Министерство цифрового развития, связи и массовых коммуникаций

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»



Отчет по курсовой работе

по дисциплине «Структура и алгоритмы обработки данных»

Выполнила: студент группы

БВТ1902

Соколова А.Ю.

Москва

2021 г

Оглавление

[Цель работы 3](#_Toc72701661)

[Задание 1 «Треугольник с максимальным периметром» 3](#_Toc72701662)

[Код программы 3](#_Toc72701663)

[Задание 2 «Максимальное число» 4](#_Toc72701664)

[Код программы 4](#_Toc72701665)

[Задание 3 «Сортировка диагоналей в матрице» 5](#_Toc72701666)

[Код программы 5](#_Toc72701667)

[Задание 4 «Шарики и стрелы» 6](#_Toc72701668)

[Код программы 6](#_Toc72701669)

[Задание 5 «Стопки монет» 8](#_Toc72701670)

[Код программы 8](#_Toc72701671)

[Задание 6 «Победная строка» 9](#_Toc72701672)

[Код программы 9](#_Toc72701673)

[Задание 7 «Палиндром» 10](#_Toc72701674)

[Код программы 10](#_Toc72701675)

[Задание 8 11](#_Toc72701676)

[Код программы 11](#_Toc72701677)

[Код программы Main 12](#_Toc72701678)

[Снимки экрана работы программ 14](#_Toc72701679)

[Вывод 14](#_Toc72701680)

## Цель работы

Написать программу для решения ниже поставленных задач.

## Задание 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.

## Код программы

static int Task1(int[] arr) {

System.out.println(" # Task 1");

Arrays.sort(arr);

arr = forTask1(arr);

for(int i = 0; i < arr.length-2; i++)

{

if ((arr[i] < (arr[i+1] + arr[i+2])) && (arr[i+1] < (arr[i] + arr[i+2])) && (arr[i+2] < (arr[i+1] + arr[i])))

{

int P=arr[i] + arr[i+1] + arr[i+2];

System.out.println("Максимальный периметр равен: "+P);

return (arr[i] + arr[i+1] + arr[i+2]);

}

}

return 0;

}

static int[] forTask1(int[] myArray){

int size = myArray.length;

for (int i = 0; i < size / 2; i++) {

int temp = myArray[i];

myArray[i] = myArray[size - 1 - i];

myArray[size - 1 - i] = temp;

}

return myArray;

}

## Задание 2 «Максимальное число»

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

## Код программы

static void Task2(int[] arr)

{

System.out.println(" # Task 2");

System.out.println("Максимальное число равно: " + maxNum(arr));

System.out.println();

}

public static String maxNum(int[] nums) {

String str = "";

List<Integer> list = new ArrayList<>(nums.length);

for (int x : nums) {

list.add(x);

}

list.sort((a, b) -> measure(b) - measure(a));

for (int x : list) {

str += x;

}

return str;

}

public static int measure(int n) {

if (n < 10) { return 100\*n + 10\*n + n; }

else if (n < 100) { return 10\*n + n%10; }

else if (n < 1000) { return n; }

else { return -1; }

}

## Задание 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.

## Код программы

static void Task3(int[][] arr, int m,int n){

System.out.println(" # Task 3");

for(int i = 0; i < n-1; i++){

FuncForTask3(arr, 0,i,m,n);

}

for (int i = 1; i < m - 1; i++)

{

FuncForTask3(arr, i, 0, m, n);

}

for(int i = 0; i < n-1; i++){

for (int j = 1; j < m - 1; j++){

System.out.print(arr[i][j]+" ");

}

System.out.println();

}

System.out.println();

}

static void FuncForTask3(int[][] arr, int m,int k,int lenX,int lenY) {

ArrayList<Integer> NewArr = new ArrayList<>();

int m1 = m;

int k1 = k;

while (ProvForTask3(m1,k1,lenX,lenY)) {

NewArr.add(arr[m1][k1]);

m1++;

k1++;

}

Collections.sort(NewArr);

int g = 0;

while (ProvForTask3(m, k, lenX, lenY)) {

arr[m][k]=NewArr.get(g);

m++;

k++;

g++;

}

}

static boolean ProvForTask3(int indexX,int indexY, int i,int j) {

if (indexX < i && indexY < j) return true;

else return false;

}

## Задание 4 «Шарики и стрелы»

Дан массив points, где points[i] = [xstart, xend]. Напишите функцию, возвращающую минимальное количество стрел, которые нужно выпустить, чтобы уничтожить все шарики.

## Код программы

public class ballon

{

private int startx;

private int endx;

public ballon(){};

public ballon(int start, int end)

{

startx=start;

endx=end;

}

public int getStart() { return startx; }

public int getEnd() { return endx; }

public static ballon generator()

{

int startx=(int) (Math.random() \* 20);

int endx=startx+ (int) (Math.random() \* 20);

return new ballon(startx, endx);

}

public static Vector<ballon> sort (Vector<ballon> ballons)

{

for (int i=0; i<ballons.size()-1; i++)

{

for (int j=i+1; j<ballons.size(); j++)

{

if (ballons.elementAt(i).getEnd() > ballons.elementAt(j).getEnd())

{

Collections.swap(ballons, i, j);

}

}

}

return ballons;

}

}

static void Task8(Vector<ballon> ballons) {

System.out.println(" # Task 8");

int amount;

int arrow;

int count = 0;

int temp = 0;

ballons= ballon.sort(ballons);

System.out.print("Сгенерированные шары: ");

for (int i=0; i<ballons.size(); i++)

{

System.out.print(ballons.elementAt(i).getStart()+" "+ballons.elementAt(i).getEnd()+"; ");

}

System.out.println();

System.out.println();

while (temp!=ballons.size()-1)

{

count++;

arrow=ballons.elementAt(temp).getEnd();

while (arrow >= ballons.elementAt(temp).getStart() && arrow <= ballons.elementAt(temp).getEnd())

{

if (temp == ballons.size()-1)

break;

temp++;

}

}

System.out.println("Минимальное количество стрел: "+count);

}

## Задание 5 «Стопки монет»

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

## Код программы

static void Task4(Vector<Integer> arr) {

System.out.println(" # Task 4");

Collections.sort(arr);

Collections.reverse(arr);

int sum = 0;

int n = 0;

int count = arr.size();

while (n!=count/3) {

arr.remove(0);

sum += arr.get(0);

arr.remove(0);

n++;

}

System.out.println(sum);

System.out.println();

}

## Задание 6 «Победная строка»

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.

## Код программы

static void Task5(char[] ch1,char[]ch2) {

System.out.println(" # Task 5");

char[] ch3=ch1;

char[] ch4=ch2;

Arrays.sort(ch1);

Arrays.sort(ch2);

if(ch3==ch1) {

char temp = ch1[ch1.length-1];

ch1[ch1.length - 1] = ch1[ch1.length - 2];

ch1[ch1.length - 2] = temp;

}

if (ch4 == ch2) {

char temp = ch2[ch1.length - 1];

ch2[ch1.length - 1] = ch2[ch1.length - 2];

ch2[ch1.length - 2] = temp;

}

String st1 = "";

String st2 = "";

for(int i = 0; i < ch1.length; i++) {

st1 += ch1[i];

st2 += ch2[i];

}

if (st1.equals(st2) == true) {

System.out.println("Ни одна перестановка второй строки не победит первую строку");

} else {

System.out.println("Вторая строка побеждает");

}

System.out.println();

}

## Задание 7 «Палиндром»

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

## Код программы

static void Task6(String str8) {

System.out.println(" # Task 6");

int n = 0;

String str="";

String ans="";

while (n != str8.length()) {

String s = str8.substring(0, str8.length() - n);

if (IsPalindrom(s))

{

if (ans.length()<s.length())

ans=s;

}

n++;

}

for (int i=0; i<str8.length()-1; i++)

{

for (int j=i+1; j< str8.length(); j++)

{

str+=str8.charAt(j);

if (IsPalindrom(str))

{

if (ans.length()<str.length())

ans=str;

}

}

}

System.out.println(ans);

System.out.println();

}

static boolean IsPalindrom(String str)

{

StringBuilder str1= new StringBuilder(str);

str1.reverse();

return str.equals(str1.toString());

}

## Задание 8

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).

## Код программы

static void Task7(String str) {

System.out.println(" # Task 7");

boolean first=true;

int ans=0;

String substring;

String buffer="";

for (int i=0; (str.length() / 2) > i; i++)

{

buffer="";

if (first)

{

for (int j = 0; j <= (str.length() / 2); j++)

{

buffer += str.charAt(j);

substring = buffer + buffer;

if (str.indexOf(substring) != -1)

ans++;

}

first=false;

}

buffer="";

for (int j = i+1; j <= (str.length() / 2); j++)

{

buffer += str.charAt(j);

substring = buffer + buffer;

if (str.indexOf(substring) != -1)

ans++;

}

}

System.out.println(ans);

System.out.println();

}

## Код программы Main

public class Main {

public static void main(String[] args) {

int[] arr={2,6,22,14,17,9,8};

zadaniya.Task1(arr);

int[] arr2={2,6,22};

zadaniya.Task2(arr2);

int n = 10;

int m = 10;

int[][] matrix = Generator.matrix(m, n);

zadaniya.Task3(matrix,m, n);

int[] temp={9,8,7,6,5,1,2,3,4};

Vector<Integer> vector= new Vector<Integer>();

for (int i: temp)

vector.add(i);

zadaniya.Task4(vector);

String s1 = "abe";

String s2 = "acd";

char[] ch1=s1.toCharArray();

char[] ch2=s2.toCharArray();

zadaniya.Task5(ch1,ch2);

zadaniya.Task6("cbbd");

zadaniya.Task7("abcabcabc");

Vector<ballon> ballons= new Vector<>();

for (int i=0; i<15; i++)

ballons.add(ballon.generator());

zadaniya.Task8(ballons);

}

}

public class Generator

{

public static int[][] matrix(int m, int n)

{

int[][] matrix= new int[m][n];

for (int i=0; i<n; i++)

{

for (int j=0; j<m; j++)

{

matrix[i][j]=((int) (Math.random() \* 10));

}

}

return matrix;

}

}

## Снимки экрана работы программ

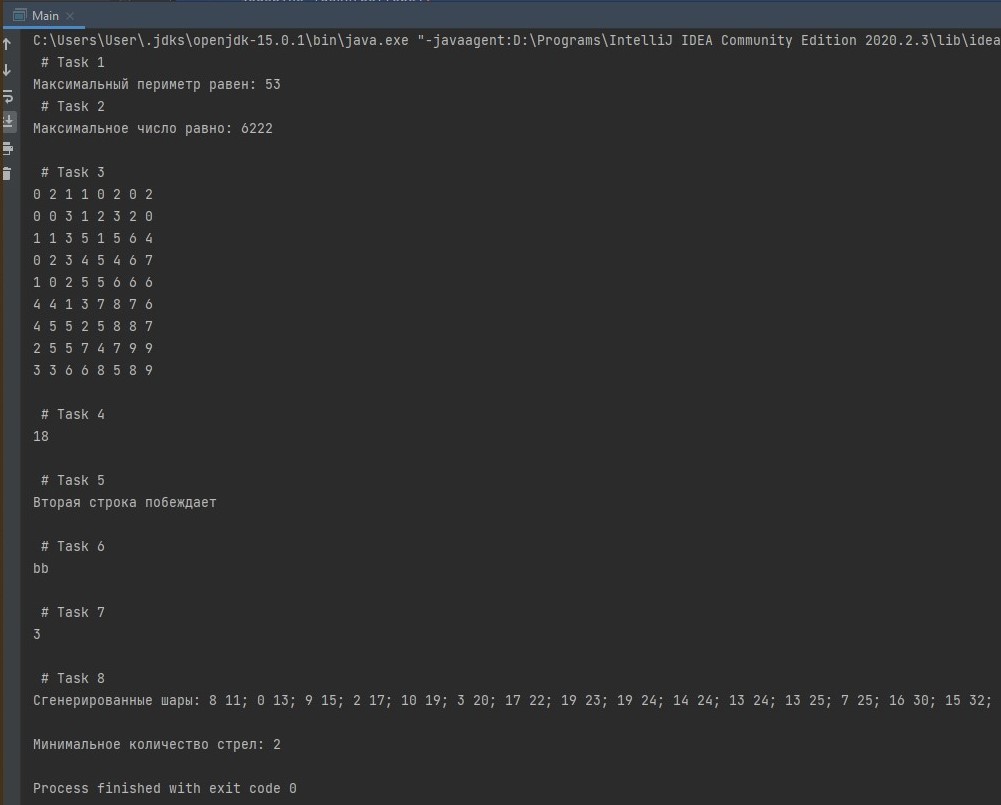


Рисунок 1 – Результат выполнения программы

## Вывод

Я написала программу для решения выше поставленных задач.