

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации

государственное бюджетное образовательное учреждение

высшего профессионального образования

ордена Трудового Красного Знамени

“Московский технический университет связи и информатики”

Лабораторная работа №1 по дисциплине

“СИАОД”

Выполнил студент

Группы БВТ1902

Магомедов А.М.

Москва 2021

Оглавление

1. Задание на лабораторную работу	3
2. Листинг программы	3
3. Вывод.....	8

1. Задание на лабораторную работу

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

min_limit = -250

max_limit = 1000 + (номер своего варианта)

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
---------	----------	---------	-------	-----------	--------------------	---------------

2. Листинг программы

```
package com.company;

import java.util.Random;
import java.util.Scanner;

public class Main {

    static int[][] array;

    public static void quickSort(int[] array, int low, int high) {
        if (array.length == 0)
            return; //завершить выполнение, если длина массива равна 0

        if (low >= high)
            return; //завершить выполнение если уже нечего делить

        // выбрать опорный элемент
        int middle = low + (high - low) / 2;
        int opora = array[middle];

        // разделить на подмассивы, который больше и меньше опорного
        // элемента
        int i = low, j = high;
        while (i <= j) {
            while (array[i] < opora) {
                i++;
            }
            while (array[j] > opora) {
                j--;
            }
            if (i < j) {
                int temp = array[i];
                array[i] = array[j];
                array[j] = temp;
            }
        }
        quickSort(array, low, j);
        quickSort(array, j + 1, high);
    }
}
```

```

        while (array[j] > opora) {
            j--;
        }

        if (i <= j) { //меняем местами
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
            i++;
            j--;
        }
    }

    // вызов рекурсии для сортировки левой и правой части
    if (low < j)
        quickSort(array, low, j);

    if (high > i)
        quickSort(array, i, high);
}

static void heapify(int[] array, 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 && array[leftChild] > array[largest]) {
        largest = leftChild;
    }

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

    // если должна произойти замена
    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;

    // Строим кучу
    int length = array.length;
    // проходим от первого без ответвлений к корню
    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);
    }
}

```

```

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

    // вторая задача
    Scanner in = new Scanner(System.in);

    System.out.println("Введите количество строк: ");
    int m = in.nextInt();
    System.out.println("Введите количество столбцов: ");
    int n = in.nextInt();
    array = new int[m][n];
    generateMass(m, n);
    Vivod(m, n);
    System.out.println("Сортировка вставкой:");
    Sort_vstavkoi(m, n);
    Vivod(m, n);

    System.out.println("Сортировка обменом:");
    Sort_puzirkom(m, n);
    Vivod(m, n);

    System.out.println("Сортировка выбором:");
    Sort_vibor(m, n);
    Vivod(m, n);

    System.out.println("Сортировка шелла:");
    Sort_Shell(m, n);
    Vivod(m, n);

    System.out.println("Быстрая сортировка:");
    Sort_Fast();
    Vivod(m, n);

    System.out.println("Пирамидальная сортировка:");
    Sort_piramid();
    Vivod(m, n);
}

public static void generateMass(int m, int n){
    int MIN_LIMIT = -250;
    int MAX_LIMIT = 1000 + 3;
    Random random = new Random();
    for (int i=0; i<m; i++){
        for (int j=0; j<n; j++){
            array[i][j] = (int) (Math.random() * ((MAX_LIMIT -
MIN_LIMIT) + 1)) + MIN_LIMIT ;
        }
    }
}

public static void Sort_vstavkoi(int m, int n){
    int temp = 0;
    int count = 0;
    int lastElement = 0;
    for(int i = 0; i < m; i++) { // i - номер текущей строки
        for (int j = 1; j < n; j++) // j - номер текущего шага
        {
            temp = array[i][j];
            count = j-1;
            while (temp<array[i][count]){
                lastElement = array[i][count];
                array[i][count] = temp;
            }
        }
    }
}

```

```

        array[i][j] = lastElement;
        count--;
        if(count < 0){
            break;
        }
    }
}

}

}

}

public static void Sort_puzirkom(int m, int n){
    int temp = 0;
    for (int h=0; h<(m * n - 1); h++)    //сдвиги очередных
элементов в правильную позицию
    /*сдвиг элемента массива в правильную позицию*/
    for (int i = 0; i<m ; i++){
        for (int j = 0; j<n-1; j++){
            /*АНАЛИЗ НА ПОСЛЕДНИЙ ЭЛЕМЕНТ МАССИВА*/
            if (i== m-1 && j== n-1){    //Если строка последняя и
справа тупик, то ничего не делаем
                continue;
            }
            /*КОНЕЦ АНАЛИЗА НА ПОСЛЕДНЮЮ СТРОКУ*/

            if (array[i][j] > array[i][j+1]){ //Если элемент не
на своей позиции
                temp = array[i][j];    //Обмен местами
                array[i][j] = array[i][j+1];
                array[i][j+1] = temp;
            }
        }
    }
}

}

public static void Sort_vibor(int m, int n){
    int temp = 0;
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n - 1; j++)
        {
            for (int k = j + 1; k < n; k++)
            {
                if (array[i][j] < array[i][k] )
                    temp = array[i][j];
                array[i][j] = array[i][k];
                array[i][k] = temp;
            }
        }
    }
}

}

public static void Sort_Shell(int m, int n) {
    int d = 0;
    for(int i = 0; i<m;i++){
        d = n/2;
        while (d>0){
            for(int j = 0; j< n - d; j++){
                int k = j;
                while ((k>=0) && (array[i][k] > array[i][k+d])){
                    int temp = array[i][k];
                    array[i][k] = array[i][k+d];
                    array[i][k+d] = temp;
                    k--;
                }
            }
        }
    }
}

```

```

        }
        d = d/2;
    }
}

public static void Sort_Fast(){
    for(int i = 0; i<array.length; i++){//Перебираем массив
        for(int j = 0; j<array[i].length; j++){//Перебираем массив
            for (int s = 0; s<array[i].length; s++){//Перебираем
строку
                int low = 0;
                int high = array[i].length-1;
                // цикл выбора наименьшего элемента
                for (int z = array[i].length-1; z>s; z--
) {//Перебираем строку
                    quickSort(array[i],low,high);
                }
            }
        }
    }

    public static void Sort_piramid() {
        for(int i = 0; i<array.length; i++){//Перебираем массив
            for(int j = 0; j<array[i].length; j++){//Перебираем массив
                for (int s = 0; s<array[i].length; s++){//Перебираем
строку
                    for (int z = array[i].length-1; z>s; z--
) {//Перебираем строку
                        heapSort(array[i]);
                    }
                }
            }
        }

        public static void Vivod(int m, int n){           // Вывод двумерного
массива
            for (int i=0; i<m; i++){
                for (int j=0; j<n; j++){
                    System.out.print(" " + array[i][j] + " ");
                }
                System.out.println();
            }
            System.out.println();
        }
    }
}

```

3. Вывод

Мы реализовали метод генерации массива, а также алгоритмы указанных в условии сортировок.