+Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |
| --- |
| **Отчет** |
| по лабораторной работе №3 по дисциплине «Объектно-ориентированное программирование»  «Работа с потоками данных и обработка  исключений» |
| наименование темы  Вариант №9 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | АСУб-19-1 |  |  |  | К.У. Ботурхонов |
|  |  | шифр |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Т.В. Маланова |
|  |  |  |  | подпись |  | И.О. Фамилия |
| Работа защищена с оценкой | | | |  | | |

Иркутск 2020 г.

**Содержание**

[1 Постановка задачи 3](#_Toc53658100)

[2 Проектирование классов 4](#_Toc53658101)

[3 Описание методов классов и спецификация локальных переменных 5](#_Toc53658102)

[4 Таблица тестов 6](#_Toc53658103)

[5 Результаты тестирования 7](#_Toc53658104)

[6 Исходный код 9](#_Toc53658105)

[Список использованных источников 14](#_Toc53658106)

1 Постановка задачи

a. Сгенерируйте числовой ряд длиной не более 50, представляющий

собой числа Фибоначчи (каждый последующий элемент представляют суммы двух предыдущих, первые два элемента равны 1). Выведите на печать первые 10, отмечая четные числа каким-нибудь

символом.

b. Дан массив чисел, знак операции и операнд. Выведите на печать

массив, полученный в результате арифметического действия, совершенного над элементами массива.

c. Дана строка содержащая ряд чисел в двоичном виде, необходимо

проверить упорядочены ли они.

Провести объектную декомпозицию задачи из индивидуального задания и разработать класс, содержащий соответствующие поля для хранения необходимых данных и методы, обеспечивающие достаточную для решения задачи функциональность класса. Разработать класс, реализующий текстовое меню, позволяющее осуществлять выбор вариантов выполнения индивидуального задания. Создание объекта класса, решающего задание, и вызов его методов должны осуществляться из класса реализующего меню. При выполнении задания предусмотреть методы для организации ввода и выводы данных как на консоль, так и в текстовые файлы. Работа с файлами может быть организована методами класса реализующего пользовательское меню, либо в рамках специально разработанного класса. Для всех проверяемых исключений должны быть реализованы обработчики, в которых выводится сообщение о возникновении исключительной ситуации и предпринимаются меры для продолжения работы программы в штатном режиме.

2 Проектирование классов

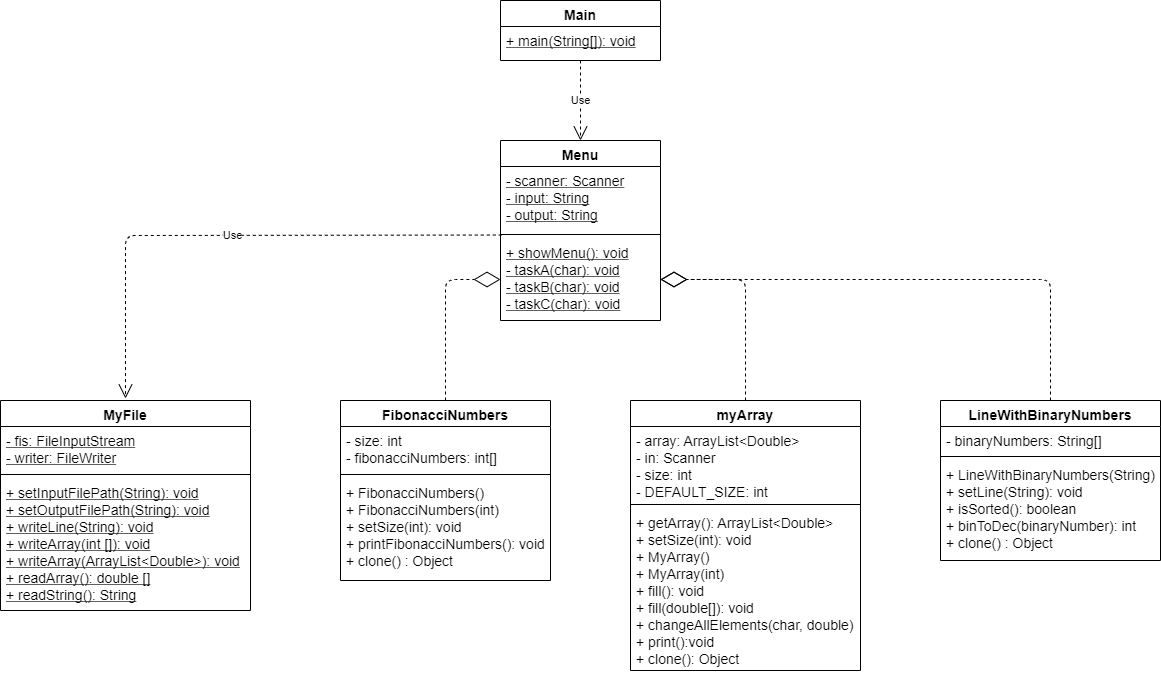
****

Рисунок 1 - Структура классов

3 Описание методов классов и спецификация локальных переменных

Таблица 1 – Описание методов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Назначение | Модификатор доступа | Аргументы | Возвращаемый тип |
| **Класс Menu** | | | | |
| showMenu | пользовательская меню | public static | - | void |
| taskA | Выполнение задания A | public static | char inputOption | void |
| taskB | Выполнение задания B | public static | char inputOption | void |
| taskC | Выполнение задания C | public static | char inputOption | void |
| **Класс MyFile** | | | | |
| setInputFilePath | Установка пути файла для чтения | public static | String path | void |
| setOutputFilePath | Установка пути файла для записи | public static | String path | void |
| writeLine | Запись строки в файл | public static | String line | void |
| writeArray | Запись массива в файл | public static | int [] array, ArrayList <Double> arrayList | void |
| readArray | Чтение массива из файла | public static | - | double [] |
| readString | Чтение строки из файла | public static | - | String |

4 Таблица тестов

Таблица 3 - Тесты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Задание | Вход | Выход | Примечания |
| 1 | A | - | 1 1 2% 3 5 8% 13 21 34% 55 | Первые 10 чисел Фибоначчи |
| 2 | B | taskB.bin  ch = ‘+’  operand = 5 | taskB.out | Элементы массива увеличены на 5 |
| 3 | C | “10 100 11 1000” | “Ряд не упорядочен” | Второе число больше |
| 4 | B | {88 46 9 11 5 10}  ch = ‘/’  operand = 2 | 44.0 23.0 4.5 5.5 2.5 5.0 | Элементы массива разделены на 2 |
| 5 | C | “1 10 11 100 101 110 111” | “Ряд упорядочен” | - |
| 6 | B | {4 9 6 0}  ch = ‘\*’  operand = 3 | 12.0 27.0 18.0 0.0 | Элементы массива умножены на 3 |
| 7 | C | “10 100 1000 1001” | “Ряд упорядочен” |  |

5 Результаты тестирования

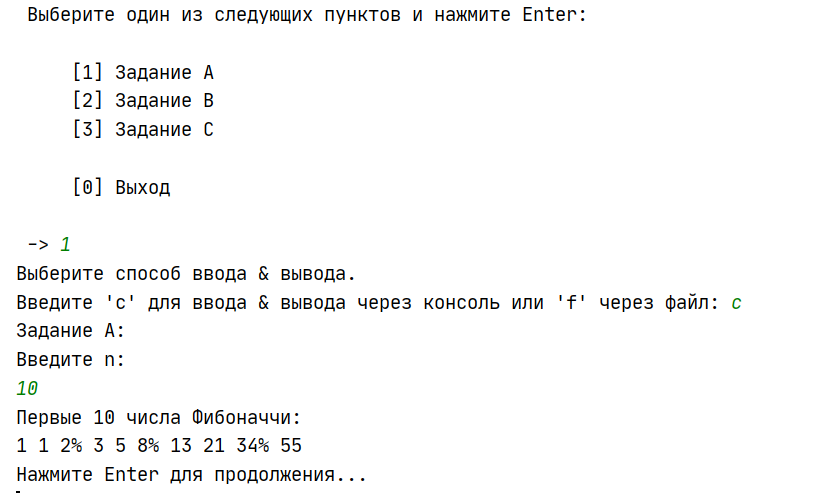


Рисунок 2 – Результат тестов 1

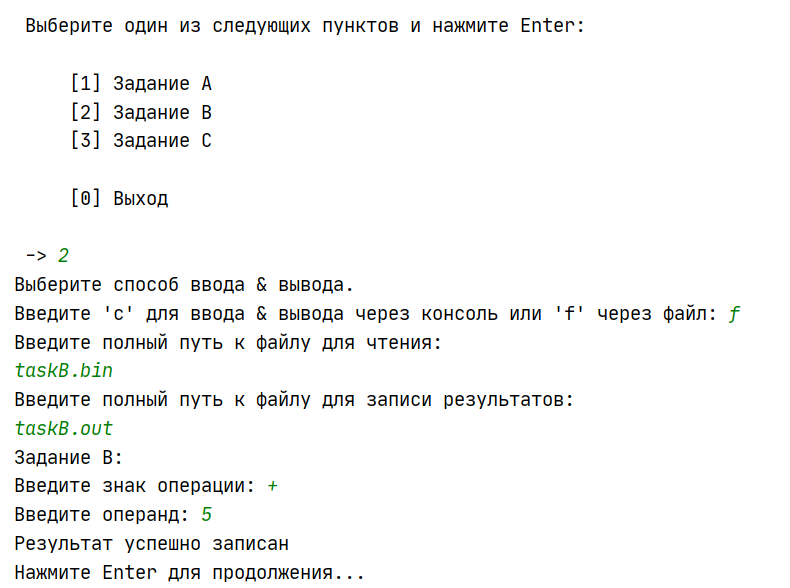


Рисунок 3 – Результат тестов 2

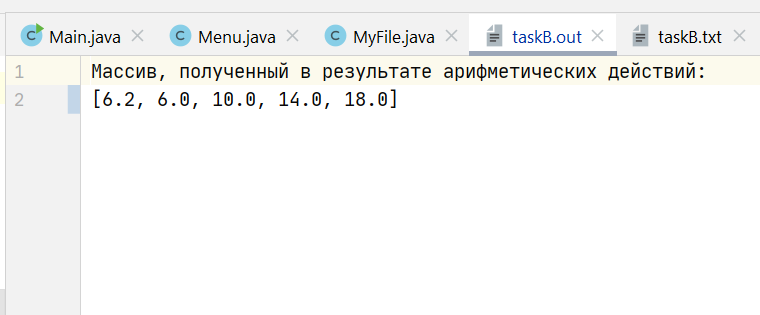


Рисунок 4 – Результат записанный в taskB.out

6 Исходный код

Исходный код класса Menu

import java.util.Scanner;  
  
public class Menu {  
 private static final Scanner *scanner* = new Scanner(System.*in*);  
 private static String *input*, *output*;  
  
 public static void showMenu() {  
 int option = 1;  
 while (option != 0) {  
 System.*out*.println(" Выберите один из следующих пунктов и нажмите Enter: \n");  
 System.*out*.println("\t [1] Задание A");  
 System.*out*.println("\t [2] Задание B");  
 System.*out*.println("\t [3] Задание C");  
 System.*out*.println();  
 System.*out*.println("\t [0] Выход\n");  
 System.*out*.print(" -> ");  
 option = *scanner*.nextInt();  
 if (option == 0) continue;  
 System.*out*.println("Выберите способ ввода & вывода.");  
 System.*out*.print("Введите 'c' для ввода & вывода через консоль или 'f' через файл: ");  
 *scanner*.nextLine();  
 char inputOption = *scanner*.nextLine().charAt(0);  
 if (inputOption == 'f') {  
 if (option != 1) {  
 System.*out*.println("Введите полный путь к файлу для чтения: ");  
 *input* = *scanner*.nextLine();  
 MyFile.*setInputFilePath*(*input*);  
 }  
 if (option != 3) {  
 System.*out*.println("Введите полный путь к файлу для записи результатов: ");  
 *output* = *scanner*.nextLine();  
 MyFile.*setOutputFilePath*(*output*);  
 }  
 }  
 switch (option) {  
 case 1:  
 System.*out*.println("Задание A: ");  
 *taskA*(inputOption);  
 break;  
 case 2:  
 System.*out*.println("Задание B: ");  
 *taskB*(inputOption);  
 break;  
 case 3:  
 System.*out*.println("Задание C: ");  
 *taskC*(inputOption);  
 break;  
*// case 0:  
// continue;* default:  
 System.*out*.println("Пункт №" + option + " не найден");  
 }  
 System.*out*.println("Нажмите Enter для продолжения...");  
 *scanner*.nextLine();  
 *scanner*.nextLine();  
 }  
  
 }  
 private static void taskA(char inputOption) {  
 System.*out*.println("Введите n: ");  
 int n = *scanner*.nextInt();  
 FibonacciNumbers fibonacciNumbers = new FibonacciNumbers();  
 fibonacciNumbers.setSize(n);  
  
 if (inputOption == 'f') {  
 MyFile.*writeLine*("Первые " + n + " числа Фибоначчи: ");  
 MyFile.*writeArray*(fibonacciNumbers.getFibonacciNumbers());  
 } else {  
 System.*out*.println("Первые " + n + " числа Фибоначчи: ");  
 fibonacciNumbers.printFibonacciNumbers();  
 }  
 }  
 private static void taskB(char inputOption) {  
 *//-----------------------------Ввод массива-------------------------------------* int size;  
 MyArray myArray = new MyArray();  
  
 if (inputOption == 'f') {  
 myArray.fill(MyFile.*readArray*());  
 } else {  
 System.*out*.print("Введите размер массива: ");  
 size = *scanner*.nextInt();  
 myArray.setSize(size);  
 System.*out*.println("Введите массив: ");  
 myArray.fill();  
 }  
  
 *//--------------------------Операция с массивом---------------------------------* System.*out*.print("Введите знак операции: ");  
 String operationSign = *scanner*.next();  
 System.*out*.print("Введите операнд: ");  
 double operand = *scanner*.nextInt();  
 myArray.changeAllElements(operationSign.charAt(0), operand);  
 *//----------------------------Вывод массива-------------------------------------* if (inputOption == 'f') {  
 MyFile.*writeLine*("Массив, полученный в результате арифметических действий:");  
 MyFile.*writeArray*(myArray.getArray());  
 } else {  
 System.*out*.println("Массив, полученный в результате арифметических действий:");  
 myArray.print();  
 }  
 }  
  
 private static void taskC(char inputOption) {  
  
 String line = "";  
  
 if (inputOption == 'f') {  
 line = MyFile.*readString*();  
 System.*out*.println("Считанная сторока из файла: ");  
 System.*out*.println(line);  
 } else {  
 System.*out*.println("Введите строку, содержащую ряд чисел в двоичном виде: ");  
 *scanner*.nextLine();  
 line = *scanner*.nextLine();  
 }  
  
 LineWithBinaryNumbers lineWithBinaryNumbers = new LineWithBinaryNumbers(line);  
  
 if (lineWithBinaryNumbers.isSorted()) {  
 System.*out*.println("Ряд упорядочен");  
 } else {  
 System.*out*.println("Ряд не упорядочен");  
 }  
 }  
}

Исходный код класса MyFile

import java.io.\*;  
import java.util.ArrayList;  
import java.util.Arrays;  
  
public class MyFile {  
  
 private static FileInputStream *fis* = null;  
 private static FileWriter *writer* = null;  
  
 public static void setInputFilePath(String path) {  
 try {  
 *fis* = new FileInputStream(path);  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void setOutputFilePath(String path) {  
 try {  
 *writer* = new FileWriter(path);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void writeLine (String line) {  
 try {  
 *writer*.write(line+"\n");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void writeArray(int [] array) {  
 try {  
 *writer*.write(Arrays.*toString*(array));  
 System.*out*.println("Результат успешно записан");  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 try {  
 *writer*.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 public static void writeArray(ArrayList<Double> arrayList) {  
 try {  
 *writer*.write(Arrays.*toString*(arrayList.toArray()));  
 System.*out*.println("Результат успешно записан");  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 try {  
 *writer*.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 public static double [] readArray () {  
 double [] array = null;  
  
 try (DataInputStream dis = new DataInputStream(*fis*)) {  
 int size = (int) dis.readDouble();  
 array = new double[size];  
 for (int i = 0; i < size; i++) {  
 array[i] = dis.readDouble();  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 return array;  
 }  
  
 public static String readString () {  
 String line = "";  
 try {  
 *// читаем посимвольно* int c;  
 while((c = *fis*.read())!=-1){  
 line += String.*valueOf*((char)c);  
 }  
 }  
 catch(IOException ex){  
  
 System.*out*.println(ex.getMessage());  
 }  
 return line;  
 }  
  
}

Список использованных источников

1. API Specification for the Java 7 SE. [официальный сайт]

URL: <http://docs.oracle.com/javase/7/docs/api/> (дата обращения:05.09.2020)

1. The Java Tutorials. SE [электронный ресурс],

URL:<http://docs.oracle.com/javase/tutorial/index.html>

(дата обращения:05.09.2020)