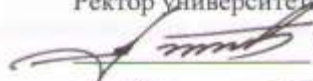




**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ**  
Брянский государственный технический университет

Утверждаю

Ректор университета

 О.Н. Федонин

« 14 » 07 2017г.

**МЕТРОЛОГИЯ  
И КАЧЕСТВО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

**ОЦЕНКА ХАРАКТЕРИСТИК ПРОГРАММ  
НА ОСНОВЕ ЛЕКСИЧЕСКОГО АНАЛИЗА.  
МЕТРИКИ ЧЕПИНА**

Методические указания  
к выполнению лабораторной работы №3  
для студентов очной формы обучения  
по направлениям подготовки  
09.03.01 «Информатика и вычислительная техника»  
09.03.04 «Программная инженерия»  
02.03.03 «Математическое обеспечение и администрирование  
информационных систем»

Брянск 2017

**УДК 004.01**

Метрология и качество программного обеспечения. Оценка характеристик программ на основе лексического анализа. Метрики Чепина [Электронный ресурс]: методические указания к выполнению лабораторной работы №3 для студентов очной формы обучения по направлению подготовки 09.03.01 «Информатика и вычислительная техника»; 09.03.04 «Программная инженерия»; 02.03.03 «Математическое обеспечение и администрирование информационных систем» – Брянск, 2017. - 23с.

Разработал

А.А. Азарченков,

канд. техн. наук, доц.

Рекомендовано кафедрой «Информатика и программное обеспечение»  
БГТУ (протокол № 7 от 01.06.2017г.)

**Методические издания публикуются в авторской редакции**

## 1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Мерой сложности понимания программ на основе входных и выходных данных является метрика Н. Чепина (Ned Chapin). Смысл метода, который предложил Чепин, состоит в оценке информационной прочности отдельно взятого программного модуля на основе результатов анализа характера использования переменных, входящих в состав списка ввода и вывода. Существует несколько модификаций метрики Чепина. Рассмотрим наиболее простой, но с точки зрения практического использования достаточно эффективный вариант этой метрики.

Все множество переменных, составляющих список ввода-вывода, разбивается на четыре функциональные группы:

*P* – вводимые переменные для расчетов и для обеспечения вывода. Примером такой переменной может служить используемая в программах лексического анализатора переменная, содержащая строку исходного текста программы — в этом случае сама переменная не модифицируется, а применяется только как контейнер для исходной информации;

*M* – модифицируемые, или создаваемые внутри программы, переменные. К таким переменным относится большинство традиционно применяемых переменных, декларируемых в программных модулях;

*C* – переменные, участвующие в управлении работой программного модуля (управляющие переменные). Такие переменные предназначены для передачи управления, изменения логики вычислительных процессов и т. д.;

*T* – не используемые в программе (так называемые паразитные) переменные. Такие переменные не принимают непосредственного участия в реализации процесса обработки информации, ради которого написана анализируемая программа, однако они заявлены в программном модуле. Такие переменные могут использоваться для выполнения промежуточных действий и не играют принципиальной роли в решении основной задачи.

В качестве особенности применения данной метрики следует назвать необходимость учета каждой переменной в каждой функциональной группе, поскольку каждая переменная может выполнять одновременно несколько функций.

Исходное выражение для определения метрики Чепина записывается в следующем виде:

$$Q = \alpha_1 \cdot P + \alpha_2 \cdot M + \alpha_3 \cdot C + \alpha_4 \cdot T$$

где  $\alpha_1, \alpha_2, \alpha_3, \alpha_4$  - весовые коэффициенты.

Весовые коэффициенты в расчетном выражении значения метрики применяются для отражения различного влияния на сложность программы каждой функциональной группы переменных. По мнению автора метрики наибольший вес, равный 3, должна иметь функциональная группа  $C$ , так как она непосредственно влияет на поток управления программы. Весовые коэффициенты остальных групп Чепин распределяет следующим образом:

$$\alpha_1 = 1;$$

$$\alpha_2 = 2;$$

$$\alpha_4 = 0,5.$$

Примечательно, что весовой коэффициент группы  $T$  не равен 0, несмотря на то, что часть используемых переменных может не применяться в программе. Это объясняется тем, что «паразитные» переменные сами по себе не увеличивают сложность потока данных программы, но очень часто затрудняют ее понимание.

Следует отметить, что метрика сложности программы Чепина также основана на анализе исходных текстов программ, что обеспечивает единый подход к автоматизации их расчета и может быть рассчитана с использованием специально разработанного программного анализатора.

## 2. ПРИМЕР ОЦЕНКИ ХАРАКТЕРИСТИК

### 2.1. Задача «Простые числа в матрице»

Дана целочисленная матрица размером  $N \times M$ . Вычислить и записать в одномерный массив количество простых чисел в каждом столбце матрицы. Размерность матрицы задается с клавиатуры, заполнение матрицы осуществляется посредством датчика случайных чисел. Разработать программу для решения задачи. На основе лексического анализа исходного текста программы определить значение метрики Чепина.

### 2.1.1 Реализация программы

Текст программы для реализации возможного алгоритма решения поставленной задачи представлен на языке C# (рис. 1).

Номера строк	Строки программы
1	using System;
2	namespace EX2
3	{
4	class Program
5	{
6	static void Main()
7	{
8	int n,m;
9	
10	int[,] a;
11	int[] b;
12	ConsoleKeyInfo клавиша;
13	int i,j,k,d;
14	bool p=false;
15	Random g;
16	do
17	{
18	Console.Clear();
19	Console.Write("Сколько строк: ");
20	n = int.Parse(Console.ReadLine());
21	Console.Write("Сколько столбцов: ");
22	m = int.Parse(Console.ReadLine());
23	g=new Random();
24	a = new int[n, m];
25	for (i = 0; i < n; i++)
26	for (j = 0; j < m; j++)
27	{
28	a[i, j] = int.Parse(g.Next(0,101));
29	}
30	
31	Console.WriteLine("\nИсходная матрица");
32	for (i = 0; i < n; i++, Console.WriteLine())
33	for (j = 0; j < m; j++)
34	Console.Write("{0,8:d}", a[i, j]);
35	b = new int[m];
36	
37	for (j = 0; j < m; j++)
38	{
39	for (i = k = 0; i < n; i++)
40	{
41	p = true;
42	for (d = 2; d < a[i,j]; d++)
43	if (a[i,j] % d == 0) p = false;
44	
45	
46	if (p) k++;
47	b[j] = k;
48	}
49	}
50	
51	
52	Console.WriteLine("\nКоличество простых");
53	for (i = 0; i < b.Length; i++)
54	Console.Write("{0,8:d}", b[i]);
55	
56	Console.WriteLine("\nДля выхода нажмите клавишу ESC");
57	клавиша = Console.ReadKey(true);
58	} while (клавиша.Key!= ConsoleKey.Escape);
59	
60	}
61	}
62	}

Рис. 1. Текст программы для простых чисел в матрице

### 2.1.2 Оценка характеристик программы

Рассмотрим текст программы для оценки ее качества с помощью метрики Чепина, которая позволяет оценить меру трудности понимания программы на основе входных и выходных данных. Все множество переменных, составляющих список ввода-вывода, разбивается на четыре функциональные группы:

$P$  – вводимые переменные для расчетов и для обеспечения вывода;

$M$  – модифицируемые, создаваемые внутри программы переменные;

$C$  – переменные, участвующие в управлении работой программного модуля (управляющие переменные);

$T$  – не используемые в программе переменные. Проанализируем исходный текст программы, чтобы определить ее характеристики для расчета метрики Чепина (табл. 1).

Таблица 1.

Результат анализа объявленных переменных

№ п/п	Наименования переменных	Номера строк
<b><math>P</math> (для расчетов и обеспечения вывода)</b>		
1	$m$	8
2	$n$	8
3	$a$	10
<b><math>M</math> (модифицируемые или создаваемые)</b>		
1	$i$	13
2	$j$	13
3	$k$	13
4	$d$	13
5	$b$	11
<b><math>C</math> (управляющие переменные)</b>		
1	$g$	15
2	$p$	14
3	клавиша	12
<b><math>T</math> (не используемые в программе)</b>		
	Отсутствуют	

Переменные  $m, n, a$  используются в качестве исходных данных.

Переменные  $i, j, k, d, b$  в процессе выполнения программы создаются и модифицируются.

Переменные  $g, p$  и клавиша используются для управления выполнением программы.

Таким образом, исходя из результатов анализа исходного текста программы, получаем следующие значения характеристик:

$P = 3$  – количество переменных для расчетов;

$M = 5$  – количество модифицируемых переменных;

$C = 3$  – количество переменных, используемых в управлении программой;

$T = 0$  – количество неиспользуемых переменных (такие переменные в программе отсутствуют).

Расчет метрики Чепина:

$$Q = P + 2M + 3C + 0,5T = 3 + 2 \cdot 5 + 3 \cdot 3 + 0,5 \cdot 0 = 22$$

На основе полученных значений метрики Чепина уровень сложности данного решения можно считать сравнительно низким, так как в исходном тексте программы используется незначительное количество переменных, что не затрудняет понимание программы.

## 2.2. Задача «Сортировка строк матрицы»

Дана вещественная матрица размером  $N \times M$  элементов. Размер матрицы вводится с клавиатуры ( $M$  не больше 10).

Необходимо отсортировать строки матрицы по возрастанию их поэлементных сумм. Матрица заполняется случайными числами в диапазоне от 1 до 5.

Разработать программу для решения задачи. На основе лексического анализа исходного текста программы определить значение метрики Чепина.

### 2.2.1 Реализация программы

Текст программы для реализации возможного алгоритма решения поставленной задачи представлен на языке программирования C# (рис. 2).

Номера строк	Строки программы
1	using System;
2	class MyArray
3	{
4	public static void sum(double[,] a, int n, int m, double[] s)
5	{
6	int i,j;
7	for(i=0; i<n; i++)
8	for(s[i]=0,j=0; j<m; j++)
9	s[i] += a[i,j];
10	}
11	public static void change(double[,] a, int m, int row1, int row2)
12	{
13	double buf;
14	int j; //Номер столбца
15	for(j=0; j<m; j++)
16	{
17	buf = a[row1,j];
18	a[row1,j] = a[row2,j];
19	a[row2,j] = buf;
20	}
21	}
22	public static void sort(double[,] a, int n, int m, double[] s)
23	{
24	int i;
25	bool flag;
26	double buf;
27	do
28	{
29	n--;
30	for(flag=false,i=0; i<n; i++)
31	if(s[i] > s[i+1])
32	{
33	change(a,m,i,i+1);
34	buf = s[i];
35	s[i] = s[i+1];
36	s[i+1] = buf;
37	flag = true;
38	}



39	} while(flag);
40	}
41	}
42	class MyMetod
43	{
44	public static void Main()
45	{
46	double[,] a;
47	double[] s;
48	double min = 1.0,max = 5.0;
49	int n,m;
50	char rep;
51	string sinp;
52	do
53	{
54	Console.Write("Строк: ");
55	sinp = Console.ReadLine();
56	n = int.Parse(sinp);
57	Console.Write("Столбцов: ");
58	sinp = Console.ReadLine();
59	m = int.Parse(sinp);
60	a = new double[n,m];
61	s = new double[n];
62	IOArray.rand(a,n,m,min,max);
63	IOArray.print(a,n,m,"{0,8:f2}");
64	Console.WriteLine();
65	MyArray.sum(a,n,m,s);
66	MyArray.sort(a,n,m,s);
67	IOArray.print(a,n,m,"{0,8:f2}");
68	Console.Write("\nДля повтора нажмите клавишу Y: ");
69	rep = char.Parse(Console.ReadLine());
70	Console.WriteLine();
71	}while(rep == 'Y'    rep == 'y');
72	}
73	}
74	class IOArray
75	{
76	public static void print(double[,] a, int n, int m, string fmt)
77	{
78	int i,j;
79	for(i=0; i<n; i++, Console.WriteLine())
80	for(j=0; j<m; j++)
81	Console.Write(fmt,a[i,j]);
82	}
83	
84	public static void rand(double[,] a, int n, int m, double min, double max)
85	{
86	Random ran = new Random();
87	int i,j;
88	for(i=0; i<n; i++)
89	for(j=0; j<m; j++)
90	a[i,j] = min + (max-min)*ran.NextDouble();
91	}
92	}

Рис. 2. Текст программы для сортировки строк матрицы

### 2.2.2 Оценка характеристик программы

Рассмотрим текст программы для оценки ее качества с помощью метрики Чепина, которая позволяют оценить меру трудности понимания программы на основе входных и выходных данных.

На основе анализа исходного текста составим табл. 2.

Таблица 2.

#### Результат анализа объявленных переменных

№ п/п	Наименования переменных	Номера строк
<b><i>P</i> (для расчетов и обеспечения вывода)</b>		
1	<i>sinp</i>	51
2	<i>n</i>	56
3	<i>m</i>	59
4	<i>min</i>	48
5	<i>max</i>	48
<b><i>M</i> (модифицируемые или создаваемые)</b>		
1	<i>a</i>	46
2	<i>s</i>	47
3	<i>i</i>	6
4	<i>j</i>	6
5	<i>buf</i>	13
6	<i>row1</i>	11
7	<i>row2</i>	11
<b><i>C</i> (управляющие переменные)</b>		
1	<i>ran</i>	75
2	<i>flag</i>	25
3	<i>rep</i>	50
4	<i>fmt</i>	65
<b><i>T</i> (не используемые в программе)</b>		
	Отсутствуют	

Исходя из полученных на основе анализа данных имеем:

$P = 5$  – количество переменных для расчетов;

$M = 1$  – количество модифицируемых переменных;

$C = 2$  – количество переменных, используемых в управлении программой;

$T = 0$  – количество неиспользуемых переменных.

Расчет метрики Чепина:

$$Q = P + 2M + 3C + 0,5T = 5 + 2 \cdot 1 + 3 \cdot 2 + 0,5 \cdot 0 = 31$$

Рассматриваемая задача имеет многомодульное построение решения, что повышает уровень ее сложности. Этот фактор оказывает влияние и на уровень сложности решения.

Использование многомодульного формирования алгоритма решения в значительной степени усложняет понимание программы из-за дополнительного ввода переменных для расчета, модифицируемых переменных и переменных управления вычислительным процессом.

### 2.3. Задача «Формирование вещественной матрицы»

Определить класс с методами заполнения и вывода вещественных массивов. Используя эти методы, сформировать вещественную матрицу размером  $N \times M$  элементов. Размер матрицы вводится с клавиатуры ( $M$  не больше 10). Отсортировать каждую строку матрицы по возрастанию элементов строки. Матрица заполняется случайными числами в диапазоне от 1 до 5. Разработать программу для решения задачи. На основе лексического анализа исходного текста программы определить значение метрики Чепина.

#### 2.3.1 Реализация программы

Текст программы для реализации возможного алгоритма решения поставленной задачи представлен на языке программирования C# (рис. 3).

Номера строк	Строки программы
1	<code>using System;</code>
2	<code>class MyMetod</code>
3	<code>{</code>

```

4 public static void Main()
5 {
6     double[][] a;
7     double min = 1.0, max = 5.0;
8     int n=0, m=0, i;
9     char rep;
10    string sinp;
11    do
12    {
13        try
14        {
15            Console.Write("Строк: ");
16            sinp = Console.ReadLine();
17            n = int.Parse(sinp);
18            Console.Write("Столбцов: ");
19            sinp = Console.ReadLine();
20            m = int.Parse(sinp);
21            a = new double[n][];
22            for(i=0; i<n; i++)
23                a[i] = new double[m]
24                IOArray.rand(a,min,max);
25                IOArray.print(a,"{0,8:f2}");
26            Console.WriteLine();
27            for(i=0; i<=n; i++)
28                Array.Sort(a[i]);
29                IOArray.print(a,"{0,8:f2}");
30        }
31        catch (IndexOutOfRangeException)
32        { Console.WriteLine("Переполнение массива!!!"); }
33        Console.Write("\nДля повтора нажмите клавишу Y: ");
34        rep = char.Parse(Console.ReadLine());
35        Console.WriteLine();
36        } while(rep == 'Y' || rep == 'y');
37    }
38
39    class IOArray
40    {
41    public static void rand(int[][] a, int min, int max)
42    {
43        Random ran = new Random();
44        int i,j;
45        for(i=0; i<a.Length; i++)
46            for(j=0; j<a[i].Length; j++)
47                a[i][j] = ran.Next(min,max+1);
48    }
49    public static void print(double[][] a, string fmt)
50    {
51        int i,j;
52        for(i=0; i<a.Length; i++, Console.WriteLine())
53            for(j=0; j<a[i].Length; j++)
54                Console.Write(fmt,a[i][j]);
55    }
56    }

```

*Рис. 3. Текст программы для формирования вещественной матрицы*

### 2.3.2 Оценка характеристик программы

Рассмотрим текст программы для оценки ее качества с помощью метрики Чепина, которая позволяет оценить меру трудности понимания программы на основе входных и выходных данных. Определим основные характеристики для расчета метрики Чепина.

На основе анализа исходного текста составим табл. 3.

Таблица 3.

Результат анализа объявленных переменных		
№ п/п	Наименования переменных	Номера строк
<i>P</i> (для расчетов и обеспечения вывода)		
1	<i>sinp</i>	10
2	<i>n</i>	8
3	<i>m</i>	8
4	<i>min</i>	7
5	<i>max</i>	7
<i>M</i> (модифицируемые или создаваемые)		
1	<i>a</i>	6
2	<i>i</i>	8
3	<i>j</i>	6
<i>C</i> (управляющие переменные)		
1	<i>ran</i>	75
2	<i>rep</i>	50
3	<i>fmt</i>	65
<i>T</i> (не используемые в программе)		
	Отсутствуют	

Исходя из полученных на основе анализа данных имеем:

$P = 5$  – количество переменных для расчетов;

$M = 3$  – количество модифицируемых переменных;

$C = 3$  - количество переменных, используемых в управлении программой;

$T = 0$  - количество неиспользуемых переменных.

Расчет метрики Чепина:

$$Q = P + 2M + 3C + 0,5T = 5 + 2 \cdot 3 + 3 \cdot 3 + 0,5 \cdot 0 = 20.$$

Уровень сложности данной задачи ниже уровня предыдущей задачи. Снижение уровня сложности можно объяснить тем, что количество программных модулей в решении для этой задачи меньше. Снижение уровня модульности программы влечет за собой уменьшение количества всех видов переменных, что в значительной степени снижает ее сложность.

## 2.4. Задача «Заправка топливных баков»

Определить класс «Бак», описывающий понятие «Топливный бак». При этом должны быть использованы следующие поля:

- ширина, длина и высота в сантиметрах;
- вид топлива.

Следует учитывать операции (методы):

- полная заправка бака заданным видом топлива;
- вычисление стоимости полной заправки бака.

Бак в общем случае имеет вид параллелепипеда. Стандартным считается бак, имеющий вид куба (ширина, длина и высота совпадают).

Возможные разновидности баков:

- бак с одинаковой шириной и длиной, но с индивидуальной высотой;
- бак с индивидуальными размерами по ширине, длине и высоте.

Стоимость одного кубического сантиметра топлива определяется полями класса «Топливо». Стоимость топлива в рамках решения задачи неизменна. Выдача стоимости топлива реализуется специальной операцией этого класса.

Заполнить и вывести на экран стоимость заправки четырех баков:

- 10 x 20 x 30 см: топливо - газ;
- 10 x 10 x 20 см: топливо - керосин;
- 10 x 10 x 10 см: топливо - бензин;
- 20 x 20 x 20 см: топливо - бензин.

### 2.4.1 Реализация программы

Текст программы для реализации возможного алгоритма решения поставленной задачи представлен на языке программирования C# (рис. 4).

Номера строк	Строки программы
1	using System;
2	namespace EX2_1
3	{
4	class Топливо
5	{
6	private static double ценаГаз = 0.05;
7	private static double ценаКеросин = 0.08;
8	private static double ценаБензин = 0.1;
9	public static double Цена(string вид)
10	{
11	switch (вид)
12	{
13	case "газ": return ценаГаз;
14	case "керосин": return ценаКеросин;
15	case "бензин": return ценаБензин;
16	default: return 0.0;
17	}
18	}
19	}
20	class Бак
21	{
22	private double x, y, h;
23	private string видТоплива;
24	public void Заполнить(double xb, string вид)
25	{
26	x = xb; y = xb; h = xb; видТоплива = вид;
27	}
28	public void Заполнить(ref double xb, string вид)
29	{
30	x = xb; y = xb; h = xb; видТоплива = вид;
31	}
32	public void Заполнить(double xb, double yb, string вид)
33	{
34	x = xb; y = yb; h = xb; видТоплива = вид;
35	}
36	public void Заполнить(double xb, double yb, double hb, string вид)

```

37     {
38         x = xb; y = yb; h = hb; видТоплива = вид;
39     }
40     public double Оплата()
41     {
42         return x * y * h * Топливо.Цена(видТоплива);
43     }
44 }
45 class Program
46 {
47     static void Main(string[] args)
48     {
49         double x = 20.0;
50         Бак b;
51         b = new Бак();
52         b.Заполнить(10.0,20.0,30.0, "газ");
53         Console.WriteLine("Стоимость заправки: " + b.Оплата());
54         b.Заполнить(10.0, 20.0,"керосин");
55         Console.WriteLine("Стоимость заправки: " + b.Оплата());
56         b.Заполнить(10.0,"бензин");
57         Console.WriteLine("Стоимость заправки: " + b.Оплата());
58         b.Заполнить(ref x, "бензин");
59         Console.WriteLine("Стоимость заправки: " + b.Оплата());
60         Console.ReadKey();
61     }
62 }
63

```

*Рис. 4. Текст программы для задачи «Заполнение топливных баков»*

#### 2.4.2 Оценка характеристик программы

Рассмотрим текст программы для оценки ее качества с помощью метрики Чепина, которая позволяет оценить меру трудности понимания программы на основе входных и выходных данных. Определим основные характеристики для расчета метрики Чепина.

Необходимо отметить, что в исходном тексте программы используются одноименные программные модули и имена переменных входных параметров этих модулей. Несмотря на совпадение имен, эти переменные являются различными элементами программы, так как принадлежат различным программным модулям.

На основе анализа исходного текста составим табл. 4. Для одноименных переменных введем дополнительный столбец в табл. 4 с целью облегчения подсчета метрики Чепина.



## Результат анализа объявленных переменных

№ п/п	Наименования переменных	Номера строк	Количество переменных
Р (для расчетов и для обеспечения вывода)			
1	$xb$	24, 28, 32, 36	4
2	$yb$	32, 36,	2
3	$hb$	36	1
М (модифицируемые или создаваемые)			
1	ценаГаз	6	1
2	ценаКеросин	7	1
3	ценаБензин	8	1
4	$x$	22, 50	2
5	$y$	22	1
6	$h$	22	1
7	видТоплива	23	1
8	$b$	51	1
С (управляющие переменные)			
1	вид	9	1
Т (не используемые в программе)			
	Отсутствуют		

Исходя из полученных на основе анализа данных имеем:

$P = 1$  – количество переменных для расчетов;

$M = 9$  – количество модифицируемых переменных;

$C = 1$  – количество переменных, используемых в управлении программой;

$T = 0$  – количество неиспользуемых переменных.

Расчет метрики Чепина:

$$Q = P + 2M + 3C + 0,5T = 1 + 2 \cdot 9 + 3 \cdot 1 + 0,5 \cdot 0 = 28.$$

Уровень сложности задачи является достаточно высоким, так как реализация задачи имеет многомодульную схему, и, как следствие, используется большое количество переменных для расчета ( $P = 1$ ) и модифицируемых переменных ( $M = 9$ ).

### 3. ЗАДАЧИ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

В задачах, предлагаемых для самостоятельного решения, необходимо выполнить следующее:

- разработать программу, реализующую заданный алгоритм (рекомендуется использовать язык программирования C#);

- сформировать таблицы по образцу, приведенному в рассмотренных задачах;
- на основе лексического анализа исходного текста программы определить значение метрики Чепина;
- провести анализ полученных результатов, сформировав содержательные выводы.

Задача 1. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы главной диагонали на номера столбцов (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -10 до 10. Исходную и видоизмененную матрицы вывести на экран.

Задача 2. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы главной диагонали на значения элементов одномерного массива  $B$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массивов  $A$  и  $B$  осуществить при помощи датчика случайных чисел в диапазоне от -10 до 10. Исходную и видоизмененную матрицы, а также массив  $B$  вывести на экран.

Задача 3. В целочисленной матрицей  $A$  размером  $N \times M$  заменить элементы столбца, расположенного по центру, на значения элементов одномерного массива  $B$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массивов  $A$  и  $B$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы, а также массив  $B$  вывести на экран.

Задача 4. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы нечетных строк на значения элементов одномерного массива  $B$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массивов  $A$  и  $B$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы, а также массив  $B$  вывести на экран.

Задача 5. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы нечетных строк на значения элементов одномерного массива  $B$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массивов  $A$  и  $B$  осуществить при помощи датчика случайных чисел в диапазоне от -

20 до 20. Исходную и видоизмененную матрицы, а также массив В вывести на экран.

Задача 6. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы главной диагонали и расположенные выше нее на значение 2 (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы вывести на экран.

Задача 7. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы главной диагонали и расположенные ниже нее на значение 3 (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы вывести на экран.

Задача 8. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы побочной диагонали и расположенные выше нее на значение 4 (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы вывести на экран.

Задача 9. В целочисленной матрице  $A$  размером  $N \times M$  заменить элементы побочной диагонали и расположенные ниже нее на значение 4 (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Первоначальное заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 15. Исходную и видоизмененную матрицы вывести на экран.

Задача 10. В целочисленной матрице  $A$  размером  $N \times M$  подсчитать сумму элементов главной диагонали и расположенных выше нее (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -5 до 5. Исходную матрицу и сумму элементов заданной области матрицы вывести на экран.

Задача 11. В целочисленной матрице  $A$  размером  $N \times M$  подсчитать сумму элементов главной диагонали и расположенных ниже нее (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от -3 до 3. Исходную матрицу и сумму элементов заданной области матрицы вывести на экран.

Задача 12. В целочисленной матрице  $A$  размером  $N \times M$  подсчитать сумму элементов побочной диагонали и расположенных выше нее (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от 1 до 10. Исходную матрицу и сумму элементов заданной области матрицы вывести на экран.

Задача 13. В целочисленной матрице  $A$  размером  $N \times M$  подсчитать сумму элементов побочной диагонали и расположенных ниже нее (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Заполнение массива  $A$  осуществить при помощи датчика случайных чисел в диапазоне от 2 до 12. Исходную матрицу и сумму элементов заданной области матрицы вывести на экран.

Задача 14. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов строк вывести на экран с указанием номера строки. Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -2,2 до 10,2, матрицу вывести на экран.

Задача 15. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов столбцов вывести на экран с указанием номера столбца. Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -5,2 до 5,2, матрицу вывести на экран.

Задача 16. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов строк занести в одномерный массив  $B$ . Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -3,3 до 3,3. Массивы  $A$  и  $B$  вывести на экран.

Задача 17. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов столбцов занести в одномерный массив  $B$ . Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -5,5 до 5,5. Массивы  $A$  и  $B$  вывести на экран.

Задача 18. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов нечетных строк занести в одномерный массив  $B$ . Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -10,1 до 3,1. Массивы  $A$  и  $B$  вывести на экран.

Задача 19. Сформировать вещественную матрицу  $A$  размером  $N \times M$  (числа  $N$  и  $M$  задаются с клавиатуры в диапазоне от 3 до 10). Суммы элементов нечетных столбцов занести в одномерный массив  $B$ . Первоначальное заполнение матрицы осуществить при помощи датчика случайных чисел в диапазоне от -2,5 до 2,5. Массивы  $A$  и  $B$  вывести на экран.

Задача 20. Создать двухмерный массив размером  $N \times M$  и заполнить случайным образом нулями и единицами. Проверить, есть ли строгое чередование нулей и единиц в каждой строке массива. Исходный массив и номера строк, для которых выполняется условие чередования, вывести на экран. Значения  $N$  и  $M$  вводятся с клавиатуры.

Задача 21. Создать двухмерный массив размером  $N \times M$  и заполнить случайным образом нулями и единицами. Проверить, есть ли строгое чередование нулей и единиц в каждом столбце массива. Исходный массив и номера столбцов, для которых выполняется условие чередования, вывести на экран. Значения  $N$  и  $M$  вводятся с клавиатуры.

Задача 22. Сформировать целочисленный двухмерный массив размером  $N$  строк. Строки имеют разное количество элементов. Правило формирования длины строки и значений элементов строки показано ниже. Полученный массив построчно вывести на экран. Значение  $N$  вводится с клавиатуры.

1	2					
1	2	3	4			
...	...	...	...	...	...	
1	2	3	4	...	$2 \cdot N - 1$	$2 \cdot N$

Задача 23. Сформировать целочисленный двухмерный массив размером  $N$  строк. Строки имеют разное количество элементов. Правило формирования длины строки и значений элементов строки показано ниже. Полученный массив построчно вывести на экран. Значение  $N$  вводится с клавиатуры.

1	2	3	4	5	6	...	$2 \cdot N - 1$	$2 \cdot N$
1	2	3	4	...	$2 \cdot N - 3$	$2 \cdot N - 2$		
...	...	...	...	...				
1	2	3	4					
1	2							

Задача 24. Сформировать целочисленный двумерный массив размером  $N$  строк. Строки имеют разное количество элементов. Правило формирования длины строки и значений элементов строки показано на рисунке. Полученный массив построчно вывести на экран. Значение  $N$  вводится с клавиатуры.

1				
2	1			
3	2	1		
...	...	...	1	
$N$	$N - 1$	$N - 2$	...	1

Задача 25. Сформировать целочисленный двумерный массив размером  $N$  строк. Строки имеют разное количество элементов. Правило формирования длины строки и значений элементов строки показано на рисунке. Полученный массив построчно вывести на экран. Значение  $N$  вводится с клавиатуры.

$N$	$N - 1$	$N - 2$	...	1
$N - 1$	$N - 2$	...	1	
...	...	...		
2	1			
1				

## СПИСОК РЕКОМЕНДУЕМО ЛИТЕРАТУРЫ

1. Черников Б.В. Управление качеством программного обеспечения:/ Б.В. Черников. – М.: ИД «ФОРУМ»: ИНФРА-М, 2012. – 240 с.:
2. Черников Б.В. Оценка качества программного обеспечения: Практикум: учебное пособие / Б.В. Черненко. Б.Е. Поклонов/ Под ред. Б.В. Чернякова. – М.: ИД «Форум»: ИНФРА-М, 2012. – 400 с.

Метрология и качество программного обеспечения. Оценка характеристик программ на основе лексического анализа. Метрики Чепина: методические указания к выполнению лабораторной работы №3 для студентов очной формы обучения по направлению подготовки 09.03.01 «Информатика и вычислительная техника»; 09.03.04 «Программная инженерия»; 02.03.03 «Математическое обеспечение и администрирование информационных систем»

АЗАРЧЕНКОВ АНДРЕЙ АНАТОЛЬЕВИЧ

Научный редактор Д.А. Коростелев

Компьютерный набор А.А. Азарченков

Иллюстрации А.А. Азарченков

---

Подписано в печать 07.07.2017г. Формат 60х84 1/16 Бумага офсетная. Офсетная печать. Усл.печ.л. 1,4 Уч.-изд.л. 1,4 Тираж 1 экз.

---

Брянский государственный технический университет

Кафедра «Информатика и программное обеспечение», тел. 56-09-84

241035, Брянск, бульвар 50 лет Октября, 7 БГТУ