Федеральное агентство связи
Ордена Трудового Красного Знамени федеральное государственное
бюджетное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и
информационных технологий

Лабораторная работа №1

по дисциплине: «Структуры и алгоритмы обработки данных»

на тему: «Методы сортировки»

Выполнил студент
группы БФИ1902
Гусев Н. С.
Проверил:
Мкртчян Г. М.

Москва, 2020 г.

Оглавление

1. Цель работы 3
2. Задание на лабораторную работу 3
3. Листинг программы 3

1. Цель работы

Цель работы: рассмотреть работу методов сортировки строк числовой матрицы: Выбором, вставкой, обменом, Шелла, турнирная, быстрая сортировка, пирамидальная.

2. Задание на лабораторную работу

1) Написать программу, которая выводит текст «Hello, world!»;
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.
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.
4) Создать публичный репозиторий на GitHub и запушить выполненное задание.

3. Листинг программы

```
import java.util.Arrays;

public class Main {

    public static int m = 20; // строки
    public static int n = 20; // столбцы
    public static int min_lim = -250; // Минимальный элемент
    public static int max_lim = 1003; // Максимальный элемент

    public static void main(String[] args) {
       System.out.println("1) Hello World!");
        System.out.println();

        int[][] a = new int[m][n];
        Random(a);

        System.out.println("2) Матрица до преобразований");
        showMatrix(a);
        System.out.println();

        System.out.println("3) Системная сортировка:");
        long start0 = System.nanoTime();
        for(int i = 0; i < n; i++) {
            int[] c = a[i];
```

```
    Arrays.sort(c);
    System.out.println(Arrays.toString(c));
}
long finish0 = System.nanoTime();
long elapsed0 = finish0 - start0;
System.out.println("Прошло времени, нс: " + elapsed0);
System.out.println();

System.out.println("Сортировка выбором:");
long start1 = System.nanoTime();
for (int i = 0; i < n; i++)
    selectionSort(a[i]);
long finish1 = System.nanoTime();
long elapsed1 = finish1 - start1;
System.out.println("Прошло времени, нс: " + elapsed1);
System.out.println();

System.out.println("Сортировка вставками:");
long start2 = System.nanoTime();
for (int i = 0; i < n; i++)
    InsertionSort(a[i]);
long finish2 = System.nanoTime();
long elapsed2 = finish2 - start2;
System.out.println("Прошло времени, нс: " + elapsed2);
System.out.println();

System.out.println("Сортировка обменом:");
long start3 = System.nanoTime();
for (int i = 0; i < n; i++)
    bubbleSort(a[i]);
long finish3 = System.nanoTime();
long elapsed3 = finish3 - start3;
System.out.println("Прошло времени, нс: " + elapsed3);
System.out.println();

System.out.println("Сортировка Шелла:");
long start4 = System.nanoTime();
for (int i = 0; i < n; i++)
    Shell(a[i]);
long finish4 = System.nanoTime();
long elapsed4 = finish4 - start4;
System.out.println("Прошло времени, нс: " + elapsed4);
System.out.println();

System.out.println("Быстрая сортировка:");
long start5 = System.nanoTime();
for (int i = 0; i < n; i++) {
    quicksort(a[i], 0, m - 1);
    System.out.println(Arrays.toString(a[i]));
}
long finish5 = System.nanoTime();
long elapsed5 = finish5 - start5;
System.out.println("Прошло времени, нс: " + elapsed5);
System.out.println();

System.out.println("Пирамидальная сортировка:");
long start6 = System.nanoTime();
for (int i = 0; i < n; i++)
    heapSort(a[i]);
long finish6 = System.nanoTime();
long elapsed6 = finish6 - start6;
System.out.println("Прошло времени, нс: " + elapsed6);
System.out.println();
```

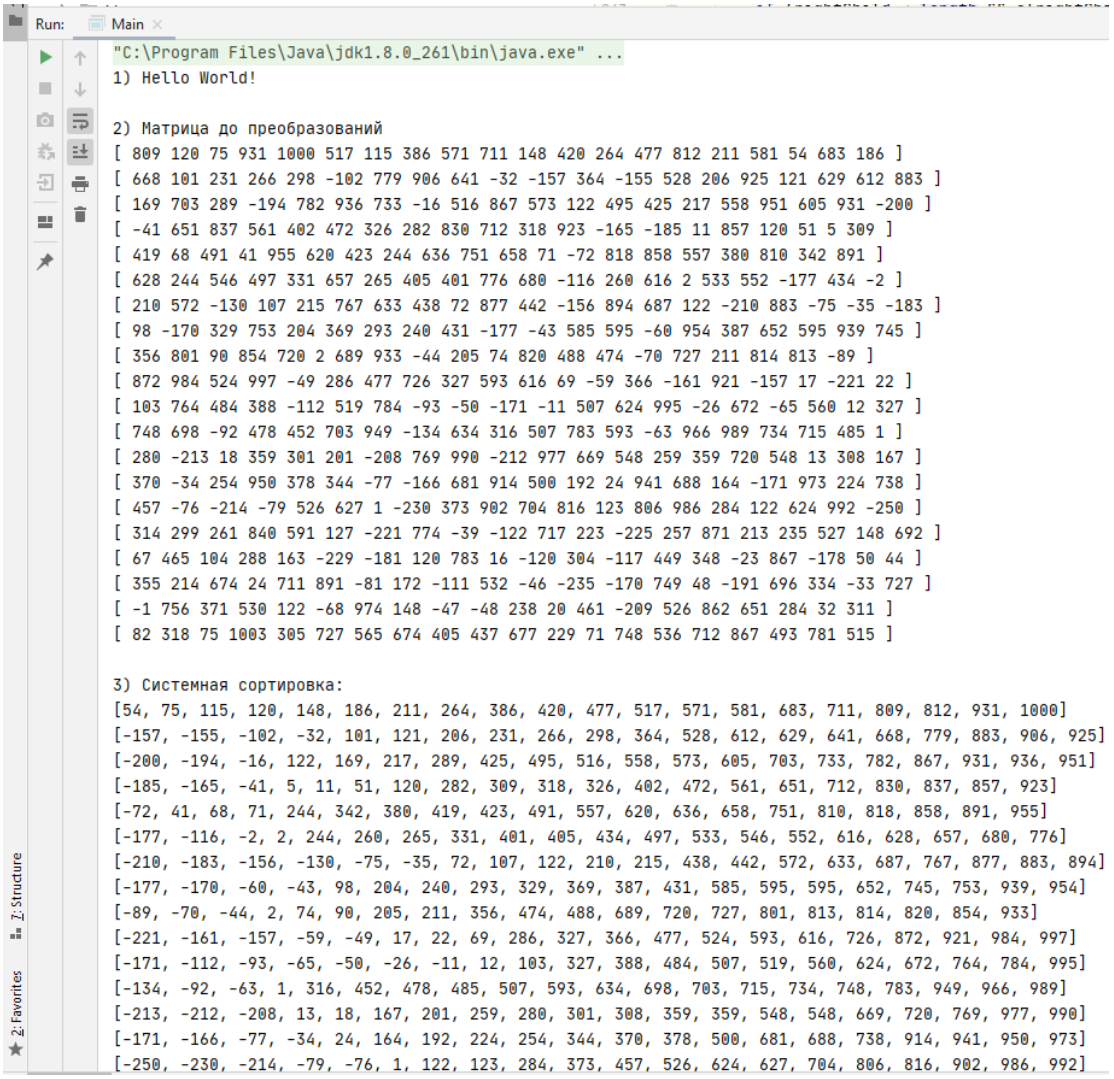
```
        if (elapsed1 <= elapsed2 && elapsed1 <= elapsed3 && elapsed1 <=
elapsed4 && elapsed1 <= elapsed5 && elapsed1 <= elapsed6 && elapsed1 <=
elapsed0) {
            System.out.println("Сортировка выбором - самая быстрая
сортировка");
        } else if (elapsed2 <= elapsed1 && elapsed2 <= elapsed3 && elapsed2
<= elapsed4 && elapsed2 <= elapsed5 && elapsed2 <= elapsed6 && elapsed2 <=
elapsed0) {
            System.out.println("Сортировка вставками - самая быстрая
сортировка");
        } else if (elapsed3 <= elapsed2 && elapsed3 <= elapsed1 && elapsed3
<= elapsed4 && elapsed3 <= elapsed5 && elapsed3 <= elapsed6 && elapsed3 <=
elapsed0) {
            System.out.println("Сортировка обменом - самая быстрая
сортировка");
        } else if (elapsed4 <= elapsed1 && elapsed4 <= elapsed2 && elapsed4
<= elapsed3 && elapsed4 <= elapsed5 && elapsed4 <= elapsed6 && elapsed4 <=
elapsed0) {
            System.out.println("Сортировка Шелла - самая быстрая
сортировка");
        } else if (elapsed5 <= elapsed1 && elapsed5 <= elapsed2 && elapsed5
<= elapsed3 && elapsed5 <= elapsed4 && elapsed5 <= elapsed6 && elapsed5 <=
elapsed0) {
            System.out.println("Быстрая сортировка - самая быстрая
сортировка");
        } else if (elapsed6 <= elapsed1 && elapsed6 <= elapsed2 && elapsed6
<= elapsed3 && elapsed6 <= elapsed4 && elapsed6 <= elapsed5 && elapsed6 <=
elapsed0) {
            System.out.println("Пирамидальная сортировка - самая быстрая
сортировка");
        } else {
            System.out.println("Системная сортировка - самая быстрая
сортировка");
        }

    }

    // Генерация случайных чисел
    public static void Random(int[][] a) {
        for(int i = 0; i < m; i++) {
            for(int j = 0; j < n; j++) {
                a[i][j] =  min_lim + (int)(Math.random()*((max_lim - min_lim)
+ 1));
            }
        }
    }

    // Вывод матрицы
    public static void showMatrix(int[][] a){
        for(int i=0; i < m; i++){
            System.out.print("[ ");
            for(int j=0; j < n; j++){
                System.out.print(a[i][j] + " ");
            }
            System.out.print("]");
            System.out.println();
        }
    }

    // Сортировка вставкой
    public static void InsertionSort(int[] array)
    {
        int[] b = array;
        for (int i = 1; i < array.length; i++)
```

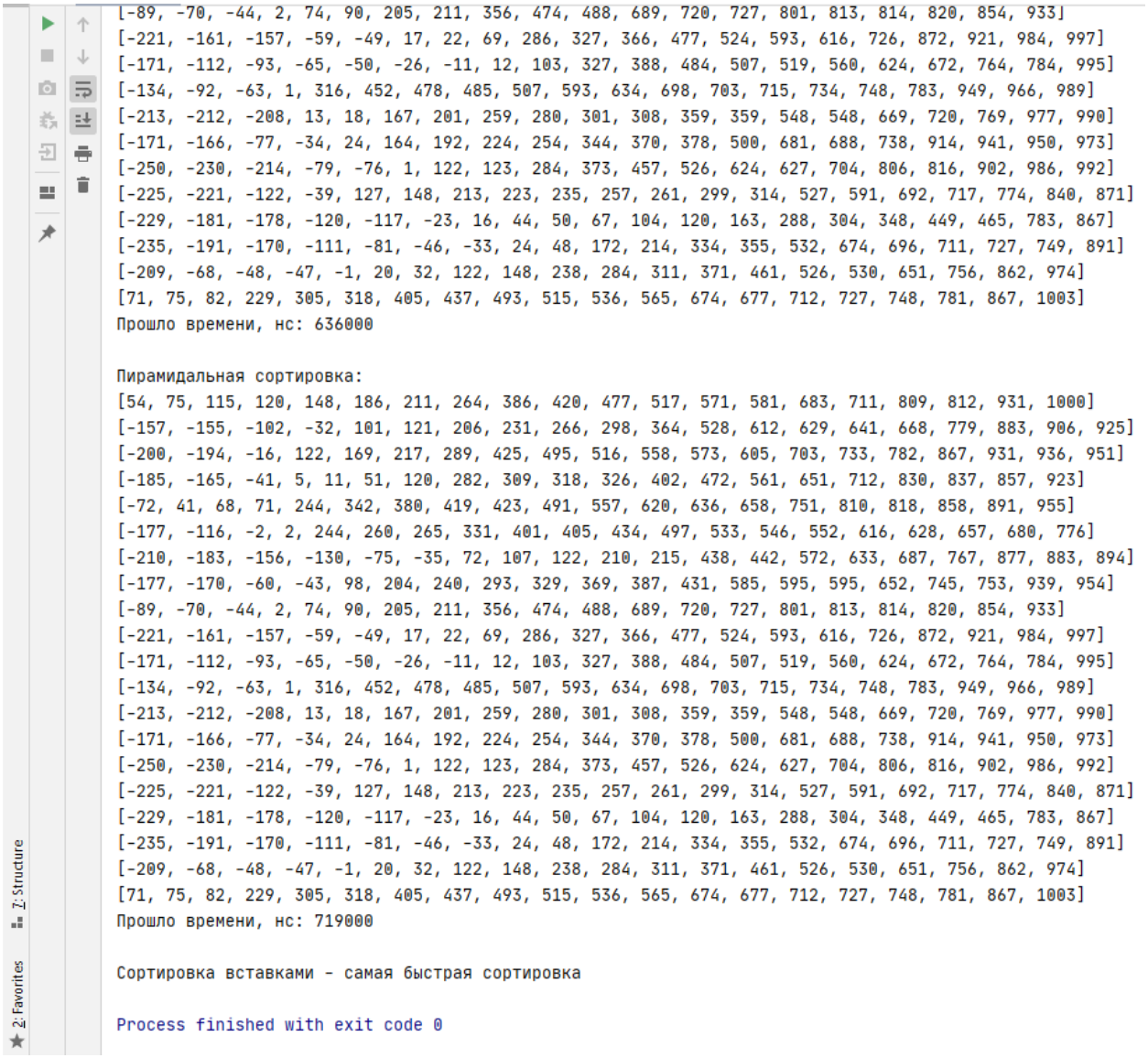
```
        {
            int j;
            int buf = b[i];
            for (j = i - 1; j >= 0; j--)
            {
                if (b[j] < buf)
                    break;
                b[j + 1] = b[j];
            }
            b[j + 1] = buf;
        }
        System.out.println(Arrays.toString(b));
    }


    // Сортировка выбором
    public static void selectionSort(int[] a) {
        int[] b = a;
        int tmp;
        for(int i = 0; i < m; i++)
        {
            int pos = i;
            tmp = b[i];
            for(int j = i + 1; j < m; j++)
            {
                if (b[j] < tmp)
                {
                    pos = j;
                    tmp = b[j];
                }
            }
            b[pos] = b[i];
            b[i] = tmp;
        }
        System.out.println(Arrays.toString(b));
    }

    // Сортировка обменом, или пузырьком
    public static void bubbleSort(int[] a) {
        int[] b = a;
        boolean sorted = false;
        int temp;
        while(!sorted) {
            sorted = true;
            for (int i = 0; i < b.length - 1; i++) {
                if (b[i] > b[i+1]) {
                    temp = b[i];
                    b[i] = b[i+1];
                    b[i+1] = temp;
                    sorted = false;
                }
            }
        }
        System.out.println(Arrays.toString(b));
    }

    // Сортировка Шелла
    public static void Shell(int[] a) {
        int[] b = a;
        for (int step = n / 2; step > 0; step /= 2) {
            for (int i = step; i < n; i++) {
                for (int j = i - step; j >= 0 && b[j] > b[j + step]; j -=
step) {
                    int x = b[j];
```

```
                b[j] = b[j + step];
                b[j + step] = x;
            }
        }
    }
    System.out.println(Arrays.toString(b));
}

// Быстрая сортировка
public static int partition (int[] a, int start, int end)
{
    int marker = start;
    for ( int i = start; i <= end; i++ )
    {
        if (a[i] <= a[end] )
        {
            int temp = a[marker]; // swap
            a[marker] = a[i];
            a[i] = temp;
            marker += 1;
        }
    }
    return marker - 1;
}

public static void quicksort(int[] a, int start, int end) {
    int[] b = a;
    if ( start >= end ) {
        return;
    }
    int pivot = partition (b, start, end);
    quicksort (b, start, pivot-1);
    quicksort (b, pivot+1, end);
}

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

    // если левый дочерний больше родительского
    if (leftChild < length && a[leftChild] > a[largest]) {
        largest = leftChild;
    }

    // если правый дочерний больше родительского
    if (rightChild < length && a[rightChild] > a[largest]) {
        largest = rightChild;
    }

    // если должна произойти замена
    if (largest != i) {
        int temp = a[i];
        a[i] = a[largest];
        a[largest] = temp;
        heapify(a, length, largest);
    }
}
public static void heapSort(int[] a) {
    int[] b = a;
    // Строим кучу
    int length = b.length;
    // проходим от первого без ответвлений к корню
```

```
        for (int i = length / 2-1; i >= 0; i--)
            heapify(b, length, i);
        for (int i = length-1; i >= 0; i--) {
            int temp = b[0];
            b[0] = b[i];
            b[i] = temp;
            heapify(b, i, 0);
        }
        System.out.println(Arrays.toString(b));
    }
}
```

4. Результат работы программы





5. Вывод

Я рассмотрел различные методы сортировки строк числовой матрицы.