

Министерство цифрового развития, связи и массовых коммуникаций  
Российской Федерации

Ордена Трудового Красного Знамени

«Московский технический университет связи и информатики»

Кафедра «Математической кибернетики и информационных технологий»

дисциплина «Структуры и алгоритмы обработки данных»

Отчет по лабораторной работе №1

“Работа с сортировками”

Выполнил: студенты группы БФИ1901

Козырев Сергей Владимирович

Проверил:

Кутейников Иван Алексеевич

Москва 2021

## Оглавление

1.Задание на лабораторную работу. ....	3
2.Листинг программы (Lab1).....	4
2.1 Листинг программы Lab1.....	4
3.Вывод .....	10
Список использованных источников .....	11

## 1.Задание на лабораторную работу.

Задание на лабораторную работу показано на рисунках 1-2.

### Задание №1:


1. Создать Jupyter Notebook со следующим наименованием: Lab1\_Группа\_ФИО
2. Создать новую ячейку с помощью кнопки 
3. В созданной ячейке по указанной ниже форме заполните оглавление файла, заменив наименование группы и вписав свое ФИО,



Рисунок 1 - Форма оглавления файла в ячейке

после чего создайте еще одну ячейку и напишите следующий код:

```
print("Hello, World!")
```


4. С помощью кнопки  запустите выполнение всех ячеек.
5. После выполнения у вас должна отформатироваться ячейка с оглавлением и должен выполняться "Hello, World!" (см. рис. 2).

Рисунок 1 – Задание на работу.

### Задание №2:

Написать генератор случайных матриц(многомерных), который принимает опциональные параметры **m**, **n**, **min\_limit**, **max\_limit**, где **m** и **n** указывают размер матрицы, а **min\_lim** и **max\_lim** - минимальное и максимальное значение для генерируемого числа . По умолчанию при отсутствии параметров принимать следующие значения:

$$m = 50$$

$$n = 50$$

$$\text{min\_limit} = -250$$

$$\text{max\_limit} = 1000 + (\text{номер своего варианта})$$

### Задание №3:

Реализовать методы сортировки строк числовой матрицы в соответствии с заданием. Оценить время работы каждого алгоритма сортировки и сравнить его со временем стандартной функции сортировки. Испытания проводить на сгенерированных матрицах.

Методы:

Выбором	Вставкой	Обменом	Шелла	Турнирная	Быстрая сортировка	Пирамидальная
---------	----------	---------	-------	-----------	--------------------	---------------

## Рисунок 2 – Задание на работу.

### 2.Листинг программы (Lab1)

#### 2.1 Листинг программы Lab1.

```
package Lab1;

import java.util.Scanner;

public class Lab1 {
    public static void main(String[] args) {
        System.out.println("Задание 1");
        System.out.println("Hello world!");
        Lab1 s = new Lab1();
        s.Zadanie2_3();
    }

    public void Zadanie2_3() {
        System.out.println("Задание 2");
        Scanner sc = new Scanner(System.in);
        System.out.println("Введите длину массива");
        String m1 = sc.nextLine();
        System.out.println("Введите ширину массива");
        String n1 = sc.nextLine();
        System.out.println("Введите минимальный предел массива");
        String min_lim1 = sc.nextLine();
        System.out.println("Введите максимальный2 предел массива");
        String max_lim1 = sc.nextLine();

        if (n1.equals(""))
            n1 = "50";
        if (m1.equals(""))
            m1 = "50";
        if (min_lim1.equals(""))
            min_lim1 = "-250";
        if (max_lim1.equals(""))
            max_lim1 = "1014";
        int n = Integer.parseInt(n1);
        int m = Integer.parseInt(m1);
        int min_lim = Integer.parseInt(min_lim1);
        int max_lim = Integer.parseInt(max_lim1);
        int[][] arr = new int[n][m];
        int max = 0, index = 0;
        int min = 0;
        for (int i = 0; i < n; i++) {
            System.out.print("\n");
            for (int j = 0; j < m; j++) {
                arr[i][j] = (int) ((Math.random() * (max_lim - min_lim)) + min_lim);
```

```

        System.out.print(arr[i][j] + "\t");
    }
    System.out.println();
}
System.out.println("Задание 3");
System.out.println("Выбором");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        min = arr[i][j];
        index = j;
        for (int c = j+1; c < m; c++) {
            if (arr[i][c] < min) {
                min = arr[i][c];
                index = c;
            }
        }
        if (j != index) {
            int zero = arr[i][j];
            arr[i][j] = min;
            arr[i][index] = zero;
        }
    }
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        System.out.print(arr[i][j] + "\t");
    }
    System.out.println();
}
System.out.println("Вставкой");
for (int i = 0; i < n; i++) {
    for (int j = 1; j < m; j++) {
        for (int c = j; c > 0 && arr[i][c - 1] > arr[i][c]; c--) {
            int z = arr[i][c];
            arr[i][c] = arr[i][c - 1];
            arr[i][c - 1] = z;
        }
    }
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        System.out.print(arr[i][j] + "\t");
    }
    System.out.println();
}
System.out.println("Обменом");
for (int i = 0; i < n; i++) {
    boolean needIteration = true;
    while (needIteration) {
        needIteration = false;
        for (int j = 1; j < m; j++) {

```

```

        if (arr[i][j] < arr[i][j - 1]) {
            int z = arr[i][j];
            arr[i][j] = arr[i][j - 1];
            arr[i][j - 1] = z;
            needIteration = true;
        }
    }
}

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

System.out.println("Шелла");
int d = m / 2;
for (int i = 0; i < n; i++) {
    while (d > 0) {
        for (int j = 0; j < m - d; j++) {
            int q = j;
            while (q >= 0 && arr[i][q] > arr[i][q + d]) {
                int temp = arr[i][q];
                arr[i][q] = arr[i][q + d];
                arr[i][q + d] = temp;
                q--;
            }
        }
        d = d / 2;
    }
}

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

System.out.println("Пирамидальная");
int[] arr1 = new int[m];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        arr1[j] = arr[i][j];
    }

    heapSort(arr1);

    for (int l = 0; l < m; l++) {
        System.out.print(arr1[l] + "\t");
    }
    System.out.println();
}

```

```

    }
    System.out.println("Быстрая сортировка");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            arr1[j] = arr[i][j];
        }

        quickSort(arr1, 0, m - 1);

        for (int l = 0; l < m; l++) {
            System.out.print(arr1[l] + "\t");
        }
        System.out.println();
    }
    System.out.println("Турнирная сортировка");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            arr1[j] = arr[i][j];
        }

        Sort(arr1);

        for (int l = 0; l < m; l++) {
            System.out.print(arr1[l] + "\t");
        }
        System.out.println();
    }
}

static void heapify(int[] array, int length, int i) {
    int leftChild = 2 * i + 1;
    int rightChild = 2 * i + 2;
    int largest = i;

    if (leftChild < length && array[leftChild] > array[largest]) {
        largest = leftChild;
    }

    if (rightChild < length && array[rightChild] > array[largest]) {
        largest = rightChild;
    }

    if (largest != i) {
        int temp = array[i];
        array[i] = array[largest];
        array[largest] = temp;
        heapify(array, length, largest);
    }
}

```

```

}

public static void heapSort(int[] array) {
    if (array.length == 0) return;
    int length = array.length;
    for (int i = length / 2 - 1; i >= 0; i--)
        heapify(array, length, i);
    for (int i = length - 1; i >= 0; i--) {
        int temp = array[0];
        array[0] = array[i];
        array[i] = temp;
        heapify(array, i, 0);
    }
}

static int partition(int[] array, int begin, int end) {
    int pivot = end;

    int counter = begin;
    for (int i = begin; i < end; i++) {
        if (array[i] < array[pivot]) {
            int temp = array[counter];
            array[counter] = array[i];
            array[i] = temp;
            counter++;
        }
    }
    int temp = array[pivot];
    array[pivot] = array[counter];
    array[counter] = temp;

    return counter;
}

public static void quickSort(int[] array, int begin, int end) {
    if (end <= begin) return;
    int pivot = partition(array, begin, end);
    quickSort(array, begin, pivot - 1);
    quickSort(array, pivot + 1, end);
}

private class Node
{
    public int data;
    public int id;

    public Node()
    {

    }

    public Node(int _data, int _id)//
    {

```



```

        data = _data;
        id = _id;
    }
}

public void Adjust(Node[] data, int idx)
{
    while(idx != 0)
    {
        if(idx % 2 == 1)
        {
            if(data[idx].data < data[idx + 1].data)
            {
                data[(idx - 1)/2] = data[idx];
            }
            else
            {
                data[(idx-1)/2] = data[idx + 1];
            }
            idx = (idx - 1)/2;
        }
        else
        {
            if(data[idx-1].data < data[idx].data)
            {
                data[idx/2 - 1] = data[idx-1];
            }
            else
            {
                data[idx/2 - 1] = data[idx];
            }
            idx = (idx/2 - 1);
        }
    }
}

public void Sort(int[] data)
{
    int nNodes = 1;
    int nTreeSize;
    while(nNodes < data.length)
    {
        nNodes *= 2;
    }
    nTreeSize = 2 * nNodes - 1;

    Node[] nodes = new Node[nTreeSize];

    int i, j;
    int idx;

```

```

        for( i = nNodes - 1; i < nTreeSize; i++)
        {
            idx = i - (nNodes - 1);
            if(idx < data.length)
            {
                nodes[i] = new Node(data[idx], i);
            }
            else
            {
                nodes[i] = new Node(Integer.MAX_VALUE, -1);
            }
        }

        for( i = nNodes - 2; i >= 0; i--)
        {
            nodes[i] = new Node();
            if(nodes[i * 2 + 1].data < nodes[i * 2 + 2].data)
            {
                nodes[i] = nodes[i*2 + 1];
            }
            else
            {
                nodes[i] = nodes[i*2 + 2];
            }
        }
        for( i = 0; i < data.length; i++)
        {
            data[i] = nodes[0].data;
            nodes[nodes[0].id].data = Integer.MAX_VALUE;
            Adjust(nodes, nodes[0].id);
        }
    }
}

```

### 3.Вывод

Выполнив данную лабораторную работу, мы научились работать с методами сортировок и применять их в поставленных задачах, а именно, реализовывать сортировку числовых матриц.

## **Список использованных источников**

1 ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.

2 ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления.