Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ярославский государственный технический университет»

Кафедра «Информационные системы и технологии»

Отчет защищен

с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель

\_\_\_\_\_\_\_\_\_\_ Д.В. Дидковская

«\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2022

**Работа с массивами. поиск элемента. сортировка.**

Отчёт о лабораторной работе №2 по курсу “Информационные технологии”

ЯГТУ 09.03.02-024 ЛР

Отчет выполнил

студент группы ЭИС-26

\_\_\_\_\_\_\_\_\_А.А. Хрящев

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2022

2022

Задание:

1. Даны два одномерных целочисленных массива. Произвести поиск заданного значения в первом из них – методом последовательного перебора с использованием барьерного элемента, а во втором – бинарный поиск, предварительно отсортировав этот массив методом вставки. Первый массив отсортировать затем выбором наименьшего элемента. Произвести слияние полученных массивов (см. файл «К заданию I (варианты заданий)»)

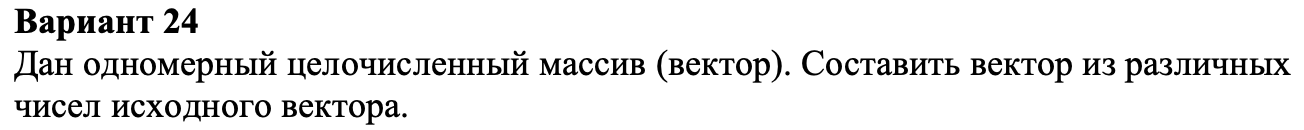


Рисунок 1 – Задание 1

2. Дано слово. Произвести сортировку данного слова методом пузырька с использованием индекса.

Код программы:

|  |
| --- |
| import java.util.Random;  import java.io.Serializable;  import java.util.Arrays;  import lombok.NonNull;  interface PrintIterator {  void print(String text);  }  class UtilArray {  public static int barrierElement(int @NonNull[] arr, int value) {  final int size = arr.length;  if (size != 0) {  int last = arr[size - 1];//Сохраним прежний элемент массива  arr[size - 1] = value;//Гарантируем, что value есть в массиве  //Есть гарантия того, что элемент есть в массиве, значит индекс можно не проверять  int i = 0;  for (i = 0; arr[i] != value; ++i) {//Одно условие в цикле  }  arr[size - 1] = last;//Восстанавливаем последний элемент  if (i != (size - 1) || value == last) {//Не уткнулись в барьер или последний элемент был искомым  return i;  }  }  return -1;  }  public static void insertionSort(int @NonNull [] arrayPtr) // сортировка вставками  {  int temp;// временная переменная для хранения значения элемента сортируемого массива  // индекс предыдущего элемента  int item;  for (int counter = 1; counter < arrayPtr.length; counter++)  {  temp = arrayPtr[counter]; // инициализируем временную переменную текущим значением элемента массива  item = counter-1; // запоминаем индекс предыдущего элемента массива  while(item >= 0 && arrayPtr[item] > temp) // пока индекс не равен 0 и предыдущий элемент массива больше текущего  {  arrayPtr[item + 1] = arrayPtr[item]; // перестановка элементов массива  arrayPtr[item] = temp;  item--;  }  }  }  }  class Work2 {  private PrintIterator printIterator;  private Work2(PrintIterator printStream) {  this.printIterator = printStream;  }  public static Work2 printAction(PrintIterator printStream) {  return new Work2(printStream);  }  public One workOne(int[] a, int[] b) {  return new One(a, b);  }  public Two workTwo(String text) {  return new Two(text);  }  public class One {  private final int[] a;  private final int[] b;  public One(int[] a, int[] b) {  this.a = a;  this.b = b;  }  public void execute() {  printIterator.print("Begin: \n" +  "a = " + Arrays.toString(a) + '\n' +  "b = " + Arrays.toString(b) + '\n');  UtilArray.barrierElement(a, 4);  UtilArray.insertionSort(b);  printIterator.print("Sort array: \n" +  "b = " + Arrays.toString(b) + '\n');  final int findItem = Arrays.binarySearch(b, 4);  final Serializable textFindItem = findItem < 0 ? "undenfided" : findItem;  printIterator.print("First: " + UtilArray.barrierElement(a, 4) + '\n');  printIterator.print("Second: " + textFindItem + '\n');  printIterator.print("Unique: " + Arrays.toString(unionListUnique(a, b)) + '\n');  }  private int[] unionListUnique(int[] a, int[] c) {  int[] arr = unionList(a,c);  return Arrays.stream(arr).distinct().toArray();  }  private int[] unionList(int[] a, int[] b) {  int[]c = new int[a.length+b.length];  int count = 0;  for(int i = 0; i<a.length; i++) {  c[i] = a[i];  count++;  }  for(int j = 0;j<b.length;j++) {  c[count++] = b[j];  }  return c;  }  }  public class Two {  private String word;  public Two(String word) {  this.word = word;  }  public void execute() {  char[] mas = word.toCharArray();  boolean isSorted = false;  char buf;  while(!isSorted) {  isSorted = true;  for (int i = 0; i < word.length()-1; i++) {  if(mas[i] > mas[i+1]){  isSorted = false;  buf = mas[i];  mas[i] = mas[i+1];  mas[i+1] = buf;  }  }  }  printIterator.print(Arrays.toString(mas));  }  }  }  class Randomize {  private static final Random rand = new Random();  public static int[] getRandom(int size, int origin, int bound) {  return rand.ints(size, origin, bound).toArray();  }  }  public class Main {  public static void main(String[] args) {  System.out.println("1 Work");  Work2 work2 = Work2.printAction(System.out::print);  Work2.One one = work2.workOne(Randomize.getRandom(5, 1, 5),  Randomize.getRandom(5, 1, 5));  one.execute();  System.out.println("\n2 Work");  Work2.Two two = work2.workTwo("REST14342");  two.execute();  } } |

Скриншоты выполнения:

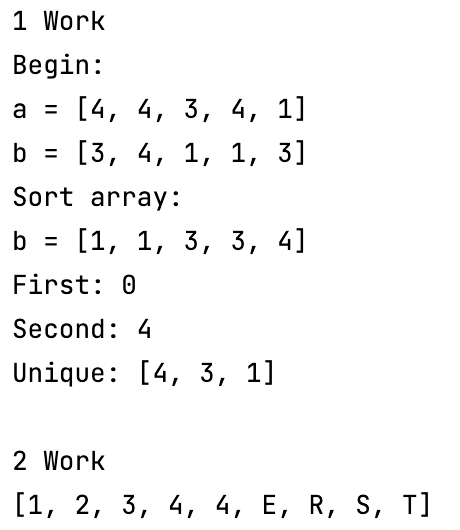


Рисунок 2 – Результат выполнения

Вывод:

Я продолжил знакомство с языком программирования Java. Создал программу, в которой присутствуют: последовательный перебор с использованием барьерного элемента, бинарный поиск, сортировка методом вставки, сортировка методом выбора наименьшего элемента, сортировка методом пузырька, слияние массивов. Тем самым выполнил 2 лабораторную работу.