

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Магнитогорский государственный технический университет им. Г. И.
Носова»

(ФГБОУ ВО «МГТУ им. Г.И. Носова»)

Кафедра вычислительной техники и программирования

Лабораторная работа №3

**по дисциплине «Метрология и стандартизация программного
обеспечения»**

**название лабораторной работы: «Оценка функциональной
сложности программы»**

Исполнитель: Варламов М.Ня., студент 3 курса, группа АВб-19-1

Руководитель: Сибилева Н.С., ст. преподаватель каф. ВТиП

Магнитогорск, 2022

Оглавление

1. Задание.	3
2. Исходный код.	3
3. Анализ качества программного кода с помощью метрик.	4
4. Выводы.	9

1. Задание.

Для натурального числа, заданного с клавиатуры получить новое число, дописав к нему цифру s (также введенную с клавиатуры) в начале и в конце.

2. Исходный код.

Программа написана на языке Java и представлена далее.

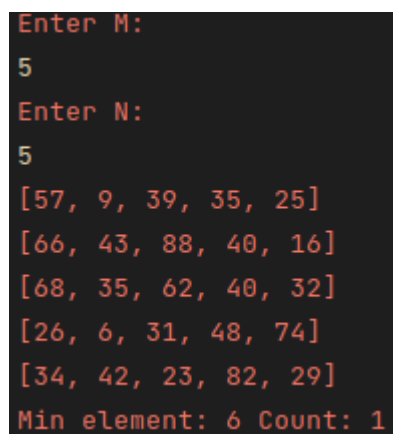
```
1. import java.util.Arrays;
2. import java.util.Scanner;
3.
4. public class Lab3 {
5.     public static void main(String[] args) {
6.         int M, N;
7.         Scanner scanner = new Scanner(System.in);
8.         System.err.println("Enter M:");
9.         M = scanner.nextInt();
10.        System.err.println("Enter N:");
11.        N = scanner.nextInt();
12.
13.        int mass[][] = new int[M][N];
14.        randomizeMass(mass);
15.        for (int[] m : mass)
16.            System.err.println(Arrays.toString(m));
17.        int minElement = findMinElement(mass);
18.        int minElementCount = findElementCount(mass, minElement);
19.        System.err.println("Min element: " + minElement + " Count: " +
minElementCount);
20.    }
21.
22.    private static int findElementCount(int[][] mass, int element) {
23.        int count = 0;
24.        for (int[] ints : mass)
25.            for (int anInt : ints)
26.                if (anInt == element)
27.                    ++count;
28.
29.        return count;
30.    }
```

```

31.
32.     private static int findMinElement(int[][] mass) {
33.         int min = Integer.MAX_VALUE;
34.         for (int[] ints : mass)
35.             for (int anInt : ints)
36.                 if (anInt < min)
37.                     min = anInt;
38.
39.         return min;
40.     }
41.
42.     public static void randomizeMass(int[][] mass) {
43.         for (int i = 0; i < mass.length; i++) {
44.             for (int j = 0; j < mass[i].length; j++) {
45.                 mass[i][j] = (int) (Math.random() * 100);
46.             }
47.         }
48.     }
49. }

```

Результат выполнения данной программы представлен на рисунке 1.



```

Enter M:
5
Enter N:
5
[57, 9, 39, 35, 25]
[66, 43, 88, 40, 16]
[68, 35, 62, 40, 32]
[26, 6, 31, 48, 74]
[34, 42, 23, 82, 29]
Min element: 6 Count: 1

```

Рисунок 1 – Результат выполнения программы.

В качестве результата программа выводит 2 целых числа, которые являются минимальным элементом и количеством этого элемента в массиве.

3. Анализ качества программного кода с помощью метрик.

Далее следует таблица 1 с видами оценочных элементов.

Оценочный элемент	Расшифровка	Количество
j1	Количество внешних вводов данных пользователем	1
j2	Количество внешних выводов данных	3
j3	Количество внешних запросов	0
j4	Количество локальных внутренних логических файлов	0
j5	Количество внешних интерфейсных файлов	0

Таблица 1 – Виды оценочных элементов

Далее следует таблица 2 со сложностью данных в зависимости от их количества.

Количество элементов данных	Низкая сложность	Средняя сложность	Высокая сложность
Внешние вводы	1-4	5-19	>19
Внешние выводы	1-4	5-19	>19
Внешние запросы	1-4	5-19	>19
Внутренние логические файлы	1-19	20-50	>50
Внутренние интерфейсные файлы	1-19	20-50	>50

Таблица 2 – Сложность данных в зависимости от их количества

Далее следует таблица 3 для расчета количества функциональных указателей.

Характеристика	Количество с учетом сложности			Итого
	Низкая	Средняя	Высокая	

Внешние вводы	$l*3 = 3$	$u*4 =$	$u*5 =$	3
Внешние выводы	$3*4 = 4$	$u*5 =$	$u*7 =$	12
Внешние запросы	$u*3 =$	$u*4 =$	$u*6 =$	0
Внутренние логические файлы	$u*7 =$	$u*10 =$	$u*15 =$	0
Внутренние интерфейсные файлы	$u*5 =$	$u*7 =$	$u*10 =$	0
Общее количество				15

Таблица 3 – Типовая таблица для расчета количества функциональных указателей

Далее следует таблица 4 с вопросами для расчета функциональных показателей и их влияния.

№	Вопрос	Коэффициенты регулировки сложности
1	Какое влияние имеет наличие средств передачи данных?	5
2	Какое влияние имеет распределенная обработка данных?	5
3	Какое влияние имеет распространенность используемой аппаратной платформы?	2
4	Какое влияние имеет критичность к требованиям производительности и ограничению времени ответа?	1
5	Какое влияние имеет частота транзакций?	1
6	Какое влияние имеет ввод данных в режиме реального времени?	5
7	Какое влияние имеет эффективность работы конечного пользователя?	1
8	Какое влияние имеет оперативное обновление локальных файлов в режиме реального времени?	1
9	Какое влияние имеет скорость обработки данных (вычислений)?	4
10	Какое влияние имеет количество и категории пользователей?	1

№	Вопрос	Коэффициенты регуливовки сложности
11	Какое влияние имеет легкость инсталляции?	1
12	Какое влияние имеет легкость эксплуатации?	2
13	Какое влияние имеет разнообразие условий применения?	1
14	Какое влияние имеет простота внесения изменений?	4

Таблица 4 – Вопросы для расчета функциональных показателей

Общее количество: 34

$$\text{Метрики } FP = \text{Общее количество} * (0,65 + 0.01 * \sum_{i=1}^{14} F_i) = 15 * (0,65 + 0.01 * 34) = 14.85$$

2.1 Для каждого реализованного модуля (функции) согласно методике определения типа связности рассчитать значение силы связности.

Связность модуля – это мера зависимостей частей данного модуля. Чем выше связность, тем лучше результат проектирования, «тем «черней» его ящик, тем меньше «ручек управления» на нем находится и тем проще эти «ручки»»

Далее следует таблица 5 с типами связностей и их силами.

№	Код	Описание	Тип связности	Сила
1.	<pre>public static void main(String[] args) { int M, N; Scanner scanner = new Scanner(System.in); System.err.println("Enter M:"); M = scanner.nextInt(); System.err.println("Enter N:"); N = scanner.nextInt(); }</pre>	Части модуля связаны по данным (работают с одной и той же структурой данных)	Коммуникативная	9

	<pre> int mass[][] = new int[M][N]; randomizeMass(mass); for (int[] m : mass) System.err.println(Arrays.toString(m)); int minElement = findMinElement(mass); int minElementCount = findElementCount(mass, minElement); System.err.println("Min element: " + minElement + " Count: " + minElementCount); } </pre>			
2.	<pre> private static int findElementCount(int[][] mass, int element) { int count = 0; for (int[] ints : mass) for (int anInt : ints) if (anInt == element) ++count; return count; } </pre>	Части модуля связаны по данным (работают с одной и той же структурой данных)	Информационная	9
3.	<pre> private static int findMinElement(int[][] mass) { int min = Integer.MAX_VALUE; for (int[] ints : mass) for (int anInt : ints) if (anInt < min) min = anInt; return min; } </pre>	Выходные данные одной части используются как входные данные в другой части модуля.	Информационная	9

Таблица 5 – Типы связанностей и их силы

3.1 Для каждого реализованного модуля (функции) согласно типу модуля по шкале сцепления, определить значение силы сцепления.

Сцепление – мера взаимозависимости модулей по данным, которую желательно уменьшать.

Далее следует таблица 6 с типами сцепления и их силами.

№	Код	Описание	Тип сцепления	Сила
1.	<pre> public static void main(String[] args) { int M, N; Scanner scanner = new Scanner(System.in); System.err.println("Enter M:"); M = scanner.nextInt(); System.err.println("Enter N:"); N = scanner.nextInt(); int mass[][] = new int[M][N]; randomizeMass(mass); for (int[] m : mass) System.err.println(Arrays.toString(m)); int minElement = findMinElement(mass); int minElementCount = findElementCount(mass, minElement); System.err.println("Min element: " + minElement + " Count: " + minElementCount); } </pre>	Модуль А явно управляет функционированием модуля В и С, посылая ему управляющие данные	Сцепление по управлению	4
2.	<pre> private static int findElementCount(int[][] mass, int element) { int count = 0; for (int[] ints : mass) for (int anInt : ints) if (anInt == element) ++count; return count; } </pre>	Модуль А вызывает модуль В. Все входные и выходные параметры вызываемого модуля – простые элементы данных	Сцепление по управлению	4
3.	<pre> private static int findMinElement(int[][] mass) { int min = Integer.MAX_VALUE; for (int[] ints : mass) for (int anInt : ints) if (anInt < min) min = anInt; } </pre>	Модуль А вызывает модуль С. Все входные и выходные параметры вызываемого модуля – простые элементы данных	Сцепление по данным	4

	return min;}			
--	--------------	--	--	--

4. Выводы.

В данной работе была реализована программу, которая подсчитывает количество минимальных элементов в целочисленной матрице размера $m \times n$, а затем выводит количество и значение минимального элемента на экран.

Так же были изучены метрики на основе функциональных указателей (также оценены их сложности и сделаны расчеты), определены коэффициенты регулировки сложности, для каждого реализованного модуля (функции) согласно методике определения типа связности рассчитаны значение силы связности и для каждого реализованного модуля (функции) согласно типу модуля по шкале сцепления, определены значение силы сцепления.