Министерство цифрового развития, связи и массовых коммуникаций
Ордена Трудового Красного Знамени федеральное государственное бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

Контрольные задачи
по дисциплине «Структуры и алгоритмы обработки данных»

Выполнил: студент группы БВТ1902
Лапин Виктор Андреевич

Проверил: Кутейников Иван Алексеевич

Москва
2021

Оглавление

1. ВВЕДЕНИЕ

Цель работы – разработать алгоритмы для решения представленных задач.

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.

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.

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.

Задача 4. «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.

Задача 5.

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.

Задача 6.

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

Задача 7.

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).

Задача 8. «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.

2. ВЫПОЛНЕНИЕ РАБОТЫ

Задачи 1 – 3.

```
1   import java.util.*;
2
3   public class Pack {
4     public static void main(String[] args) {
5       Scanner sc;
6       String[] strArray;
7       int[] intArray;
8       while (true) {
9         System.out.println("Выберите операцию:");
10        System.out.println("1. Треугольник с максимальным периметром");
11        System.out.println("2. Максимальное число");
12        System.out.println("3. Сортировка диагоналей");
13        System.out.println("0. Выход");
14        sc = new Scanner(System.in);
15        int operation = sc.nextInt();
16        switch (operation) {
17        case 1:
18          System.out.print("Введите массив чисел через пробел: ");
19          sc.useDelimiter("\n");
20          strArray = sc.next().split(" ");
21          intArray = new int[strArray.length];
22          for (int i = 0; i < strArray.length; i++) {
23            intArray[i] = Integer.parseInt(strArray[i]);
24          }
25          System.out.println(maxPerimeter(intArray));
26          break;
27        case 2:
28          System.out.print("Введите массив чисел через пробел: ");
29          sc.useDelimiter("\n");
30          strArray = sc.next().split(" ");
31          intArray = new int[strArray.length];
32          for (int i = 0; i < strArray.length; i++) {
33            intArray[i] = Integer.parseInt(strArray[i]);
34          }
35          System.out.println(maxNumber(intArray));
36          break;
37        case 3:
38          int[][] matrix = Matrix.matrix(6, 4, 1, 9);
39          System.out.println("Исходная матрица:");
40          for (int i = 0; i < matrix.length; i++) {
41            for (int j = 0; j < matrix[0].length; j++) {
42              System.out.print(matrix[i][j] + " ");
43            }
44            System.out.println();
45          }
46          matrix = diagSort(matrix);
47          System.out.println("Результат:");
```

```
48            for (int i = 0; i < matrix.length; i++) {
49              for (int j = 0; j < matrix[0].length; j++) {
50                System.out.print(matrix[i][j] + " ");
51              }
52              System.out.println();
53            }
54            break;
55          case 0:
56            return;
57          }
58          sc.useDelimiter("\n");
59          sc.next();
60        }
61      }
62
63      public static int maxPerimeter(int[] arr) {
64        int[] tempArr = new int[arr.length];
65        for (int cnt = 0; cnt < arr.length - 2; cnt++) {
66          System.arraycopy(arr, 0, tempArr, 0, arr.length);
67          int[] maxTriangle = new int[3];
68          int[] maxTriangleIndex = new int[3];
69          for (int i = 0; i < 3; i++) {
70            for (int j = 0; j < tempArr.length; j++) {
71              if (tempArr[j] > maxTriangle[i]) {
72                maxTriangle[i] = tempArr[j];
73                maxTriangleIndex[i] = j;
74              }
75            }
76            tempArr[maxTriangleIndex[i]] = 0;
77          }
78          if (maxTriangle[0] < maxTriangle[1] + maxTriangle[2]) {
79            return maxTriangle[0] + maxTriangle[1] + maxTriangle[2];
80          } else {
81            arr[maxTriangleIndex[0]] = 0;
82          }
83        }
84        return 0;
85      }
86
87      public static String maxNumber(int[] arr) {
88        StringBuilder out = new StringBuilder();
89        for (int i = 0; i < arr.length; i++) {
90          int largestNumberIndex = -1;
91          int len = Integer.MAX_VALUE;
92          int firstDigit = 0;
93          for (int j = 0; j < arr.length; j++) {
94            int newFirstDigit = arr[j];
95            int newLen = 1;
96            while (newFirstDigit / 10 > 0) {
97              newFirstDigit /= 10;
98              newLen++;
99            }
```

```
100           if (newFirstDigit > firstDigit) {
101             len = newLen;
102             firstDigit = newFirstDigit;
103             largestNumberIndex = j;
104           } else if (newFirstDigit == firstDigit) {
105             int oldNumber = arr[largestNumberIndex];
106             int newNumber = arr[j];
107             if (len > newLen) {
108               for (int k = 0; k < len - newLen; k++) {
109                 newNumber *= 10;
110               }
111             } else {
112               for (int k = 0; k < newLen - len; k++) {
113                 oldNumber *= 10;
114               }
115             }
116             if ((oldNumber == newNumber) && (len > newLen)) {
117               len = newLen;
118               firstDigit = newFirstDigit;
119               largestNumberIndex = j;
120             }
121             if (oldNumber < newNumber) {
122               len = newLen;
123               firstDigit = newFirstDigit;
124               largestNumberIndex = j;
125             }
126           }
127         }
128         out.append(arr[largestNumberIndex]);
129         arr[largestNumberIndex] = -1;
130       }
131       return out.toString();
132     }
133
134     public static int[][] diagSort(int[][] arr) {
135       int len = 1;
136       int change = 1;
137       for (int i = arr.length - 1; i > 0; i--) {
138         int[] diag = new int[len];
139         for (int j = 0; j < len; j++) {
140           diag[j] = arr[i + j][j];
141           if ((i + j == arr.length - 1) && (j == arr[0].length - 1)) {
142             change--;
143           }
144         }
145         diag = Matrix.quickSort(diag);
146         for (int j = 0; j < len; j++) {
147           arr[i + j][j] = diag[j];
148         }
149         len += change;
150       }
151       change--;
```

```
    for (int i = 0; i < arr[0].length; i++) {
      int[] diag = new int[len];
      for (int j = 0; j < len; j++) {
        diag[j] = arr[j][i + j];
        if ((j == arr.length - 1) && (i + j == arr[0].length - 1)) {
          change--;
        }
      }
      diag = Matrix.quickSort(diag);
      for (int j = 0; j < len; j++) {
        arr[j][i + j] = diag[j];
      }
      len += change;
    }
    return arr;
  }
}
```

Задача 4.

```
import java.util.Scanner;

public class BalloonsArrows {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    sc.useDelimiter("\n");
    System.out.print("Введите координаты шариков через пробел: ");
    String str = sc.next();
    String[] arr = str.split(" ");
    int[][] coords = new int[arr.length / 2][2];
    for (int i = 0; i < coords.length; i++) {
      coords[i][0] = Integer.parseInt(arr[i * 2]);
      coords[i][1] = Integer.parseInt(arr[i * 2 + 1]);
    }
    System.out.println("Результат: " + minArrowNumber(coords));
  }

  public static int minArrowNumber(int[][] arr) {
    int[][] coords = new int[arr.length * 2][2];
    int out = 0;
    boolean[] destroyed = new boolean[arr.length];
    for (int i = 0; i < arr.length; i++) {
      coords[i * 2][0] = arr[i][0];
      coords[i * 2 + 1][0] = arr[i][1];
    }
    while (true) {
      // Выбор новой координаты
      int coordMaxNumBalloons = 0;
      int maxNumBalloons = 0;
      for (int i = 0; i < coords.length; i++) {
        int k = 0;
```

```
32          for (int j = 0; j < arr.length; j++) {
33            if (coords[i][0] >= arr[j][0] &&
                  coords[i][0] <= arr[j][1] &&
                  !destroyed[j]) {
34              k++;
35            }
36          }
37          coords[i][1] = k;
38          if (k > maxNumBalloons) {
39            coordMaxNumBalloons = coords[i][0];
40            maxNumBalloons = k;
41          }
42        }
43        if (maxNumBalloons == 0) {
44          return out;
45        }
46
47        // Выпуск стрелы из координаты coordMaxNumBalloons
48        for (int i = 0; i < arr.length; i++) {
49          if (coordMaxNumBalloons >= arr[i][0] &&
                coordMaxNumBalloons <= arr[i][1] &&
                !destroyed[i]) { // если стрела сбивает шар
50            destroyed[i] = true;
51            for (int j = 0; j < coords.length; j++) {
52              if (coords[j][0] >= arr[i][0] &&
                    coords[j][0] <= arr[i][1]) {
53                coords[j][1]--;
54              }
55            }
56          }
57        }
58        out++;
59      }
60    }
61  }
```

Задачи 5 – 7.

```
1   import java.util.ArrayList;
2   import java.util.Scanner;
3
4   public class Strings {
5     public static void main(String[] args) {
6       String str;
7       Scanner sc;
8       while (true) {
9         System.out.println("Выберите операцию:");
10        System.out.println("1. Может ли перестановка строки победить
                              перестановку другой строки");
11        System.out.println("2. Самая длинная палиндромная подстрока");
12        System.out.println("3. Количество подстрок, которые могут быть
                              записаны как конкатенация");
13        System.out.println("0. Выход");
14        sc = new Scanner(System.in);
15        int operation = sc.nextInt();
16        switch (operation) {
17        case 1:
18          System.out.print("Введите первую строку: ");
19          sc.useDelimiter("\n");
20          str = sc.next();
21          System.out.print("Введите вторую строку: ");
22          System.out.println("Результат: " +
                               stringWin(str, sc.next()));
23          break;
24        case 2:
25          System.out.print("Введите строку: ");
26          sc.useDelimiter("\n");
27          System.out.println("Результат: " +
                               longestPalindrome(sc.next()));
28          break;
29        case 3:
30          System.out.print("Введите строку: ");
31          sc.useDelimiter("\n");
32          System.out.println("Результат: " + numConcat(sc.next()));
33          break;
34        case 0:
35          return;
36        }
37        sc.useDelimiter("\n");
38        sc.next();
39      }
40    }
41
42    public static boolean stringWin(String s1, String s2) {
43      if (s1.length() != s2.length()) {
44        throw new IllegalArgumentException();
45      }
46      int[] c1 = new int[s1.length()];
```

```
47        int[] c2 = new int[s2.length()];
48        for (int i = 0; i < s1.length(); i++) {
49          c1[i] = s1.charAt(i);
50          c2[i] = s2.charAt(i);
51        }
52        c1 = Matrix.insertionSort(c1);
53        c2 = Matrix.insertionSort(c2);
54        boolean firstWins = true;
55        boolean secondWins = true;
56        for (int i = 0; i < c1.length; i++) {
57          if (c1[i] < c2[i]) {
58            firstWins = false;
59          }
60          if (c2[i] < c1[i]) {
61            secondWins = false;
62          }
63        }
64        return firstWins || secondWins;
65      }
66
67      public static String longestPalindrome(String s1) {
68        for (int length = s1.length(); length > 1; length--) {
69          for (int pos = 0; pos <= s1.length() - length; pos++) {
70            boolean isPalindrome = true;
71            for (int k = 0; k < length / 2; k++) {
72              if (s1.charAt(pos + k) != s1.charAt(pos + length - k - 1))
{
73                isPalindrome = false;
74                break;
75              }
76            }
77            if (isPalindrome) {
78              return s1.substring(pos, pos + length);
79            }
80          }
81        }
82        return "";
83      }
84
85      public static int numConcat(String s1) {
86        ArrayList < String > list = new ArrayList < > ();
87        for (int length = 2; length <= (s1.length() / 2) * 2;
               length += 2) { // длины искомых двойных подстрок
88          for (int pos = 0; pos <= s1.length() - length;
                 pos++) { // начальная позиция
89            boolean isConcat = true;
90            for (int k = 0; k < length / 2; k++) {
91              if (s1.charAt(pos + k) != s1.charAt(pos + length / 2 + k)){
92                isConcat = false;
93                break;
94              }
95            }
```

```
96          if (isConcat &&
                !list.contains(s1.substring(pos, pos + length))) {
97            list.add(s1.substring(pos, pos + length));
98          }
99        }
100     }
101     return list.size();
102   }
103 }
```

Задача 8.

```
1   import java.util.Scanner;
2
3   public class Coins {
4     public static void main(String[] args) {
5       Scanner sc = new Scanner(System.in);
6       sc.useDelimiter("\n");
7       System.out.print("Введите массив чисел через пробел: ");
8       String str = sc.next();
9       String[] arr = str.split(" ");
10      int[] coinArray = new int[arr.length];
11      for (int i = 0; i < coinArray.length; i++) {
12        coinArray[i] = Integer.parseInt(arr[i]);
13      }
14      System.out.println("Результат: " + coins(coinArray));
15    }
16
17    public static int coins(int[] piles) {
18      if (piles.length % 3 != 0) {
19        throw new IllegalArgumentException();
20      }
21      piles = Matrix.quickSort(piles);
22      if (piles[0] < 1) {
23        throw new IllegalArgumentException();
24      }
25      int sum = 0;
26      for (int i = piles.length - 2; i >= piles.length / 4; i -= 2) {
27        sum += piles[i];
28      }
29      return sum;
30    }
31  }
```

3. ВЫВОД

В результате выполнения представленных заданий на языке программирования Java реализованы алгоритмы, позволяющие решать следующие задачи: нахождение треугольника с максимальным периметром среди массива чисел, составление максимального числа из массива, сортировка диагоналей матрицы, задача о шариках и стрелах, задача о перестановках строк, нахождение самой длинной палиндромной подстроки и задача о количестве подстрок, представляемых в виде конкатенации некоторой строки с самой собой.