Министерство образования Республики Беларусь

Белорусский Национальный Технический Университет

Факультет информационных технологий и робототехники

Кафедра «Программное обеспечение информационных систем

и технологий»

**Отчёт**

по лабораторной работе №06

по дисциплине ***«Объектно-ориентированные технологии***

***программирования и стандарты проектирования»***

тема: «Одномерные массивы в Java»

|  |  |  |
| --- | --- | --- |
| Исполнитель: |  | студент группы 10701119  Прадед Кирилл Витальевич |
| Преподаватель: |  | Иванченко Виктор Викторович |

2020-2021 учебный год

**ЛАБОРАТОРНАЯ РАБОТА #06**

Одномерные массивы в Java

Цель работы

Научиться работать с одномерными массивами в Java и закрепить приобретённые навыки на примере разработки интерактивных консольных Java приложений.

**Tребования**

1) Необходимо спроектировать и реализовать алгоритмы решения заданий и UML-диаграмму взаимодействия классов и объектов разрабатываемой программной системы с отображением всех связей (отношений) между классами и объектами.

2) Каждый студент выбирает по одному уникальному индивидуальному заданию.

3) Основные классы системы должны быть самодостаточными, т.е. не зависеть, к примеру, от консоли! Любые типы отношений между классами должны применяться обосновано и лишь тогда, когда это имеет смысл.

4) На базе спроектированной программной системы реализуйте простейшее интерактивное консольное приложение. Используйте при реализации архитектурный шаблона проектирования Model-View-Controller, MVC.

5) Создаваемые классы необходимо грамотно разложить по соответствующим пакетам, которые должны иметь «адекватные» названия и быть вложены в указанные стартовые пакеты: by.bntu.fitr.nameofstudent.javalabs.lab06.

6) Попытайтесь реализовать все средства инициализации при создании соответствующих бизнес-объектов программной системы.

7) При выполнения задания необходимо по максимуму пытаться разрабатывать универсальный, масштабируемый, легко поддерживаемый и читаемый код.

8) Если логически не подразумевается или в задании иного не указано, то входными и выходными данными являются вещественные числа (числа с плавающей запятой).

9) В соответствующих компонентах бизнес-логики необходимо предусмотреть «защиту от дурака».

10)Для генерирования случайных чисел воспользуйтесь методами объекта класса java.util.Random, а для реализации ввода данных с консоли (терминала) – соответствующими методами объекта класса java.util.Scanner. 11)Программа должна обязательно быть снабжена комментариями, в которых необходимо указать краткое предназначение программы, номер лабораторной работы и её название, версию программы, ФИО разработчика, название бригады (если есть), номер группы и дату разработки. Исходный текст классов и демонстрационной программы рекомендуется также снабжать поясняющими краткими комментариями.

12)Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом для взаимодействия с пользователем. Интерфейс программы и комментарии в коде должны быть на английском языке.

**Код бизнес-логики программ прикреплен в приложении A. Полный код программ можно найти на GitHub -**

**https://github.com/kirillpraded/JavaLabs**

**Задание 1**

1) В векторе, состоящем из n вещественных элементов, вычислить: сумму отрицательных элементов вектора и произведение элементов вектора, расположенных между максимальным и минимальным элементами.

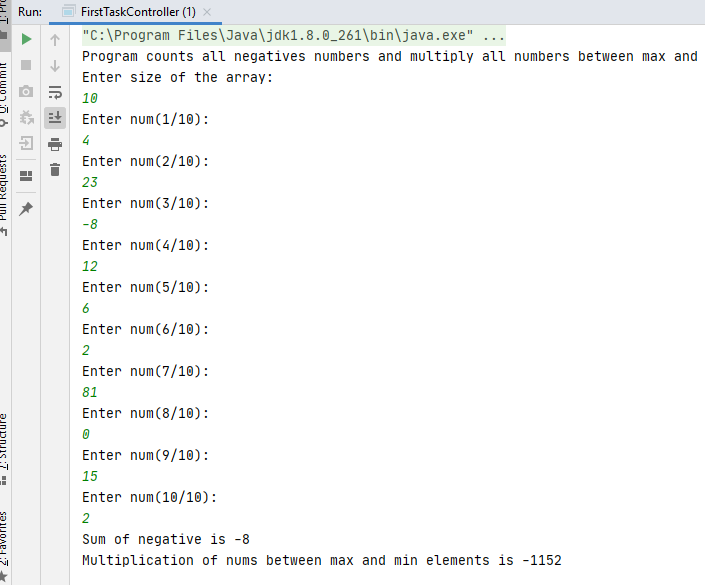


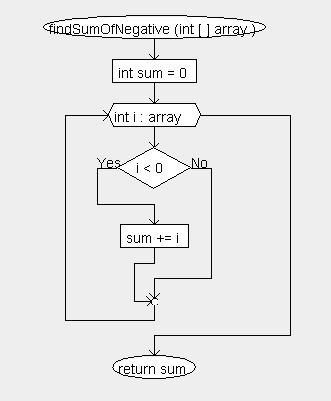
Рис. 1 – результат выполнения программы.

*Приложение А. Код программы.*

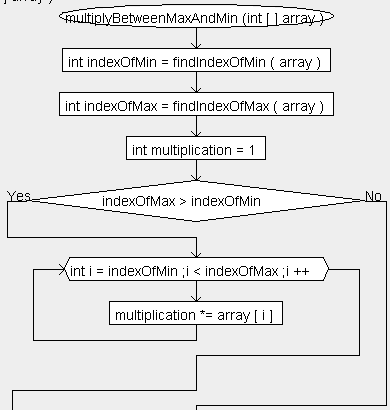
**В приложении представлен только код бизнес-логики. Код слоя view и controller можно найти на GitHub -**

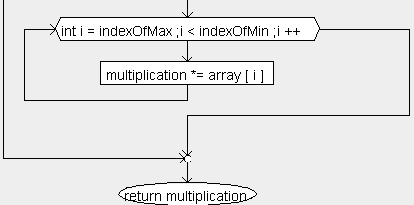
**https://github.com/kirillpraded/JavaLabs**

public static int findSumOfNegative(int[] array) {  
 int sum = 0;  
 for (int i : array) {  
 if (i < 0) {  
 sum += i;  
 }  
 }  
 return sum;  
}  
  
public static int multiplyBetweenMaxAndMin(int[] array) {  
 int indexOfMin = *findIndexOfMin*(array);  
 int indexOfMax = *findIndexOfMax*(array);  
  
 int multiplication = 1;  
  
 if (indexOfMax > indexOfMin) {  
 for (int i = indexOfMin; i < indexOfMax; i++) {  
 multiplication \*= array[i];  
 }  
 } else {  
 for (int i = indexOfMax; i < indexOfMin; i++) {  
 multiplication \*= array[i];  
 }  
 }  
 return multiplication;  
}  
  
private static int findIndexOfMin(int[] array) {  
 int index = 0;  
 int value = array[0];  
 for (int i = 0; i < array.length; i++) {  
 if (array[i] < value) {  
 value = array[i];  
 index = i;  
 }  
 }  
 return index;  
}  
  
private static int findIndexOfMax(int[] array) {  
 int index = 0;  
 int value = array[0];  
 for (int i = 0; i < array.length; i++) {  
 if (array[i] > value) {  
 value = array[i];  
 index = i;  
 }  
 }  
 return index;  
}

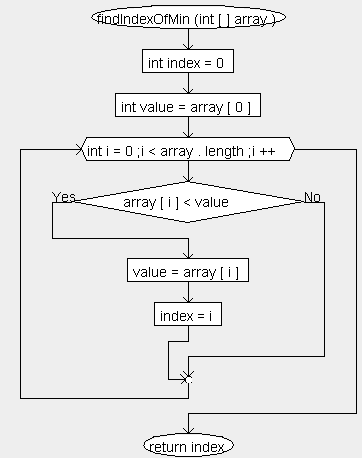


Блок-схема метода findSumOfNegative(int[] array)

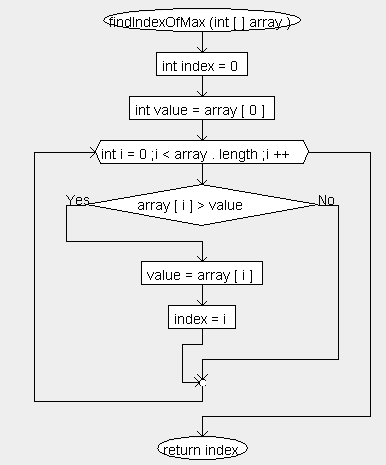




Блок-схема метода multiplyBetweenMaxAndMin(int[] array)



Блок-схема метода findIndexOfMin(int[] array)



Блок-схема метода findIndexOfMax(int[] array)

**Контрольные вопросы**

1) Что такое массив?

Структура данных, хранящая набор значений одного типа, идентифицируемых по индексу.

2) Зачем нужно при проектировании и реализации программы использовать массивы?

Массивы удобная структура данных, их можно использовать для хранения и/или передачи объектов одного типа.

3) Какие существуют типы массивов в Java?

Одномерные/многомерные/рваные

4) Чем массивы в Java отличаются от массивов в C/C++/C#?

Массивы являются ссылочным типом данных, даже если это массивы примитивных типов. в массиве java хранится ссылка на объект.

5) Опишите особенности Java-массивов и его свойства.

Массивы являются ссылочным типом данных, даже если это массивы примитивных типов. в массиве java хранится ссылка на объект.

Работать с ними надо как с любым ссылочным типом, даже если это массив примитивов

6) Почему в Java массивы занимают особое место по сравнению с другими контейнерами? Опишите основные преимущества и недостатки Java-массивов по сравнению с существующими библиотечными контейнерами: эффективность, типизация и возможность хранить примитивы.

Потому что на основе массивов построены такие структуры как, например, arraylist.

Преимущества: простота в использовании, скорость работы, объем занимаемой памяти

Недостатки: статический размер

В хранений примитивов в сравнении с другими структурами не имеет, так как благодаря классам-оберткам и коллекции могут хранить классы-обертки.

7) Каково время доступа к элементам массива?

O(1)

8) Почему нумерация элементов массива начинается с нуля?

потому что индекс массива представляет собой не что иное, как смещение памяти, поэтому первый элемент массива находится в памяти, на которую он уже указывает

9) Какие существуют нотации в Java объявления переменной типа массива?

Массив может объявляться так:

* int[] array = new int[size];
* int array[] = new int[size];

10) Способы объявления, конструирования и инициализации массивов в Java?

* int[] array = new int[size];
* int array[] = new int[size];
* int[] array = {1,23,4,5,6};

11) Если массив – это объект, а все объекты создаются на базе описанного класса, то где хранятся классы-массивов?

Метод getClass возвращает class [I

byte[] me = new byte[1];

int[] me2 = new int[1];

double[] me3 = new double[1];

float[] me4 = new float[1];

String[] me5 = new String[1];

Integer[] me6 = new Integer[1];

System.out.println(me.getClass());

System.out.println(me2.getClass());

System.out.println(me3.getClass());

System.out.println(me4.getClass());

System.out.println(me5.getClass());

System.out.println(me6.getClass());

class [B

class [I

class [D

class [F

class [Ljava.lang.String;

class [Ljava.lang.Integer;

Element Type Encoding

boolean Z

byte B

char C

class or interface Lclassname;

double D

float F

int I

long J

short S

Javadoc об этом - https://docs.oracle.com/javase/7/docs/api/java/lang/Class.html#getName%28%29

12) Как передать массив в качестве входного параметра в метод?

(Int[] array)

Также есть вариант когда мы например хотим передать сколько-то переменных типа инт, но не уверены сколько, но точно знаем что они будут идти как простые переменные а не как массив

Тогда можно объявить метод так:

Method(int… nums);

Важно, чтобы эта констркукция была последним параметром который принимает метод. В методе мы можем работать с nums как с массивом intов.

13) Как возвратить из метода значение в виде массива?

Public static int[] getInts(){

Return new int[5];

}

14) Как инициализировать элементы одномерного массива во время объявления ссылочной переменной?

* int[] array = {1,23,4,5,6};

С ссылочными типами данных работает также, но необходимо каждый элемент создавать через new.

15) Как инициализировать элементы одномерного массива, который создаётся «на лету» в виде анонимного объекта и либо передаётся в качестве параметра в вызываемый метод, либо возвращается как результат работы метода?

Также можно объявить как и в предыдущем ответе.

16) Массивы и примитивные типы данных.

17) Массивы и ссылочные типы данных.

18) Что физически содержит (хранит) внутри себя массив?

Ссылки на объекты в хипе

19) Какие могут возникнуть исключения при работе с массивами и на что они указывают?

Indexoutofbound – обращение к индексу, которого не существует(вышел за границы массива)

20) Какие существуют способы копирования (клонирования) массивов (элементарных и ссылочных типов)?

С помощью утилитного класса Arrays

.clone()

вручную

21) Что такое поверхностное и глубокое копирование объектов?

Поверхностное – просто передавая ссылки, то есть на выходе мы получим два объекта, в которых внутренние поля – одинаковые ссылки. Изменив один мы изменим и второй

Глубокое – полное копирование объекта, со всеми полями. При глубоком копировании изменив один объект – второй не изменится.

22) Что такое поверхностное и глубокое копирование массивов?

То же самое, что и объектов.

23) Для чего используется утилитный класс java.util.Arrays?

Содержит множество вспомогательных методов, от сортировки, до заполнения и копирования части массива.

24) Как сравнить два массива?

Arrays.equals

Objects.*equals*(new int[5], new int[5]) – не вызывает никаких исключений но рекомендует использовать Arrays для массивов.

25) Как отсортировать элементы массивов?

Arrays.sort – самый простой способ, там реализована quick sort, поэтому сложность выполнения в среднем будет O(nlogn). Можно вручную реализовать этот способ сортировки или Merge sort, но это не имеет смысла, тк производительнее алгоритма не получится сделать.

26) Как произвести поиск элемента в массиве?

Можно использовать Arrays.binarySearch(\*)

Или реализовать бинарный поиск вручную

27) Как объявить метод с переменным количество передаваемых параметров? Как реализована логика данного метода?

Также есть вариант когда мы например хотим передать сколько-то переменных типа инт, но не уверены сколько, но точно знаем что они будут идти как простые переменные а не как массив

Тогда можно объявить метод так:

Method(int… nums);

Важно, чтобы эта констркукция была последним параметром который принимает метод. В методе мы можем работать с nums как с массивом intов

28) Каковы основные правила объявления методов с переменным количеством параметров?

Важно, чтобы эта констркукция была последним параметром который принимает метод. В методе мы можем работать с nums как с массивом intов

29) Как использовать расширенный цикл for (ещё его иногда называют for-each), который появился с JDK 5.0?

Для массивов или коллекций

For(int I : array){}

30) Чего нельзя сделать с помощью расширенного цикла for в Java?

Получить индекс элемента.