ФГБОУ ВО

ННГАСУ

Кафедра информационных систем и технологий

Расчетно-графическая работа

Разработка в среде OpenSource

**Разработка приложения сортировки и поиска**

Выполнила студентка 3 курса группы ИС-29

Хазова А.С. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Проверил старший преподаватель

Лахов А. Я. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Нижний Новгород

2023 год

Содержание

[Задание 3](#_Toc133522229)

[Алгоритм метода сортировки 4](#_Toc133522230)

[Алгоритм метода поиска 5](#_Toc133522231)

[Код на языке Java 6](#_Toc133522232)

[Сеанс работы с приложением 13](#_Toc133522233)

# Задание

Разработать консольное приложение на Java, генерирующее файл Data.txt со случайными целыми числами.

Разработать графическое приложение на Java, реализующее функции:

а) чтение содержания файла Data.txt (название файла вводится через JOptionPane.showInputDialog), запись значений в массив (например, int a []), вывод введенных значений в TextArea,

б) выполнение имитации сортировки значений массива заданным методом, вывод отсортированных значений в TextArea,

в) выполнение имитации поиска заданного значения int Key (значение Key вводится через JOptionPane.showInputDialog) заданным методом, найденный индекс, при успешности поиска, выводится в метку Label, при неудаче выводится сообщение Not Found.

***Вариант 8.***

*Задание 1.* Выполнение сортировки значений массива методом вставки.

*Задание 2.* Выполнение поиска заданного значения int Key методом бинарного поиска.

# Алгоритм метода сортировки

Сортировка массивов, в данной работе, происходит методом вставки.

Сортировка вставками - алгоритм, при котором каждый последующий элемент массива сравнивается с предыдущими элементами(отсортированными) и вставляется в нужную позицию.

На первом шаге сортируются два первых элемента. Затем на свое место среди них вставляется третий элемент. К трем упорядоченным добавляется четвертый, который занимает свое место в четверке и т. д. Для этого на каждом шаге алгоритма мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированной части массива, до тех пор пока весь набор входных данных не будет отсортирован. Метод выбора очередного элемента из исходного массива произволен, однако обычно (и с целью получения устойчивого алгоритма сортировки), элементы вставляются по порядку их появления во входном массиве.

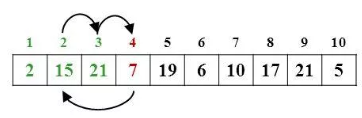


Рис.1 – Наглядное представление сортировки массива методом вставки.

Так как в процессе работы алгоритма могут меняться местами только соседние элементы, каждый обмен уменьшает число [инверсий](https://neerc.ifmo.ru/wiki/index.php?title=%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D0%B8%D0%BD%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9) на единицу. Следовательно, количество обменов равно количеству инверсий в исходном массиве вне зависимости от реализации сортировки. Максимальное количество инверсий содержится в массиве, элементы которого отсортированы не по возрастанию. Число инверсий в таком массиве n\*(n-1)/2.

# Алгоритм метода поиска

Алгоритм поиска выполняется с помощью методом бинарного поиска.

Бинарным (или двоичным) называют поиск элемента упорядоченного множества через многократное деление этого множества пополам. Искомый элемент всегда будет оказываться в одной из двух частей. Поиск прекращается, когда обнаруживается совпадение граничного элемента между двумя разделенными блоками с заданным, или когда заданный элемент не обнаруживается вовсе.

Реализация этого метода возможна только применимо к отсортированным множествам.

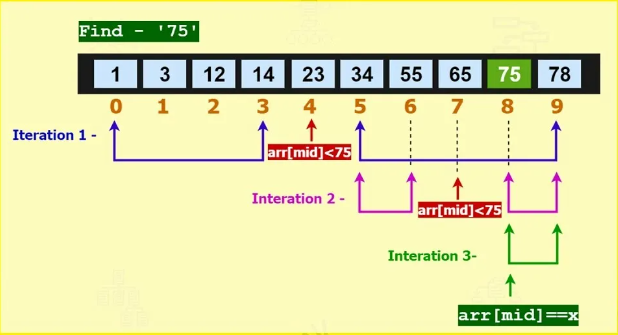


Рис.2 – Наглядное представление бинарного поиска

# Код на языке Java

Реализация консольного приложения для создания текстового файла Data.txt и записи в него массива из двадцати случайных чисел:

//CreateFile.java

import java.io.File;

import java.io.\*;

import java.util.\*;

public class CreateFile {

    public static void main(String[] args)throws IOException {

        int[] a = new int[20];

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

            a[i] = (int)(Math.random() \* 1001);

            System.out.println(a[i]); }

        File file = new File("Data.txt");

        file.createNewFile();

        FileWriter writer = new FileWriter(file);

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

            final String s = Integer.toString(a[i]);

            writer.write(s);

            writer.write(System.lineSeparator()); }

        writer.close();

    }

}

Реализация графического приложения по сортировке и поиску элементов в массиве целых случайных чисел:

//DataInputOutput.java

import java.awt.\*;

import java.awt.event.\*;

import java.io.IOException;

import java.io.\*;

import javax.swing.\*;

import java.io.BufferedReader;

public class DataInputOutput extends JFrame implements ActionListener{

    int size = 20;

    int[] a5 = new int[size];

    private JTextArea txt = new JTextArea(4, 57);

    Label lbl = new Label("         Нажмите на меню      ", Label.CENTER);

    Label lbl3 = new Label("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_", Label.CENTER);

    public void actionPerformed(ActionEvent e){

        System.out.println("Menu event: "+e);

        String cmd = e.getActionCommand();

        if(cmd.equals("Open")){

            lbl.setText("        Нажата кнопка Open      ");

            String file="";

            file = JOptionPane.showInputDialog("Введите название файла с данными");

            System.out.println(file);

            try{

                FileInputStream fis = new FileInputStream(file);

                InputStreamReader isr = new InputStreamReader(fis);

                BufferedReader br = new BufferedReader(isr);

                String line;

                int i = 0;

                while((line = br.readLine()) != null){

                    a5[i] = Integer.parseInt(line);

                    i++;

                    System.out.println(line); }

                br.close();

            }

            catch(IOException el){

                el.printStackTrace();

            }

            txt.setLineWrap(true);

            txt.setWrapStyleWord(true);

            txt.append("Массив: ");

            for(int i = 0; i< a5.length; i++){

                txt.append(a5[i] + " ");

            }

            txt.append("\n");

        }

        else if (cmd.equals("Exit")){

            lbl.setText("       Нажата кнопка Exit        ");

            dispose();

            System.exit(0);

        }

        else if(cmd.equals("Sort")){

            lbl.setText("        Нажата кнопка Sort        ");

            txt.setLineWrap(true);

            txt.setWrapStyleWord(true);

            txt.append("Отсортированный массив: ");

            int in;

            for (int out = 1; out<a5.length; out++){

                int temp = a5[out];

                in = out;

                while(in>0 && a5[in-1] >= temp){

                    a5[in] = a5[in-1];

                    --in;

                }

                a5[in] = temp;

            }

            for(int i = 0; i<a5.length; i++){

                txt.append(a5[i] + " ");

            }

            txt.append("\n");

        }

        else if(cmd.equals("Find")){

            lbl.setText("        Нажата кнопка Find       ");

            String val="";

            val = JOptionPane.showInputDialog("Введите ключ поиска?");

            int val1 = Integer.parseInt(val);

            System.out.println(val1);

            int k=-1, low = 0, m;

            int high = a5.length - 1;

            while (low <= high){

                m = low + (high - low)/2;

                if (a5[m] < val1){

                    low = m +1;

                } else if (a5[m]>val1){

                    high = m -1;

                } else if (a5[m] == val1){

                    k = m;

                    break;

                }

            }

            if (k!=-1)

                lbl3.setText("Найден элемент "+val1+" с индексом " +k);

            else lbl3.setText("Данного значения нет в массиве");

        }

    }

    void Menu(){

        JMenu m1 = new JMenu("File");

        JMenuItem Item1 = new JMenuItem("Open");

        JMenuItem Item2 = new JMenuItem("Exit");

        m1.add(Item1);

        Item1.addActionListener(this);

        m1.add(Item2);

        Item2.addActionListener(this);

        JMenu m2 = new JMenu("Work");

        JMenuItem Item3 = new JMenuItem("Sort");

        JMenuItem Item4 = new JMenuItem("Find");

        m2.add(Item3);

        Item3.addActionListener(this);

        m2.add(Item4);

        Item4.addActionListener(this);

        JMenuBar mBar = new JMenuBar();

        mBar.add(m1);

        mBar.add(m2);

        setJMenuBar(mBar);

    }

    public DataInputOutput(){

        super("DataIntupOutput");

        Menu();

        setSize(660, 350);

        Container cp = getContentPane();

        JLabel lbl2 = new JLabel();

        lbl2.setText("<html>Расчетно-графическая работа Хазовой Александры Сергеевны ИС-29 Вариант 8 <br> Метод сортировки: вставка <br> Метод поиска: бинарный</html>");

        cp.add(lbl2);

        cp.setLayout(new FlowLayout(FlowLayout.CENTER));

        cp.add(lbl);

        cp.add(new JScrollPane(txt));

        cp.add(lbl3);

        setVisible(true);

    }

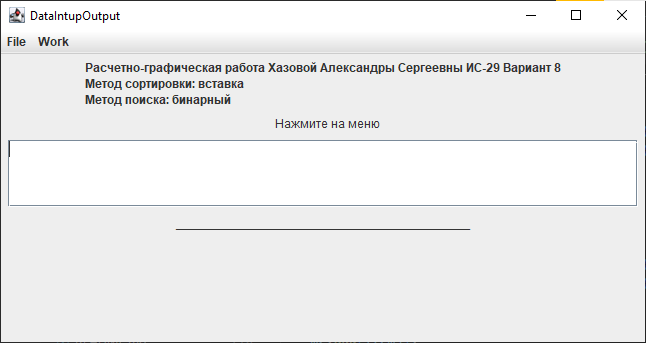
    public static void main(String[] args) throws Exception {

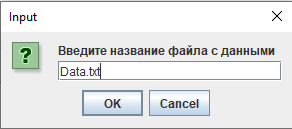
        new DataInputOutput();

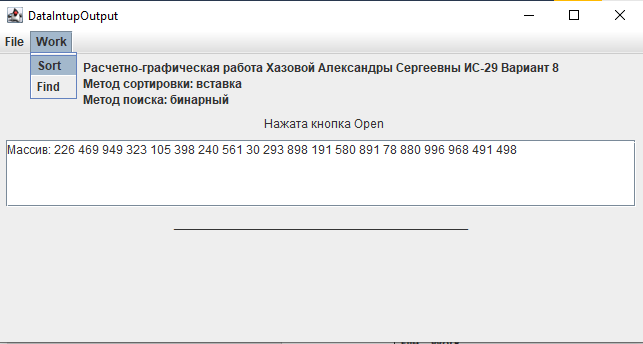
    }

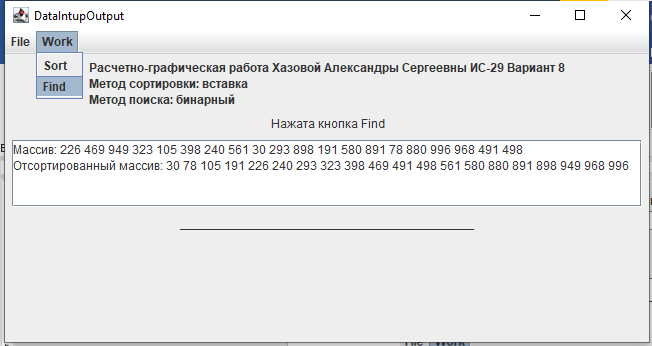
}

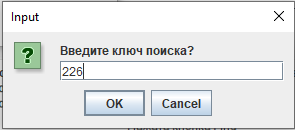
# Сеанс работы с приложением









’

