.indexOf(letter, i + 1);
        if (index == -1) {
            continue;
        }

        while (index + index - i - 1 < text.length() && index != -1){
            String left = text.substring(i, index);
            String right = text.substring(index, index + index - i);
            if (left.equals(right) && !substrings.contains(left + right)) {
                substrings.add(left + right);
            }
            index = text.indexOf(letter, index + 1);
        }
    }
    System.out.println(substrings);
    return substrings.size();
}

public static int coins (int [] piles){
    int sum = 0;
```

```
        if(piles.length == 0){
            System.out.println("На столе нет стопок монет");
            return 0;
        }
        if(piles.length % 3 != 0){
            System.out.println("На столе должно быть 3n стопок монет");
            return 0;
        }

        // Для достижения максимума каждый раз мы выбираем две стопки с
наибольшим количеством монет и одну с наименьшим.
        // Тогда нам нужно просмотреть 2 * (piles.length / 3) наибольших стопок,
из которых каждую вторую мы забираем себе

        int count = 2 * (piles.length / 3);
        Arrays.sort(piles);
        for (int i = piles.length - 2; i >= piles.length - count - 1; i-=2){
            sum += piles[i];
        }
        return sum;
    }
}
```

Результат выполнения работы



```
Второй блок задач. Задачи со строками

Задача 1:
abe
acd
false

Задача 2:
bab

Задача 3:
[abcabc, bcabca, cabcab]
3

Задача про стопки монет:
18
```

Вывод

В ходе курсовой работы мы выполнили реализацию поставленных задач.