МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

Технологія розроблення програмного забезпечення 1.

Професійна практика програмної інженерії 1

Лабораторна робота № 2

«Робота з масивами і рядками»

Виконали:

студенти групи КП-61

Чумак Надія,

Свинарчук Максим, Пойда Андріан

Київ 2018

**Завдання 1.**

Вивести внизу прізвище розробника, дату і час отримання завдання, а також дату і час здачі завдання. Для отримання останньої дати і часу використати клас Date. Додати коментарі до програм у вигляді / \*\* коментар \* /, витягнути цю документацію в HTML- файл і переглянути отриману сторінку Web - оглядачем.

Модифікувати програму вирішення завдання, здійснюючи введення рядків як аргументів командного рядка.

Ввести n рядків з консолі. Упорядкувати, а потім вивести рядки в порядку зростання їх довжин, а також значення їх довжин.

|  |
| --- |
| StringContainer.java |
| package com.gmail.kp61.dataStructures; import java.util.Arrays; import java.util.Comparator;  public class StringContainer {  private String[] strings;   public StringContainer(String[] strings) {  this.strings = strings;  }   /\*\*  \* Sorting array in order of increasing strings length  \*/   public void sort() {  Arrays.sort(strings, Comparator.comparingInt(String::length));  }   /\*\*  \* @return the result of calling toString for a non-null object  \*/  @Override  public String toString() {  StringBuilder result = new StringBuilder();  for (int i = 0; i < strings.length; i++) {  result.append(String.format("String[%d]: %s, Length: %d\n", i, strings[i], strings[i].length()));  }  return result.toString();  }   /\*\*  \* @param obj an object to be compared with current object for equality  \* @return true if the objects are equal to each other and false otherwise  \*/  @Override  public boolean equals(Object obj) {  if (this == obj) return true;  if (obj == null || getClass() != obj.getClass()) return false;  StringContainer that = (StringContainer) obj;  return Arrays.equals(strings, that.strings);  }   /\*\*  \* @return the hash code of a non-null object and 0 for a null argument.  \*/  @Override  public int hashCode() {  return Arrays.hashCode(strings);  }} |
| Developer.java |
| package com.gmail.kp61.staff;  import java.util.Date; import java.util.Objects;  public class Developer {  private String fullName;  private String currentTask;   public Developer(String fullName) {  this.fullName = fullName;  this.currentTask = null;  }   public String getFullName() {  return fullName;  }   public String getCurrentTask() {  return currentTask;  }   /\*\*  \* Method to get task  \*  \* @param task assigned task for developer  \* @return date of received task  \*/  public String assignTask(String task) {  currentTask = task;  Date date = new Date(System.currentTimeMillis() - 100000);  return date.toString();  }   /\*\*  \* Method to finish task  \*  \* @return date of finished task or  \* message that the developer does not have the current task  \*/  public String doneTask() {  if (currentTask == null)  return "Developer does not have the current task!";   currentTask = null;   Date date = new Date();  return date.toString();  }    /\*\*  \* @return the result of calling toString for a non-null object  \* and "null" for a null argument  \*/  @Override  public String toString() {  return "Developer{" +  "fullName='" + fullName + '\'' +  ", currentTask='" + currentTask + '\'' +  '}';  }   /\*\*  \* @param obj an object to be compared with current object for equality  \* @return true if the objects are equal to each other and false otherwise  \*/  @Override  public boolean equals(Object obj) {  if (this == obj) return true;  if (obj == null || getClass() != obj.getClass()) return false;  Developer developer = (Developer) obj;  return Objects.equals(fullName, developer.fullName) &&  Objects.equals(currentTask, developer.currentTask);  }   /\*\*  \* @return the hash code of a non-null object and 0 for a null argument.  \*/  @Override  public int hashCode() {  return Objects.hash(fullName, currentTask);  } } |

|  |
| --- |
| SortingStrings.java |
| package com.gmail.kp61.main;  import com.gmail.kp61.dataStructures.StringContainer; import com.gmail.kp61.staff.Developer;  public class SortingStrings {  public static void main(String[] args) {  Developer developer = new Developer("Nadia Chumak");  String dateOfAssignedTask = developer.assignTask(  "Sort and output strings in order of increasing their length.");   StringContainer strCont = new StringContainer(args);  System.out.print("Input strings before sorting:\n" + strCont);  strCont.sort();  System.out.print("Strings in order of increasing their length:\n" + strCont);   System.out.println("Developer: " + developer.getFullName());  System.out.println("Task: " + developer.getCurrentTask());  System.out.println("Date of receipt of the assignment:" +  dateOfAssignedTask);  System.out.println("End date of the assignment:" +  developer.doneTask());  } } |

**Завдання 2.**

В одновимірному масиві, що складається з n цілих елементів, обчислити:

1. Добуток елементів масиву з парними номерами;

2. Суму елементів масиву, розташованих між першим і останнім нульовими елементами.

Перетворити масив таким чином, щоб спочатку розташовувалися всі додатні елементи, а потім - всі від’ємні (елементи, рівні 0, вважати додатними).

|  |
| --- |
| SmartArray.java |
| package com.gmail.kp61.dataStructures;  import java.util.Arrays;  public class SmartArray {  private int[] data;   public SmartArray(int[] arr) {  if (arr == null)  throw new IllegalArgumentException("Input array can not be NULL!");  data = arr;  }   public int multiplOfElmtsWithEvenIndices() throws Exception {  if (data.length == 0)  throw new Exception("Array can not be empty!");   int multiple = 1;  for (int i = 0; i < data.length; i += 2) {  multiple \*= data[i];  }  return multiple;  }   public int sumBtwFirstAndLastZero() throws Exception {  if (data.length == 0)  throw new Exception("Array can not be empty!");   int sum = 0;  int firstIndx = 0;  int lastIndx = data.length - 1;   while (data[firstIndx] != 0 && firstIndx < data.length - 1)  firstIndx++;  while (data[lastIndx] != 0 && lastIndx > 0)  lastIndx--;   for (int i = firstIndx; i < lastIndx; i++)  sum += data[i];   return sum;  }   public int[] sortPositiveAndNegativeElmts() {  int[] resultArr = new int[data.length];  int counter = 0;  for (int value : data)  if (value >= 0)  resultArr[counter++] = value;  for (int value : data)  if (value < 0)  resultArr[counter++] = value;  return resultArr;  }   @Override  public String toString() {  return Arrays.*toString*(data);  }   @Override  public boolean equals(Object obj) {  if (this == obj)  return true;  if (obj == null || getClass() != obj.getClass())  return false;  SmartArray arrForComp = (SmartArray) obj;  return Arrays.*equals*(data, arrForComp.data);  }   @Override  public int hashCode() {  return Arrays.*hashCode*(data);  } } |

|  |
| --- |
| Main.java |
| package com.gmail.kp61.main;  import com.gmail.kp61.dataStructures.SmartArray;  import java.util.Arrays;  public class Main {  public static void main(String[] args) {  try {  SmartArray smArr = new SmartArray(new int[]{1, 0, -3, 0, -5, -6, 2, 0});  System.out.println("Product of array elements with even numbers: " +  smArr.multiplOfElmtsWithEvenIndices());  System.out.println("The sum of array elements located between the first " +  "and last zero elements: " + smArr.sumBtwFirstAndLastZero());  System.out.println("Sorted array (all positive elements " +  "are first located, then all negative ones): " +  Arrays.toString(smArr.sortPositiveAndNegativeElmts()));  } catch (Exception e) {  System.out.println(e.toString());  }   } } |