### 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

Центр программной инженерии

ОТЧЕТ

к лабораторной работе №3 по дисциплине:

|  |
| --- |
| «Работа с потоками данных и обработка исключений» |

наименование темы

Вариант №9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент | ИСТб 19-2 |  |  |  | Козлов М.В. |
|  | шифр группы |  | подпись |  | Фамилия ИО |
| Проверил | | |  |  | Маланова Т.В |
|  | | | подпись |  | Фамилия ИО |
| Работа защищена с оценкой | | |  | | |

Иркутск 2020 г

**Содержание**

[1 Постановка задачи 3](#_Toc57246105)

[2 Описание структуры пользовательского меню 4](#_Toc57246106)

[3 Описание структуры классов 5](#_Toc57246107)

[4 Описание методов классов и спецификации локальных переменных 11](#_Toc57246108)

[5 Таблица тестов 16](#_Toc57246109)

[6. Результаты тестирования 19](#_Toc57246110)

[7. Исходный код 33](#_Toc57246111)

[Список использованных источников 46](#_Toc57246112)

1 Постановка задачи

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.

Вариант №9

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.

2 Описание структуры пользовательского меню

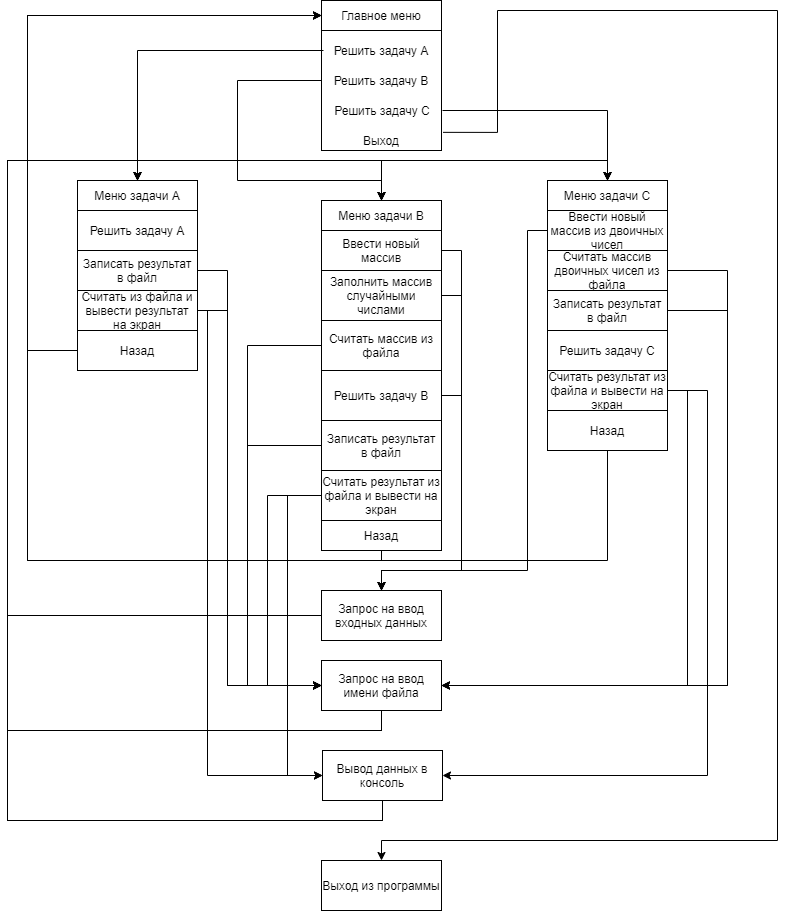


Рисунок 2.1 – Описание структуры пользовательского меню

3 Описание структуры классов

**3.1** **Класс main**

Класс предназначен для вызова меню.

Таблица 3.1 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| main | void | Да | public | String[] args | Точка входа в программу |

**3.2** **Класс Menu**

Класс предназначен для представления меню

Таблица 3.2 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| Start | void | Да | public | - | Вызов главного меню |
| mainMenu | void | Да | private | - | Меню выбора задачи для выполнения |
| menuTaskA | void | Да | private | - | Меню для задачи А |
| menuTaskB | void | Да | private | - | Меню для задачи B |
| menuTaskC | void | Да | private | - | Меню для задачи C |
| inputFileName | String | Да | public | - | Метод ввода имени файла |

**3.3** **Класс InputDataClass**

Класс предназначен для ввода или генерации необходимых данных для заданий.

Таблица 3.3 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| userInputBinaryNum | String | Да | public | - | Метод ввода двоичных чисел |
| userInput | int | Да | public | String message | Метод ввода целых числовых значений |
| completeTaskB | int[] | Да | public | int[] Arr | Метод, предназначенный для ввода необходимых данных для решения задачи В |
| console | int[] | Да | public | - | Метод, реализующий ввод значений с консоли |
| generateRandomArr | int[] | Да | public | - | Метод, предназначенный для генерации случайных чисел |

**3.4** **Класс InputDataClass**

Класс предназначен для ввода или генерации необходимых данных для заданий.

Таблица 3.4 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| outputByte | int[] | Да | public | String filename | Метод для считывания данных из файла, исполь-зующий байтовый поток |
| FromTxtFile | String | Да | public | String filename | Метод для считывания данных из файла, исполь-зующий символьный поток |
| ToTxtFile | void | Да | public | String task,  String filename | Метод для записи данных в файл, использующий символьный поток. |
| InputByte | void | Да | public | int[] arrInt  String filename | Метод для записи данных в файл, использующий байтовый поток. |

**3.5** **Класс TaskA**

Этот класс реализует решение задания А.

Таблица 3.5.1 – Таблица полей

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Уровень доступа | Назначение |
| taskA | String | private | Строка – решение задачи А. |

Таблица 3.5.2 – Таблица конструкторов

|  |  |  |
| --- | --- | --- |
| Уровень доступа | Аргументы | Описание |
| public | - | Конструктор, реализующий решение задачи А, через метод completeTaskA. |

Таблица 3.5.3 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| completeTaskA | String | Нет | private | - | Метод, решающий задачу A |
| getTaskA | String | Нет | public | - | Метод, возвращающий решение задачи А |

**3.6** **Класс TaskB**

Этот класс реализует решение задания B.

Таблица 3.6.1 – Таблица полей

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Уровень доступа | Назначение |
| taskB | int[] | private | Массив – решение задачи B. |

Таблица 3.6.2 – Таблица конструкторов

|  |  |  |
| --- | --- | --- |
| Уровень доступа | Аргументы | Описание |
| public | int[] Arr, int operand, String sign | Конструктор, реализующий решение задачи B,через вызов метода completeTaskB. |

Таблица 3.6.3 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| completeTaskB | int[] | Нет | private | int[] Arr,  int operand, String sign | Метод, решающий задачу B |
| getTaskB | int[] | Нет | public | - | Метод, возвращающий решение задачи B |

**3.7** **Класс TaskC**

Этот класс реализует решение задания C.

Таблица 3.7.1 – Таблица полей

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Уровень доступа | Назначение |
| taskC | String | private | Строка – решение задачи С. |

Таблица 3.7.2 – Таблица конструкторов

|  |  |  |
| --- | --- | --- |
| Уровень доступа | Аргументы | Описание |
| public | String binNumStr | Конструктор, реализующий решение задачи С, через вызов метода completeTaskB. |

Таблица 3.7.3 – Таблица методов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Тип | Статический | Уровень доступа | Аргументы | Назначение |
| completeTaskC | String | Нет | private | String binNumStr | Метод, решающий задачу B |
| binaryCodeToDecimal | int | Да | private | String str | Метод, переводящий двоичные числа в десятичные |
| getTaskC | String | Нет | public | - | Метод, возвращающий решение задачи B |

4 Описание методов классов и спецификации локальных переменных

**4.1** **Класс TaskC**

Этот класс реализует решение задания C.

Таблица 4.1 – Таблица описания методов и локальных переменных

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя | Назначение | Тип |
| **Class Main** | | | |
| **Статический публичный метод main (String[] args) – Точка входа в программу.** | | | |
| args | String[] | Аргумент. Массив аргументов коммандной строки | - |
| **Class Menu** | | | |
| **public static void Start() – метод для вызова главного меню** | | | |
| **public static String inputFileName()** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | filename | Имя файла | String |
| **private static void mainMenu() – главное меню, меню выбора задания** | | | |
| 1 | choice | Число, обозначающее выбор пользователя | int |
| **private static void menuTaskB() – меню задачи В** | | | |
| 1 | choice | Число, обозначающее выбор пользователя | int |
| 2 | resultTaskB | Результат решения задачи В | int[] |
| 3 | Arr | Входной массив | int[] |
| **private static void menuTaskA() – меню задачи А** | | | |
| 1 | choice | Число, обозначающее выбор пользователя | int |
| 2 | taskA | Решение задачи А | String |
| 3 | flag | Флаг, который предназначен для проверки, решена задача или нет. | boolean |
| 4 | a | Объект класса Task A,явлющийся решением задачи | TaskA |
| **private static void menuTaskC()** | | | |
| 1 | choice | Число, обозначающее выбор пользователя | int |
| 2 | flag | Флаг, который предназначен для проверки, решена задача или нет. | boolean |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 4.1 | | | |
| 3 | resultTaskC | Решение задачи С | String |
| 4 | cls | Объект класса Task С, явлющийся решением задачи | TaskC |
| **Class InputDataClass** | | | |
| **public static String userInputBinaryNum() – метод для ввода двоичных чисел** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | size | Число, предназначенное для обозначения количества введённых двоичных чисел. | int |
| 3 | bin | Строка двоичных чисел | String |
| 4 | binaryNumber | Строка, которая проходит проверку, является ли она двоичным числом или нет. | String |
| **userInput() - метод для ввода целых чисел** | | | |
| 1 | message | Сообщение о том, что необходимо ввести в данный момент | String |
| 2 | i | Выводимое число | int |
| **public static int[] completeTaskB (int[] Arr) – метод для ввода необходимых данных для задачи В, и вызова метода решения задачи В.** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | Arr | Входной массив чисел для решения задачи | int[] |
| 3 | num | Операнд для задачи В | int |
| 4 | sign | Арифметический знак для задачи В | String |
| 5 | B | Возвращаемое решение задачи В | int[] |
| **public static int[] console () – метод для самостоятельного заполнения массива числами** | | | |
| 1 | size | Размерность массива | int |
| 2 | array |  | int[] |
| **generateRandomArr() – метод для заполнения массива случайными значениями.** | | | |
| 1 | array | Заполненный случайными значениями массив | int[] |
| 2 | size | Размерность массива | int |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 4.1 | | | |
| **Сlass TaskA** | | | |
| 1 | taskA | Поле класса для записи ответа на задачу А. | String |
| **private String completeTaskA() – метод решения задачи А** | | | |
| 1 | array | Массив чисел, который будет заполнен числами Фибоначи | int[] |
| 2 | i | Переменная- счётчик итераций | int |
| 3 | answerA | Результат решения задачи А: первые 10 чисел Фибоначи с отмеченными каким-либо символом чётными числами | String |
| **Class TaskB** | | | |
| 1 | taskB | Поле класса для записи ответа на задачу B. | int[] |
| **private int[] completeTaskB(int[] Arr,int operand, String sign) – метод решения задачи В** | | | |
| 1 | Arr | Массив чисел для арифметического действия | int[] |
| 2 | sign | Арифметический знак для выбора арифметического действия | String |
| 3 | operand | Операнд для арифметического действия | int |
| **Class TaskC** | | | |
| 1 | taskС | Поле класса для записи ответа на задачу С. | String |
| **private static int binaryCodeToDecimal(String str) – метод перевода двоичных чисел в десятичные** | | | |
| 1 | chars | Преобразованное в массив символов двоичное число | char[] |
| 2 | str | Двоичное число | String |
| 3 | result | Выходное значение - десятичное число | int |
| 4 | multiply | Вспомогательная переменная для проведения расчётов | int |
| **private String completeTaskC(String binNumStr) – метод для решения задачи С** | | | |
| 1 | up | Флаг, предназначенная для определения, является ли массив чисел возрастающим или нет | boolean |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 4.1 | | | |
| 2 | down | Флаг, предназначенная для определения, является ли массив чисел убывающим или нет | boolean |
| 3 | Answer | Выходная строка – ответ, которая принимает значение в зависимости от того, является ли строка убывающей, возрастающей или неупорядоченной. | String |
| 4 | binNumStr | Входная строка двоичных чисел | String |
| **Class InputOutputClass** | | | |
| **public static int[] outputByte(String filename) – метод для чтения данных из файла с использованием байтового потока.** | | | |
| 1 | tempArr | Считанные данные из файла | int [] |
| 2 | size | Переменная – количество считанных байт. | int |
| 3 | filename | Имя файла | String |
| 4 | dis | Объект класса DataInputStream, используемый для считывания из файла | DataInputStream |
| **public static String FromTxtFile(String filename) –** **метод для чтения данных из файла с использованием байтового потока.** | | | |
| 1 | s | Считанная строка из файла | String |
| 2 | filename | Имя файла | String |
| 3 | br | Объект класса BufferedReader, используемый для считывания из файла | BufferedReader |
| **public static void ToTxtFile(String task,String filename) – метод для записи данных в файл с использованием символьного потока** | | | |
| 1 | task | Значение, что необходимо записать в файл | String |
| 2 | filename | Имя файла | String |
| 3 | br | Объект класса BufferedWriter, используемый для записи в файл | BufferedWriter |
| **public static void inputByte(int[] arrInt,String filename) – метод для записи данных в файл с использованием байтового потока.** | | | |
| 1 | arrInt | Массив чисел для записи | int[] |
| 2 | filename | Имя файла | String |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 4.1 | | | |
| 3 | dos | Объект класса DataOutput-Stream, используемый для записи в файл | DataOutputStream |

5 Таблица тестов

Таблица 5.1 – Таблица тестов

|  |  |  |
| --- | --- | --- |
| № | Действия | Результат |
| 1 | Выбор несуществующего варианта действия | Вывод сообщения:  "Вы выбрали несуществующий вариант!" |
| 2 | Решить задачу А | Переход к меню задачи А |
| 3 | Выбор несуществующего варианта действия | Вывод сообщения:  "Вы выбрали несуществующий вариант!" |
| 4 | Решить задачу А | Вывод сообщения: “Задача А решена” |
| 5 | Записать результат в файл | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения: "Запись успешно сохранена в файл: “имя файла”» |
| 6 | Вывести результат на экран | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения:  “Файл “имя файла” успешно считан  Числовой ряд Фибоначи: 1 1 \*2 3 5 \*8 13 21 \*34 55” |
| 7 | Попытка записи в файл нерешённой задачи | Вывод сообщения: “Задача не решена!” |
| 8 | Попытка считывания из несуществующего файла | Вывод сообщения: “Файл не найден!” |
| 9 | Считывание пустого файла | Вывод сообщения:”Файл пуст!” |
| 10 | Решить задачу В | Переход к меню задачи В |
| 11 | Ввести новый массив | Переход к вводу размерности и значений элементов массива |
| 12 | Заполнить массив случайными числами | Переход к вводу размерности массива |

|  |  |  |
| --- | --- | --- |
|  | | |
| 13 | Считать массив из файла | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения: "Файл (имя файла) успешно считан" |
| 14 | Решить задачу В | Переход к вводу операнда и знака арифметического действия. |
| 15 | Записать результат в файл | Вывод сообщения: "Запись успешно сохранена в файл: "+ (имя файла) |
| 16 | Считать результат из файла и вывести на экран | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения: “Запись успешно считана из файла: (имя файла)”  Результат |
| 17 | Попытка решения задачи с пустым массивом чисел | Вывод сообщения: "Исходный массив пуст!" |
| 18 | Попытка записи в файл нерешённой задачи | Вывод сообщения: "Задача не решена!" |
| 19 | Выбор несуществующего варианта действия | Вывод сообщения:  "Вы выбрали несуществующий вариант!" |
| 20 | Считывание пустого файла | Вывод сообщения:  "Файл подозрителен..."  «Файл пуст» |
| 21 | Считывание файла некорректного содержания | Вывод сообщения: "Формат файла некорректен" |
| 22 | Решить задачу С | Переход к меню задачи C |
| 23 | Ввести новый массив из двоичных чисел | Вывод сообщения: "Введите размерность массива"  Переход к вводу размерности массива  Вывод сообщения: "Enter a Binary Number: --> "  Переход к вводу двоичных чисел |
| 24 | Попытка ввода некорректного двоичного числа | Вывод сообщения: "Invalid Binary Number Entered! Try Again..." |

|  |  |  |
| --- | --- | --- |
| Продолжение таблицы | | |
| 25 | Попытка ввода некорректной размерности (буквы) | Вывод сообщения: "Вы выбрали несуществующий вариант!" |
| 26 | Попытка ввода некорректной размерности (меньше или равен 1) | Вывод сообщения: "Введите значение размерности массива больше 1 |
| 27 | Считать массив двоичных чисел из файла | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения:  “Файл “имя файла” успешно считан |
| 28 | Решить задачу С | - |
| 29 | Попытка решить задачу С без двоичных чисел | Вывод сообщения: "Нет входных данных!" |
| 30 | Запись решения задачи | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения: “Запись успешно сохранена в файл: (имя файла)” |
| 31 | Попытка записи в файл без решения задачи | Вывод сообщения: "Задача не решена!" |
| 32 | Попытка считывания пустого файла | Вывод сообщения: "Файл пуст!” |
| 33 | Попытка считывания файла с некорректным содержанием | Вывод сообщения:“Файл (имя файла) успешно считан”  "Формат файла некорректен" |
| 34 | Считать результат из файла и вывести на экран | Вывод сообщения: “Введите имя файла”  Переход к вводу имени файла  Вывод сообщения: "Файл (имя файла) успешно считан" |
| 35 | Задача В. Ввод некорректной величины (буква) | Вывод сообщения: "Вы выбрали несуществующий вариант!" |
| 36 | Задача В. Ввод некорректной величины размерности массива (0) | Вывод сообщения: “Размерность массива не может быть меньше или равной 0 |
| 37 | Задача В. Ввод некорректного арифметического знака | Вывод сообщения: “Неверный ввод арифметического знака!” |

6. Результаты тестирования

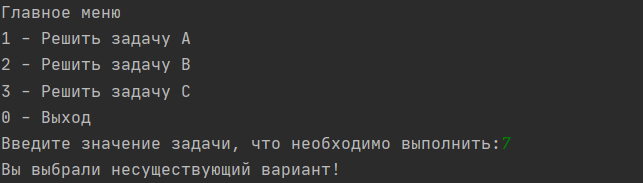


Рисунок 6.10 – Результат теста 1

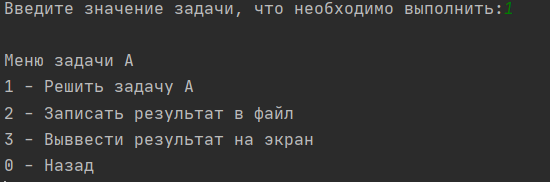


Рисунок 6.11 – Результат теста 2

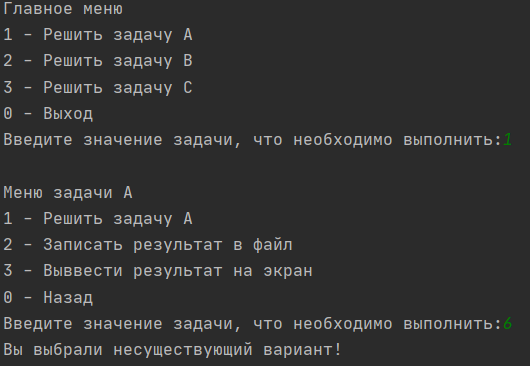


Рисунок 6.12 – Результат теста 3

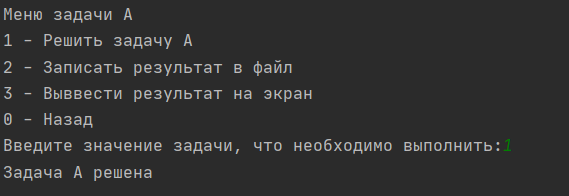


Рисунок 6.13 – Результат теста 4

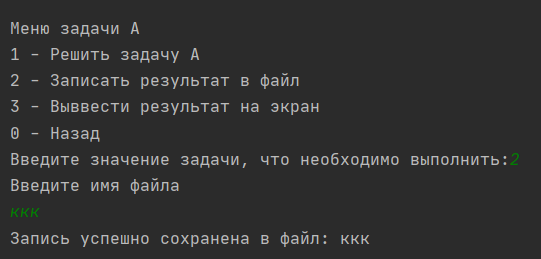


Рисунок 6.14 – Результат теста 5

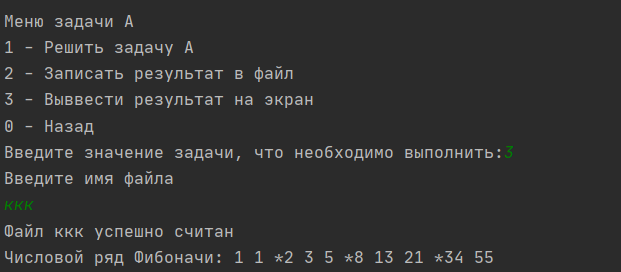


Рисунок 6.15 – Результат теста 6

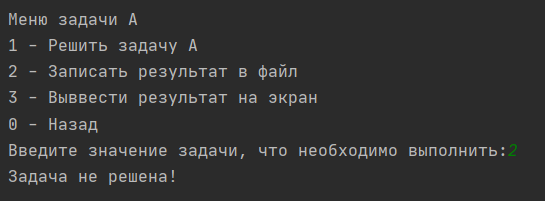


Рисунок 6.16 – Результат теста 7

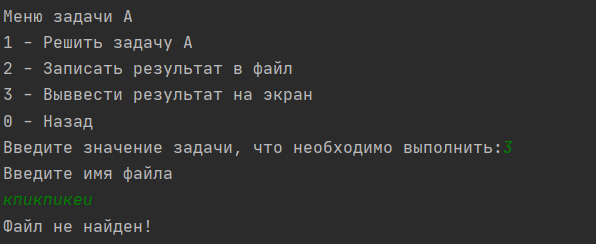


Рисунок 6.17 – Результат теста 8

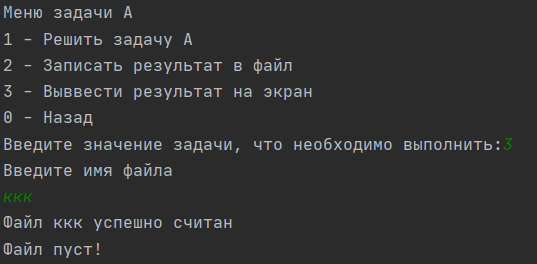


Рисунок 6.18 – Результат теста 9

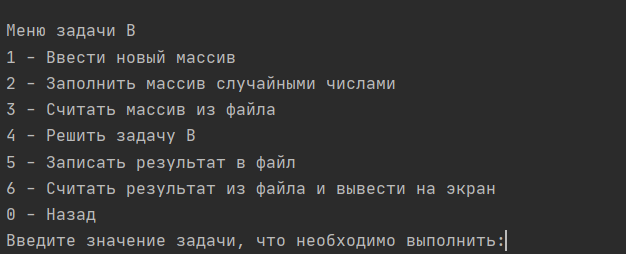


Рисунок 6.19 – Результат теста 10

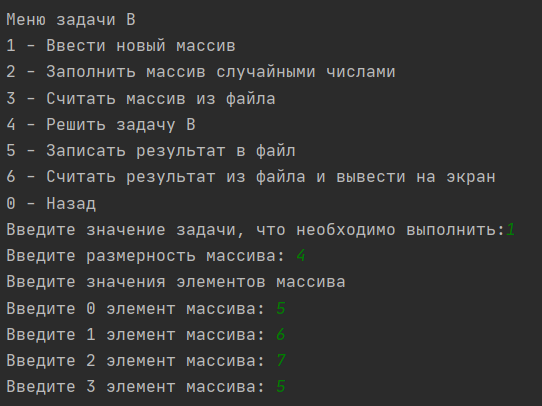


Рисунок 6.20 – Результат теста 11

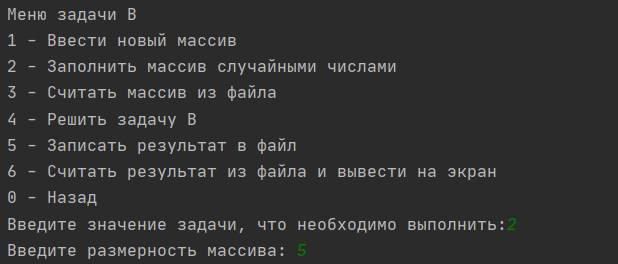


Рисунок 6.21 – Результат теста 12

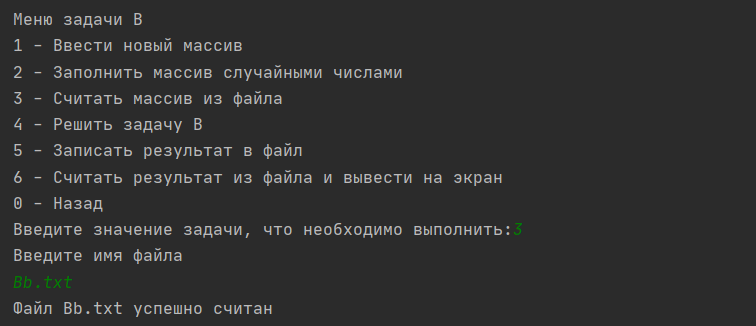


Рисунок 6.22 – Результат теста 13

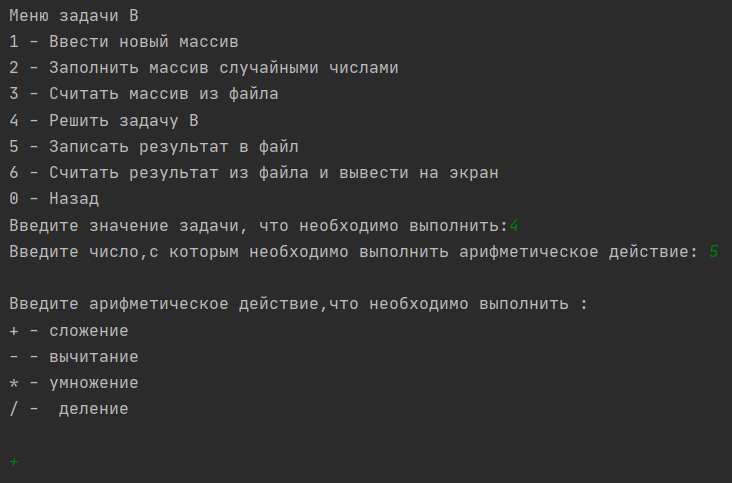


Рисунок 6.23 – Результат теста 14

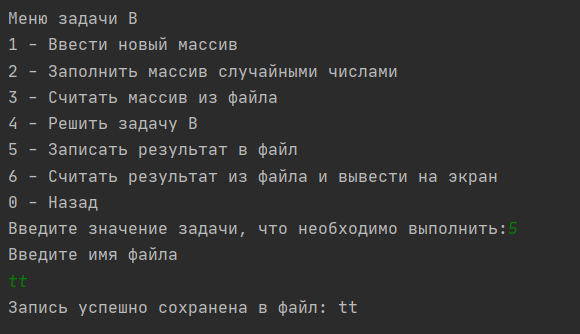


Рисунок 6.24 – Результат теста 15

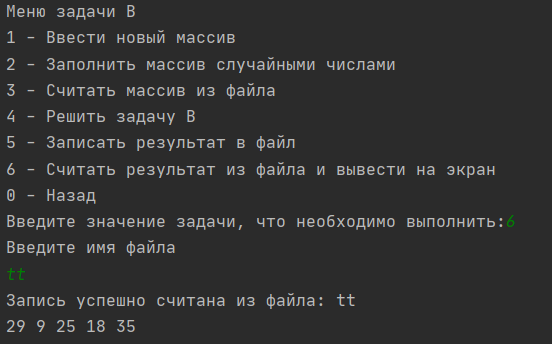


Рисунок 6.25 – Результат теста 16

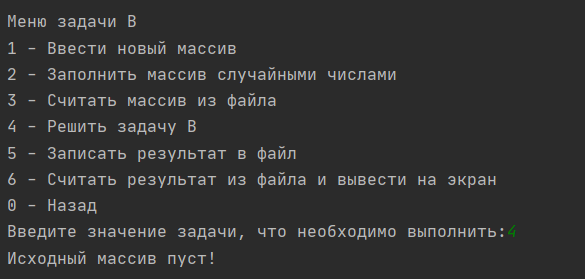


Рисунок 6.26 – Результат теста 17

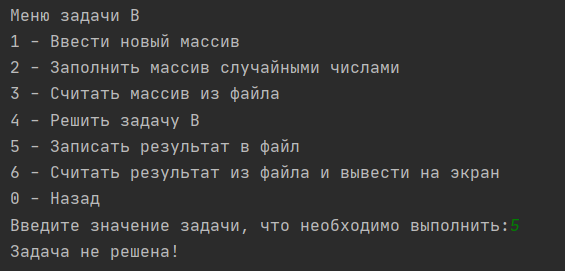


Рисунок 6.27 – Результат теста 18

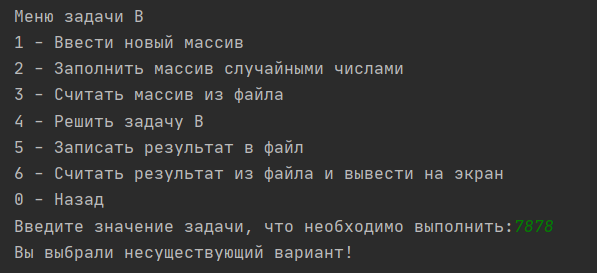


Рисунок 6.28 – Результат теста 19

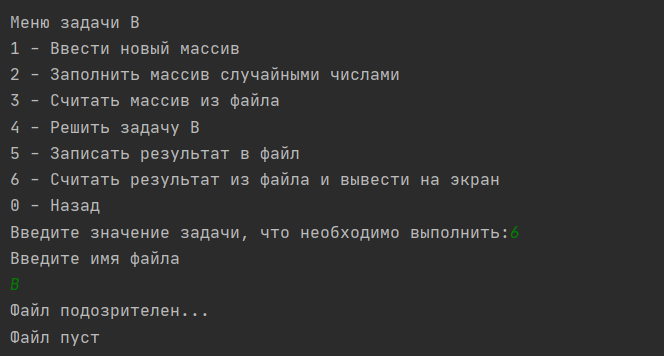


Рисунок 6.29 – Результат теста 20

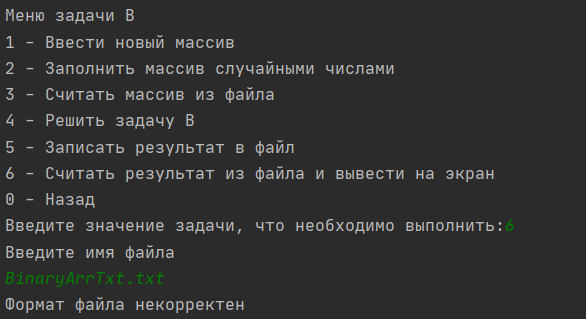


Рисунок 6.30 – Результат теста 21

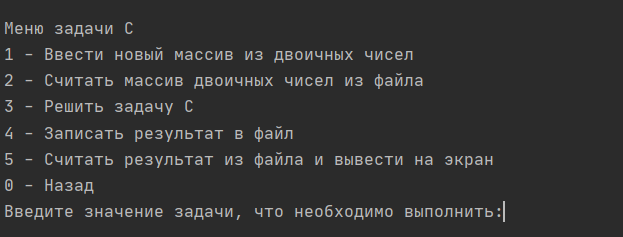


Рисунок 6.31 – Результат теста 22

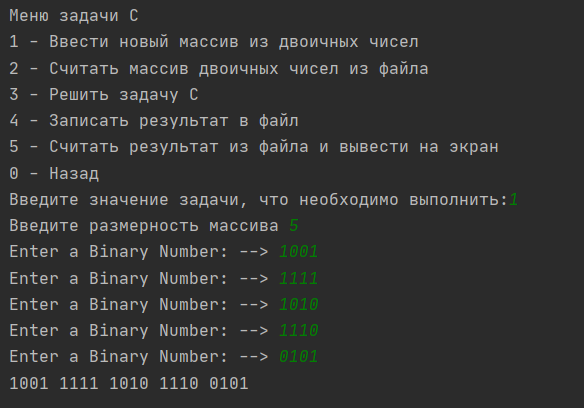


Рисунок 6.32 – Результат теста 23

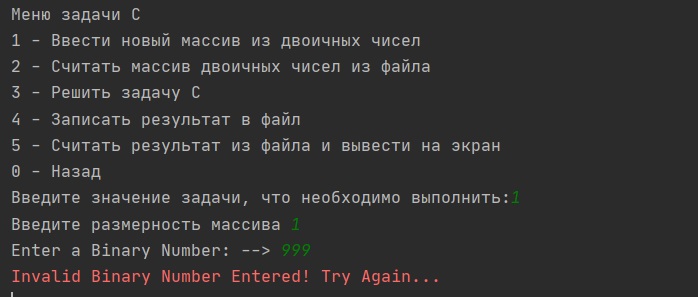


Рисунок 6.33 – Результат теста 24

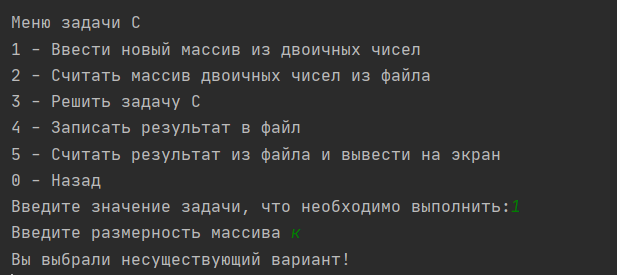


Рисунок 6.34 – Результат теста 25

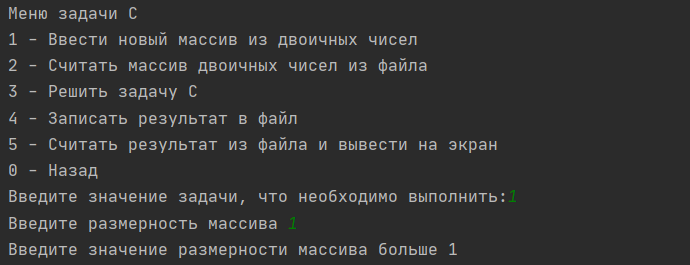


Рисунок 6.35 – Результат теста 26

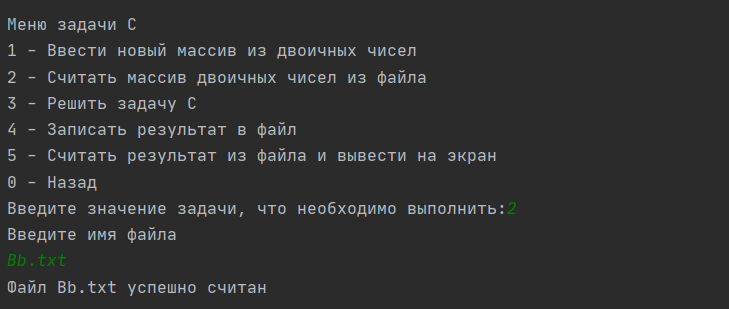


Рисунок 6.36 – Результат теста 27

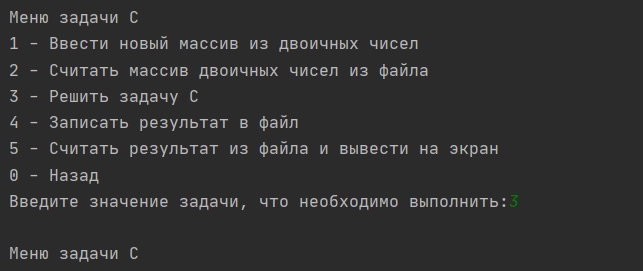


Рисунок 6.37 – Результат теста 28

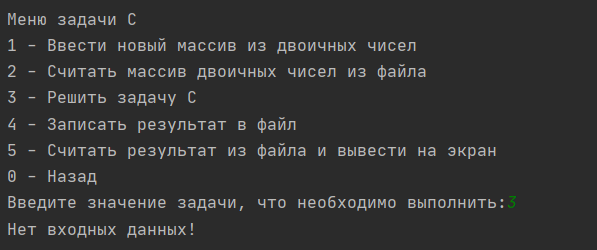


Рисунок 6.38 – Результат теста 29

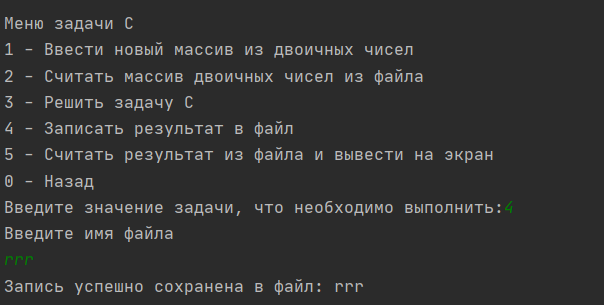


Рисунок 6.39 – Результат теста 30

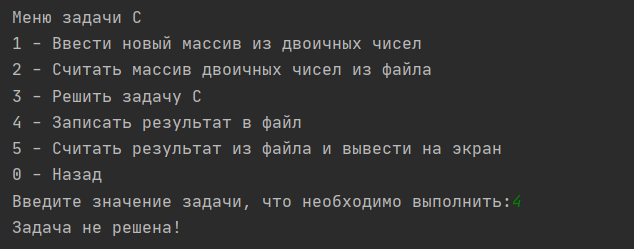


Рисунок 6.40 – Результат теста 31

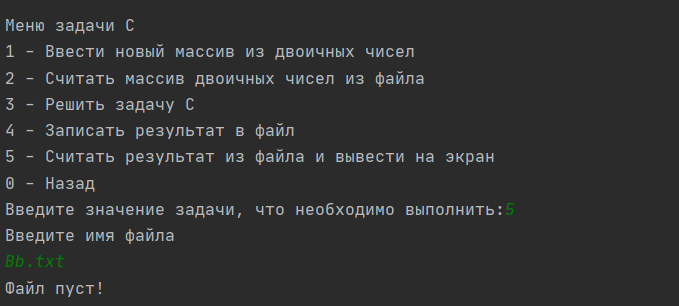


Рисунок 6.41 – Результат теста 32

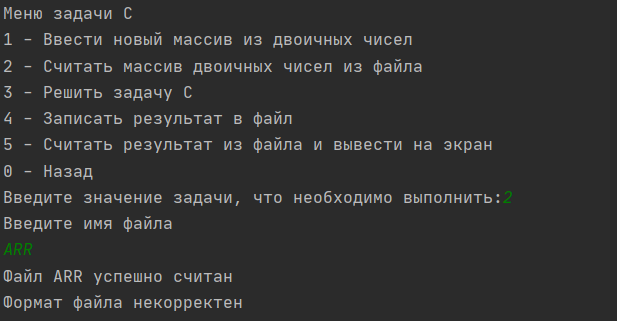


Рисунок 6.42 – Результат теста 33

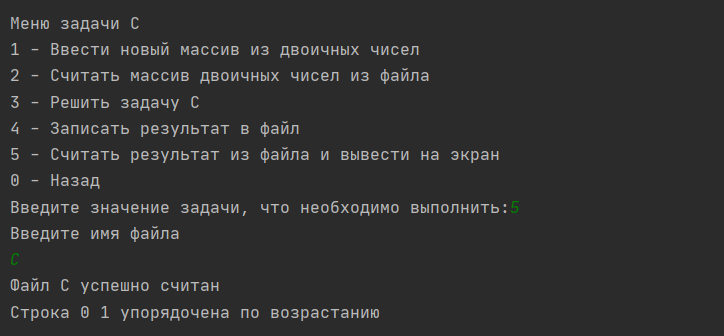


Рисунок 6.43 – Результат теста 34

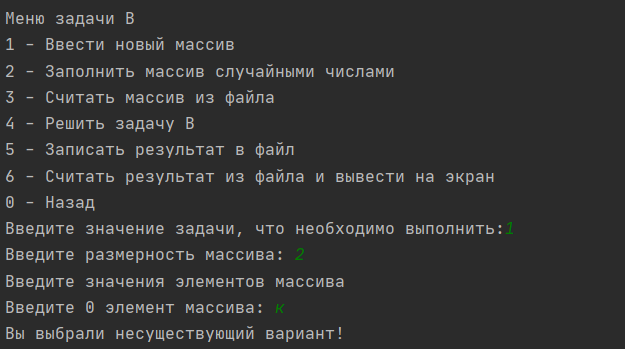


Рисунок 6.43 – Результат теста 35

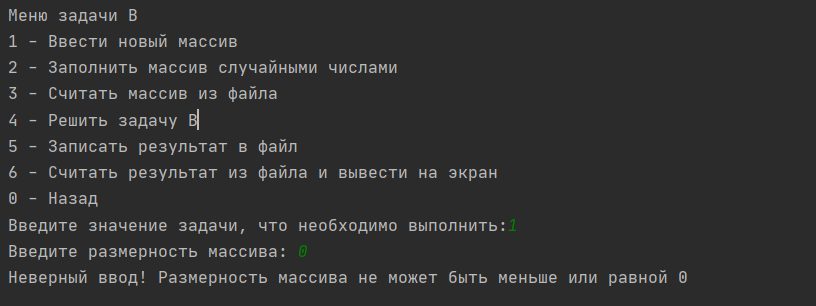


Рисунок 6.44 – Результат теста 36

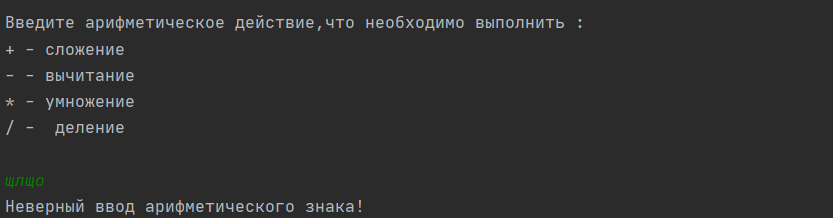


Рисунок 6.45 – Результат теста 37

7. Исходный код

**Класс Main**

package com.company;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

Menu.Start();

}

}

**Класс Menu**

package com.company;  
  
import java.io.IOException;  
import java.io.OutputStream;  
import java.util.Scanner;  
  
public class Menu {  
  
  
 public static void Start() {  
 *mainMenu*();  
 }  
 public static String inputFileName()  
 {  
 Scanner in = new Scanner(System.*in*);  
 String filename = "";  
 System.*out*.println("Введите имя файла");  
 filename = in.nextLine();  
 while (filename=="") {  
 filename = in.nextLine();  
 }  
 return filename;  
  
 }  
 private static void mainMenu(){  
 Scanner in = new Scanner(System.*in*);  
 int choice = -1;  
 do {  
 System.*out*.println("Главное меню");  
 System.*out*.println("1 - Решить задачу А");  
 System.*out*.println("2 - Решить задачу В");  
 System.*out*.println("3 - Решить задачу С");  
 System.*out*.println("0 - Выход");  
  
 int Choice = InputDataClass.*userInput*("Введите значение задачи, что необходимо выполнить:");  
 switch (Choice) {  
 case 1: {  
 *menuTaskA*();  
  
 break;  
 }  
 case 2: {  
 *menuTaskB*();  
 break;  
 }  
 case 3: {  
 *menuTaskC*();  
 break;  
 }  
 case 0:  
 {  
 choice = 0;  
 break;  
 }  
 default: {  
 System.*out*.println("Вы выбрали несуществующий вариант!");  
 }  
  
 }  
 }while (choice != 0);  
  
 }  
  
 private static void menuTaskB() {  
 int[] resultTaskB = new int[]{};  
 int[] Arr= new int[]{};  
 try {  
 Scanner in = new Scanner(System.*in*);  
 int choice = -1;  
 do {  
 System.*out*.println("\nMеню задачи В");  
 System.*out*.println("1 - Ввести новый массив");  
 System.*out*.println("2 - Заполнить массив случайными числами");  
 System.*out*.println("3 - Считать массив из файла");  
 System.*out*.println("4 - Решить задачу В");  
 System.*out*.println("5 - Записать результат в файл");  
 System.*out*.println("6 - Считать результат из файла и вывести на экран");  
 System.*out*.println("0 - Назад");  
  
 int Choice = in.nextInt();  
 switch (Choice) {  
 case 1: {  
 Arr = InputDataClass.*console*();  
 break;  
 }  
 case 2: {  
 Arr = InputDataClass.*generateRandomArr*();  
 break;  
 }  
 case 3: {  
 String str = InputOutputClass.*FromTxtFile*(*inputFileName*());  
  
 String[] s = str.split(" ");  
 Arr = new int[s.length];  
 for (int i = 0; i < s.length; i++) {  
 Arr[i] = Integer.*parseInt*(s[i]);  
 }  
  
 break;  
 }  
 case 4: {  
 if(Arr.length!=0) {  
  
 resultTaskB = InputDataClass.*completeTaskB*(Arr);  
 }  
  
 else{  
 System.*out*.println("Исходный массив пуст!");  
 }  
 break;  
 }  
 case 5: {  
  
 if(resultTaskB.length!=0) {  
 InputOutputClass.*inputByte*(resultTaskB,*inputFileName*());  
  
 }  
 else{  
 System.*out*.println("Задача не решена!");  
 }break;  
 }  
 case 6: {  
  
 int[] answer = InputOutputClass.*outputByte*(*inputFileName*());  
 for (int i = 0; i < answer.length; i++) {  
 System.*out*.print(answer[i] + " ");  
 }  
 break;  
 }  
 case 0: {  
 *mainMenu*();  
 break;  
 }  
 default: {  
 System.*out*.println("Вы выбрали несуществующий вариант!");  
 }  
  
 }  
 } while (choice != 0);  
 }  
 catch (NullPointerException e)  
 {  
 System.*out*.println("Файл пуст");  
 *menuTaskB*();  
 }  
 catch (NumberFormatException e){  
 System.*out*.println("Формат файла некорректен");  
 *menuTaskB*();  
 }  
 }  
 private static void menuTaskA(){  
 String taskA = "";  
 boolean flag = false;  
 Scanner in = new Scanner(System.*in*);  
 int choice = -1;  
 do {  
 System.*out*.println("\nМеню задачи А");  
 System.*out*.println("1 - Решить задачу А");  
 System.*out*.println("2 - Записать результат в файл");  
 System.*out*.println("3 - Выввести результат на экран");  
 System.*out*.println("0 - Назад");  
  
 int Choice = in.nextInt();  
 switch (Choice) {  
 case 1: {  
 TaskA a = new TaskA();  
 taskA = a.getTaskA();  
 flag = true;  
 break;  
 }  
 case 2: {  
 if (flag) {  
 InputOutputClass.*ToTxtFile*(taskA, *inputFileName*());  
 } else {  
 System.*out*.println("Задача не решена!");  
 }  
 break;  
 }  
 case 3: {  
 String str = InputOutputClass.*FromTxtFile*(*inputFileName*());  
 if(str==null)  
 {  
 System.*out*.println("Файл пуст!");  
 }  
 else {  
 System.*out*.print(str);  
 }  
  
 break;  
 }  
 case 0: {  
 *mainMenu*();// назад  
 break;  
 }  
 default: {  
 System.*out*.println("Вы выбрали несуществующий вариант!");  
 }  
  
 }  
 } while (choice != 0);  
 }  
 private static void menuTaskC() {  
 String resultTaskC = "";  
 boolean flag = false;  
 try {  
 Scanner in = new Scanner(System.*in*);  
 int choice = -1;  
 do {  
 System.*out*.println("\nMеню задачи C");  
 System.*out*.println("1 - Ввести новый массив из двоичных чисел");  
 System.*out*.println("2 - Считать массив двоичных чисел из файла");  
 System.*out*.println("3 - Решить задачу С");  
 System.*out*.println("4 - Записать результат в файл");  
 System.*out*.println("5 - Считать результат из файла и вывести на экран");  
 System.*out*.println("0 - Назад");  
  
 int Choice = in.nextInt();  
 switch (Choice) {  
 case 1: {  
 resultTaskC = InputDataClass.*userInputBinaryNum*();  
 System.*out*.println(resultTaskC);  
 break;  
 }  
 case 2: {  
 resultTaskC = InputOutputClass.*FromTxtFile*(*inputFileName*());  
 break;  
 }  
 case 3: {  
 if (resultTaskC!="") {  
 flag = true;  
 TaskC cls = new TaskC(resultTaskC);  
  
 resultTaskC = cls.getTaskC();  
 }  
 else{System.*out*.println("Нет входных данных!");}  
 break;  
 }  
 case 4: {  
 if(flag) {  
 InputOutputClass.*ToTxtFile*(resultTaskC,*inputFileName*());  
 }  
 else{  
 System.*out*.println("Задача не решена!");  
 }break;  
  
 }  
 case 5: {  
 if(flag) {

if (InputOutputClass.FromTxtFile(inputFileName())!=null) {

resultTaskC = InputOutputClass.FromTxtFile(inputFileName());

if(resultTaskC!=null)

System.out.println(resultTaskC);

}

else {

throw new NullPointerException();

}

}

else{

System.out.println("Задача не решена!");

} break;  
 }  
 case 0: {  
 *mainMenu*();  
 break;  
 }  
 default: {  
 System.*out*.println("Вы выбрали несуществующий вариант!");  
 }  
  
 }  
 } while (choice != 0);  
 }  
 catch (NullPointerException e)  
 {  
 System.*out*.println("Файл пуст");  
 *menuTaskC*();  
 }  
 catch (NumberFormatException e){  
 System.*out*.println("Формат файла некорректен");  
 *menuTaskC*();  
 }  
 }  
}

**Класс InputDataClass**

package com.company;  
  
import java.util.Random;  
import java.util.Scanner;  
  
public class InputDataClass {  
  
  
 public static String userInputBinaryNum(){  
 Scanner scanner = new Scanner(System.*in*);  
  
 int size = *userInput*("Введите размерность массива ");  
 while (size<=1)  
 {  
 System.*out*.println("Введите значение размерности массива больше 1\n");  
 size = *userInput*("Введите размерность массива ");  
 }  
 String bin = "";  
 int i=0;  
 while (i < size) {  
 String binaryNumber = "";  
 System.*out*.print("Enter a Binary Number: --> ");  
 binaryNumber = scanner.nextLine();  
 if (!binaryNumber.matches("[01]{1,}")) {  
 System.*err*.println("Invalid Binary Number Entered! Try Again...");  
 binaryNumber = scanner.nextLine();  
  
 } else {  
 bin += binaryNumber + " ";  
 i++;  
 }  
 }  
 bin = bin.substring(0, bin.length()-1);  
 return bin;  
 }  
  
 public static int userInput(String message) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.print(message);  
 int i;  
 while (true) {  
 try {  
 i = Integer.*parseInt*(sc.next());  
 break;  
 } catch (NumberFormatException e) {  
 System.*out*.println("Вы выбрали несуществующий вариант!");  
 }  
 }  
 return i;  
 }  
  
 public static int[] completeTaskB (int[] Arr){  
  
 Scanner input = new Scanner(System.*in*);  
  
 int num = *userInput*("Введите число,с которым необходимо выполнить арифметическое действие: ");  
  
 while (num == 0) {  
 num = *userInput*("Неверный ввод! Попробуйте ещё раз\n");  
 }  
  
 System.*out*.println("\nВведите арифметическое действие,что необходимо выполнить : " + "\n" +  
 "+ - сложение" + "\n" +  
 "- - вычитание" + "\n" +  
 "\* - умножение" + "\n" +  
 "/ - деление" + "\n");  
  
 String sign;  
 sign = input.nextLine();  
  
 while (!sign.equals("+") && !sign.equals("-") && !sign.equals("\*") && !sign.equals("/")) {  
 System.*out*.println("Неверный ввод арифметического знака!");  
 sign = input.nextLine();  
  
 }  
 TaskB first = new TaskB(Arr,num,sign);  
  
 int[] B = first.getTaskB();  
 return B;  
  
 }  
 public static int[] console ()  
 {  
 int size = *userInput*("Введите размерность массива: ");  
  
 while (size <= 0) {  
 size = *userInput*("Неверный ввод! Размерность массива не может быть меньше или равной 0\n");  
 }  
  
 int array[] = new int[size];  
 System.*out*.println("Введите значения элементов массива");  
 for (int i = 0; i < size; i++) {  
 array[i] = *userInput*("Введите " + (i) + " элемент массива: ");  
 }  
  
 return array;  
 }  
 public static int[] generateRandomArr()  
 {  
  
 int[] array = new int[*userInput*("Введите размерность массива")];  
 Random random = new Random();  
  
 for (int i = 0; i < array.length; i++) {  
 array[i] = random.nextInt((40 - 1) + 1) + 2;  
 }  
  
 return array;  
 }  
}

Класс TaskA

package com.company;  
  
import java.io.\*;  
  
public class TaskA {  
  
 private String taskA;  
  
 public TaskA(){ //Конструктор с параметрами  
 this.taskA = completeTaskA();  
 }  
 public String getTaskA(){  
 return this.taskA;  
 }  
  
 private String completeTaskA() {  
 int[] array = new int[50];  
  
 array[0] = array[1] = 1;  
  
 int i = 0;  
  
 while (i < 50 - 2) {  
 array[i + 2] = array[i + 1] + array[i];  
  
 i++;  
 }  
 i = 0;  
  
 String answerA = "";  
 while (i < 10) {  
 if (array[i] % 2 == 0) {  
 answerA+= "\*" + array[i] + " ";  
 } else {  
 answerA+=array[i] + " ";  
 }  
 i++;  
 }  
 System.*out*.println("Задача А решена");  
 answerA ="Числовой ряд Фибоначи: "+ answerA;  
 return answerA;  
 }  
 }

**Класс TaskB**

package com.company;  
public class TaskB {  
 private int[] taskB;  
 public TaskB(int[] Arr, int operand, String sign){ //Конструктор с параметрами  
 this.taskB = completeTaskB(Arr,operand,sign);  
 }  
  
 public int [] getTaskB() {  
 return this.taskB;  
 }  
  
 private int[] completeTaskB(int[] Arr,int operand, String sign) {  
  
 switch (sign) {  
 case "+":  
 for (int i = 0; i < Arr.length; i++) {  
 Arr[i] = Arr[i] + operand;  
 }  
 break;  
 case "-":  
 for (int i = 0; i < Arr.length; i++) {  
 Arr[i] = Arr[i] - operand;  
 }  
 break;  
 case "\*":  
 for (int i = 0; i < Arr.length; i++) {  
 Arr[i] = Arr[i] \* operand;  
 }  
 break;  
 case "/":  
 for (int i = 0; i < Arr.length; i++) {  
 Arr[i] = Arr[i] / operand;  
 }  
 break;  
 }  
 for (int i = 0; i < Arr.length; i++) {  
 System.*out*.println(Arr[i]);  
 }  
  
 return Arr;  
 }  
  
}

**Класс TaskC**

package com.company;  
  
public class TaskC {  
 private String taskC;  
  
 public TaskC(String binNumStr){ //Конструктор с параметрами  
 this.taskC = completeTaskC(binNumStr);  
 }  
  
 private static int binaryCodeToDecimal(String str) {  
 char[] chars = str.toCharArray();  
 int result = 0;  
 int multiply = 1;  
 for (int i = str.length() - 1; i >= 0; i--) {  
 if (chars[i] == '1') {  
 result += multiply;  
 }  
 multiply \*= 2;  
 }  
 return result;  
 }  
  
 public String getTaskC() {  
 return this.taskC;  
 }  
  
 private String completeTaskC(String binNumStr) {  
 boolean up = true;  
 boolean down = true;  
 String Answer = "";  
 String[] binNumStrArr = binNumStr.split(" ");  
  
 int[] binToDecimal = new int[binNumStrArr.length];  
  
 for (int i = 0; i < binToDecimal.length; i++) {  
 binToDecimal[i] = *binaryCodeToDecimal*(binNumStrArr[i]);  
 }  
  
 for (int i = 0; i < binToDecimal.length - 1; i++) {  
 if (binToDecimal[i] > binToDecimal[i + 1] && up) {  
 up = false;  
 }  
  
 }  
 for (int i = 0; i < binToDecimal.length - 1; i++) {  
 if (binToDecimal[i] < binToDecimal[i + 1] && down) {  
 down = false;  
 }  
 }  
 if (up && !down) {  
 Answer = "Строка " + binNumStr + " упорядочена по возрастанию";  
 }  
 if (!up && down) {  
 Answer = "Строка " + binNumStr + " упорядочена по убыванию";  
 }  
 if (!up && !down) {  
 Answer = "Строка " + binNumStr + " не упорядочена";  
 }  
 return Answer;  
 }  
  
}

**Класс InputOutputClass**

package com.company;  
  
import java.io.\*;  
  
public class InputOutputClass {  
  
 public static int[] outputByte(String filename){  
 int [] tempArr = null;  
 try(DataInputStream dis = new DataInputStream(new FileInputStream(filename))) {  
  
 int size = dis.readInt();  
 tempArr = new int[size];  
 for(int i =0;i<size;i++)  
 {  
 tempArr[i] = dis.readInt();  
 }  
 System.*out*.println("Запись успешно считана из файла: "+filename);  
 }catch (FileNotFoundException ex){

System.out.println("Файл не найден");

}catch (IOException ex){

System.out.println("Файл подозрителен...");

}catch (OutOfMemoryError e) {

System.out.println("Формат файла некорректен");

}catch (NegativeArraySizeException e){

System.out.println("Формат файла некорректен");

}

return tempArr;  
  
 }  
  
 public static String FromTxtFile(String filename){  
 String s = "";  
 try(BufferedReader br = new BufferedReader(new FileReader(filename))) {  
 String r = br.readLine();  
 System.*out*.println("Файл "+filename+" успешно считан");  
 s=r;  
 } catch (FileNotFoundException e)  
 {  
 System.*out*.println("Файл не найден!");  
 } catch (IOException e){  
 System.*out*.println("Ошибка чтения");  
 }  
 return s;  
 }  
 public static void ToTxtFile(String task,String filename)  
 {  
 try(BufferedWriter br = new BufferedWriter(new FileWriter(filename))) {  
 br.write(task);  
 System.*out*.println("Запись успешно сохранена в файл: "+ filename);  
 } catch (FileNotFoundException e) {  
 System.*out*.println("Файла с именем "+filename+ " не найдено");  
 } catch (IOException e) {  
 System.*out*.println("Ошибка записи");  
 }  
 }  
 public static void inputByte(int[] arrInt,String filename) { //запись бинарника  
 try(DataOutputStream dos = new DataOutputStream(new FileOutputStream(filename))) {  
  
 dos.writeInt(arrInt.length);  
 for(int i =0;i<arrInt.length;i++)  
 {  
 dos.writeInt(arrInt[i]);  
 }  
 System.*out*.println("Запись успешно сохранена в файл: "+ filename);  
 }catch (FileNotFoundException ex){  
 System.*out*.println("Файл не найден");  
 }catch (IOException ex){  
 System.*out*.println("Ошибка записи");  
 }  
 }  
  
}

Список использованных источников

1. Объектно-ориентированное программирование. Метод. указания по

выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск : Изд-во

ИРНИТУ, 2017. – 24 c.