Лабораторная работа №6
Обработка двумерных массивов и контейнеров

Цели и задачи работы: изучение алгоритмов формирования и обработки двумерных массивов, программирование и отладка программ формирования и обработки массивов (статических и динамических) и контейнеров STL.

Задание к работе: Самостоятельно решить задачи в соответствии с индивидуальным вариантом.

Методика выполнения работы:

1. Разработать алгоритм решения задачи по индивидуальному заданию.

2. Написать и отладить программу решения задачи на двух языках (C++ и второй, по выбору).

3. Протестировать работу программы на различных исходных данных.

Перечень понятий к защите лабораторной работы 6.

Массив. Статический массив: определение, использование. Виды перебора элементов массива. Обработка элементов массива по два элемента.

Многомерные массивы. Хранение, описание, доступ к элементам. Пример инициализации. Передача параметров одномерного массива (виды). Передача параметров многомерного массива (виды).

Понятие указатели и динамическая память. Схема работы стек-куча при указателях. Оператор new. Инициализация объектов в куче. Контейнеры (динамика). Доступ к элементу в куче. Указатель на функцию. Указатель на объект. Указатель на void. Инициализация указателей (4 вида). Освобождение памяти. Операции с указателями. Арифметические операции с указателями. Ссылки. Взаимосвязь ссылки и указателя. Указатели и массивы. Динамические одномерные массивы. Динамические двумерные массивы: порядок создания и освобождения памяти. Контейнер vector. назначение, инициализация, принцип работы.

Задание №6.1 Обработка двумерных массивов.

Вариант 1

1. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [0, 50]. Определите, какое простое число встречается в матрице наибольшее количество раз.
2. Напишите программу, инициализирующую матрицу, как в примере. Размерность матрицы задаётся с клавиатуры. Модифицируйте вывод таким образом, чтобы вместо “0” в консоль выводилось “000”.

N = 5	*N = 7*
	000 105 104 103 102 101 100
000 103 102 101 100	100 000 104 103 102 101 100
100 000 102 101 100	100 101 000 103 102 101 100
100 101 000 101 100	100 101 102 000 102 101 100
100 101 102 000 100	100 101 102 103 000 101 100
100 101 102 103 000	100 101 102 103 104 000 100
	100 101 102 103 104 105 000

3. В двумерном массиве размерностью $M \times N$, все элементы которого различны, требуется найти такие элементы, которые одновременно являются минимальными в своей строке и максимальными в своем столбце.
4. Дана матрица, заполненная нулями и единицами. Найдите наибольший прямоугольник, состоящий только из единиц, а также его площадь. Пример:

Ввод	Вывод
1 0 0 0 0 1 0 1 1 1 1 0 1 1 1 1 0 0 0 0	6

Вариант 2

1. Определите и инициализируйте матрицу размера *M × N* случайными целыми числами в диапазоне [10, 50]. Поменяйте в каждой строке местами максимальный и минимальный элементы. Определите и инициализируйте новую матрицу, первая строка которой содержит минимальные элементы, а вторая строка – максимальные.
2. Определите и инициализируйте матрицу размера *M × N* случайными целыми числами в диапазоне [100, 200]. Найдите строку, в которой сумма цифр элементов наибольшая.
3. Даны числа *M* и *N*. Необходимо создать матрицу размером *M × N*, заполнить её случайными числами в диапазоне [-10, 10] и вывести на экран. После чего необходимо в каждом столбце найти минимальный элемент и переупорядочить столбцы матрицы по возрастанию этой величины. Пример:

Входные данные:	Выходные данные:
3 4	-4 6 0 7 8 -9 -3 1 0 2 5 -5 6 7 -4 0 -9 1 8 -3 2 -5 0 5

4. Реализуйте клеточный автомат Джона Конвея на **ограниченной** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на нескольких примерах устойчивых фигур (фигур, которые остаются неизменными) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 3

1. Определите и инициализируйте квадратную матрицу размерности N *(где* $N > 6$*)* случайными целыми числами. Определите и инициализируйте новую матрицу, содержащую две строки: копии элементов диагоналей исходной матрицы.
2. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами. Замените в элементах диагоналей матрицы значения на разность наибольшего и второго наименьшего элементов матрицы.
3. Создайте матрицу размером $M \times N$ и заполните ее, как показано на примере. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

0	1	3	6	10	14	18	22	26	30
2	4	7	11	15	19	23	27	31	34
5	8	12	16	20	24	28	32	35	37
9	13	17	21	25	29	33	36	38	39

4. Определите, по правилами ли заполнена доска для судоку размером 9 х 9. Только заполненные ячейки должны быть проверены в соответствии со следующими правилами: каждая строка должна содержать цифры 1-9 без повторения, каждый столбец должен содержать цифры 1-9 без повторения, каждый из девяти вложенных блоков сетки размером 3 х 3 должен содержать цифры 1-9 без повторения. Доска судоку (частично заполненная) может быть валидной, но не обязательно разрешимой. Только заполненные ячейки должны быть проверены в соответствии с указанными правилами.
   Для упрощения достаточно реализовать для четырех квадратов, то есть 6 х 6.

Ввод

7		5						
	2		4			5		
6	4	3	5					1
5		4	7		6			
1	3	6	8		2	4	5	7
			4		4	9	1	6
2					3	1	7	5
		9			1		6	
						3		9

Вывод

true

Вариант 4

1. Определите и инициализируйте матрицу размерности $M \times N$ *(где $M > 7$, $N > 5$)* случайными вещественными числами. Определите и инициализируйте массив произведениями элементов столбцов матрицы.
2. Напишите программу, инициализирующую матрицу, как в примере. Количество строк матрицы задаётся с клавиатуры. Пример:

$N = 3$	$N = 5$
	10
10	12 14
12 14	16 18 20
16 18 20	22 24 26 28
	30 32 34 36 38

3. Магическим квадратом порядка N называется квадратная матрица размера $N \times N$, составленная из чисел $1, 2, \ldots, N^2$ так, что суммы по каждому столбцу, каждой строке и каждой из двух больших диагоналей равны между собой. Напишите программу, которая строит магический квадрат заданного порядка N.
4. Реализуйте клеточный автомат Джона Конвея на **замкнутой** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на нескольких примерах периодических фигур (фигур, у которых состояние повторяется через некоторое число поколений) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 5

1. Определите и инициализируйте матрицу размерности $M \times N$ (где M>7, N>5) случайными целыми числами в диапазоне (-10, 10). Найдите номер строки, содержащей наибольшее количество отрицательных чисел. Определите и инициализируйте новый массив найденной строкой.
2. Определите и инициализируйте квадратную матрицу порядка M $(M > 5)$ случайными целыми числами в диапазоне [10, 100]. Отсортируйте цифры в элементах обеих диагоналей в порядке убывания. Определите, сумма элементов какой из диагоналей наибольшая.
3. Вводится число N. Необходимо создать матрицу размером $N \times N$ и заполнить её случайными числами в диапазоне [1, 5]. После чего необходимо проверить, является ли матрица симметричной относительно главной и побочной диагоналей. (Для каждой диагонали написать в отдельной строке YES или NO).

Входные данные:	Выходные данные:
N = 3 1 0 2 0 3 3 2 3 1	YES NO

4. Дана доска для игры «морской бой» размера m x n, где каждая ячейка представляет собой линкор "X" или пустое ".". Определите количество кораблей на доске. Корабли могут быть размещены только горизонтально или вертикально на доске. По крайней мере, ячейка разделяет два линкора (т.е. корабли не могут примыкать друг к другу).

<table>
<tr><th>Ввод</th><th>Вывод</th></tr>
<tr><td><table>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>X</td><td></td><td>X</td></tr>
<tr><td></td><td></td><td></td><td>X</td></tr>
<tr><td></td><td></td><td></td><td>X</td></tr>
</table></td><td>2</td></tr>
</table>

Вариант 6

1. Определите и инициализируйте матрицу размерности $M \times N$ случайными вещественными числами в диапазоне [0, 100]. Найдите и выведите средние арифметические значения элементов матрицы, а также каждой строки. Определите номер строки, у которой среднее арифметическое значение наибольшее.
2. Определите и инициализируйте квадратную матрицу порядка M $(M > 5)$ случайными целыми числами в диапазоне [-100, 100]. Отсортируйте чётные столбцы в порядке возрастания, а нечётные – в порядке убывания. Определите, в какой половине матрицы относительно главной диагонали, включая саму диагональ, больше положительных элементов.
3. Создайте матрицу размером $M \times N$ и заполните ее, как показано на примере. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

100	110	120	130	140
200	210	220	230	240
300	300	300	300	300
290	280	270	260	250
190	180	170	160	150

4. Реализуйте клеточный автомат Джона Конвея на **ограниченной** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на нескольких примерах двигающихся фигур (фигур, у которых состояние повторяется, но со смещением) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 7

1. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [-50, 50]. Найдите в строках матрицы самую длинную возрастающую последовательность элементов. Инициализируйте новый массив данной последовательностью.
2. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [100, 200]. Найдите и выведите среднее арифметическое чётных столбцов и среднее арифметическое нечётных строк. Если больше арифметическое чётных столбцов, то отсортируйте главную диагональ в порядке убывания, если меньше – в порядке возрастания.
3. Задаётся число строк двумерного массива N. После чего следующие N чисел задают размеры каждой строки. Необходимо создать массив с данными размерами, отсортировать его по возрастанию размера строки и заполнить неповторяющимися числами по убыванию так, чтобы последнее число было равно 1.

Входные данные:	Выходные данные:
4 3 5 2 2	12 11 10 9 8 7 6 5 4 3 2 1

4. Дана матрица символов и строка, содержащая слово. Напишите программу, определяющую, содержится ли заданное слово в матрице или не содержится. Слово может быть составлено из букв последовательно смежных ячеек, где соседние ячейки являются соседними по горизонтали или вертикали. Одна и та же буквенная ячейка не может использоваться более одного раза.

Ввод “JSPROGS”,	Вывод
Q W E R H J S P E S S F R S F A D O G S S S G F I	true

Вариант 8

1. Определите и инициализируйте матрицу размера $M \times N$ случайными целыми числами в диапазоне [-10, 40]. Выведите номера строк, содержащих хотя бы три отрицательных элемента. Инициализируйте новый массив этими отрицательными элементами. Создайте новую матрицу, состоящую из найденных строк.
2. Определите и инициализируйте квадратную матрицу порядка N (где N чётное число) случайными целыми числами в диапазоне [10, 30]. Замените значения в левой нижней четверти матрицы на 0, в правой нижней части матрицы на 10, а в оставшихся четвертях поменяйте местами
3. Дано число n. Задаётся верхняя треугольная матрица размером $n \times (n + 1)$ (все элементы ниже главной диагонали равны 0). Необходимо решить СЛАУ, которая задана с помощью этой матрицы, и вывести значения всех x.
4. Реализуйте клеточный автомат Джона Конвея на **замкнутой** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на примере «ружья» (фигуры, у которой состояние повторяется, но дополнительно появляется двигающаяся фигура) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 9

1. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [0, 100]. Найдите в строках матрицы возрастающую последовательность элементов с минимальным начальным элементом. Инициализируйте новый массив найденной последовательностью.
2. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [-50, 50]. Определите, сколько раз каждый уникальный элемент встречается в матрице.
3. По введенным значениям N, M $(1 \le N \le 20,\ 1 \le M \le 20)$ заполните массив размерностью $N \times M$ числами от 1 до $N \times M$, расположив их по спирали, закрученной по часовой стрелке, так, как показано в примере. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

7	8	9	10
6	15	16	11
5	14	13	12
4	3	2	1

4. На шахматной доске расположены несколько чёрных ферзей и один белый король. Матрица координат ферзей и пара координат короля задают их положение на шахматной доске. Определите координаты ферзей, которые атакуют короля. Программа должна автоматически заполнять поле фигурами случайным образом. Реализуйте возможность добавления на поле коней и определение координат тех, которые атакуют короля.

<table>
<tr><th>Ввод</th><th>Вывод</th></tr>
<tr><td>queens = [[0,1],[1,0],[4,0],[0,4],[3,3],[2,4]]
king = [0,0]
<table>
<tr><td>K</td><td>Q</td><td></td><td></td><td>Q</td><td></td><td></td><td></td></tr>
<tr><td>Q</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>Q</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>Q</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Q</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td><td>[[0,1],[1,0],[3,3]]</td></tr>
</table>

Вариант 10

1. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [-50, 50]. Вычислите и выведите сумму всех неотрицательных элементов в чётных строках, стоящих на нечётных местах. Инициализируйте новый массив этими числами.
2. Определите и инициализируйте квадратную матрицу порядка N случайными целыми числами в диапазоне [100, 200]. Найдите сумму элементов строк матрицы. Определите и инициализируйте массив строкой со второй максимальной суммой элементов среди строк матрицы.
3. Создайте матрицу размером $M \times N$ и заполните ее, как показано на примере. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

34	32	29	25	20	15	10
33	30	26	21	16	11	6
31	27	22	17	12	7	3
28	23	18	13	8	4	1
24	19	14	9	5	2	0

4. Реализуйте клеточный автомат Джона Конвея на **ограниченной** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на примере «паровоза» (двигающиеся фигуры, которая оставляет за собой следы в виде устойчивых или периодических фигур) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 11

1. Определите и инициализируйте квадратную матрицу порядка N случайными целыми числами в диапазоне [-100, 100]. Найдите строку, содержащую наибольшее количество положительных чисел. Инициализируйте новый массив найденной строкой.
2. Определите и инициализируйте матрицу размерности $M \times N$ случайными целыми числами в диапазоне [100, 200]. Определите число, которое встречается в матрице наибольшее количество раз. Если такого числа нет, то выведите соответствующее сообщение.
3. Даны два числа M и N. Создать матрицу размером $M \times N$ и заполнить случайными числами в диапазоне [-10,10]. Необходимо найти в каждой строке максимальный по модулю элемент и отсортировать строки матрицы по возрастанию этого значения.

Входные данные:	Выходные данные:
3 3 9 9 3 3 3 -8 -4 0 7	-4 0 7 3 3 -8 9 9 3

4. Реализуйте клеточный автомат Джона Конвея на **замкнутой** плоскости по классическим и оригинальным (своим) правилам. Продемонстрируйте работу клеточного автомата на нескольких примерах двигающихся фигур (фигур, у которых состояние повторяется, но со смещением) и на развитии колоний клеток, сгенерированных в случайном порядке. Понаблюдайте за изменениями в эволюции. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 12

1. Определите и инициализируйте квадратную матрицу порядка *N (N - четное число)* случайными вещественными числами в диапазоне [0, 100]. Определите и инициализируйте новую матрицу, состоящую из левой верхней четверти исходной матрицы.
2. Определите и инициализируйте матрицу размера $M \times N$ случайными целыми числами в диапазоне [1000, 5000]. Найдите строку, в которой сумма цифр элементов наименьшая.
3. Вводится число *N*. Необходимо создать матрицу размером $N \times N$ и заполнить её случайными числами в диапазоне [1, 5]. После чего необходимо проверить, является ли матрица симметричной относительно главной и побочной диагоналей. (Для каждой диагонали написать в отдельной строке *YES* или *NO*).

Входные данные:	Выходные данные:
N = 3 1 0 2 0 3 3 2 3 1	YES NO

4. Реализуйте клеточный автомат Джона Конвея на **ограниченной** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на примере пожирателя (устойчивой фигуры, которая может пережить столкновения с некоторыми двигающимися фигурами) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 13

1. Определите и инициализируйте квадратную матрицу порядка *N (N – чётное число)* случайными целыми числами в диапазоне [-50, 50]. Модифицируйте матрицу таким образом, чтобы в чётных столбцах не было чисел из диапазона [-10, 10].
2. Определите и инициализируйте матрицу размера *M × N* случайными целыми числами в диапазоне [-100, 100]. Определите, каких чисел в столбце больше, если положительных, то отсортируйте первую половину столбца в порядке возрастания, если меньше, то возведите в квадрат все элементы в столбце, а если равно, то замените все числа на ноль.
3. Определите и инициализируйте матрицу размера *M × N*. Заполните её таким образом, как продемонстрирован в примере. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

100	110	120	130	140	150	160
110	100	110	120	130	140	150
120	110	100	110	120	130	140
130	120	110	100	110	120	130
140	130	120	110	100	110	120
150	140	130	120	110	100	110
160	150	140	130	120	110	100

4. Дана матрица размера *M × N* из единиц и нулей. Определите, сколько квадратных подматриц составляют единицы, а также сколько и какого размера.

Ввод	Вывод
0 1 1 1 1 1 1 1 0 1 1 1	Всего найдено 15 подматриц: 10 подматриц со стороной 1, 4 подматрицы со стороной 2, 1 подматрица со стороной 3.

Вариант 14

1. Определите и инициализируйте квадратную матрицу порядка *N (N – чётное число)* случайными целыми числами в диапазоне [100, 200]. Найдите и выведите сумму элементов сток, а затем столбцов матрицы.
2. Определите и инициализируйте матрицу размера $M \times N$ случайными целыми числами в диапазоне [1000, 3000]. Найдите строку, в которой содержится самая длинная последовательность чисел, в которых нет цифр 5 и 7. Инициализируйте новый массив найденной последовательностью.
3. Определите и инициализируйте матрицу размера $M \times N$ нулями. Задайте некоторое число k, а затем «нарисуйте» змейку длины k единицами в случайным образом. Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример выходных данных:

```
0 0 0 0 0 0 0 0 0 0    0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0    0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0    0 0 0 0 1 1 0 0 0 1
0 0 0 0 0 0 0 0 0 0    0 0 0 0 0 1 0 1 1 1
0 0 0 0 0 0 0 0 0 0    0 0 0 0 0 1 1 1 0 0
```

4. Реализуйте клеточный автомат Джона Конвея на **замкнутой** плоскости по классическим правилам. Продемонстрируйте работу клеточного автомата на примере «сорняка» («паразита») (фигуры, которая при столкновении с некоторыми фигурами дублируются) и на развитии колоний клеток, сгенерированных в случайном порядке. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени. Таким образом, получается непрерывная анимация.

Вариант 15

1. Определите и инициализируйте матрицу размера *M* × *N* случайными вещественными числами в диапазоне [-50, 50]. Перезапишите строки в матрице в обратном порядке (дополнительный массив не используйте). Найдите в матрице второй минимальный и максимальный элементы.
2. Напишите программу, инициализирующую матрицу, как в примере. Количество строк матрицы задаётся с клавиатуры. Пример:

N = 5	*N* = 9
	100
	105 110
100	115 120 125
105 110	130 135 140 145
115 120 125	150 155 160 165 170
105 110	130 135 140 145
100	115 120 125 115
	105 110
	100

3. Требуется заполнить массив размера *N* × *N* единицами по спирали (начиная с нижнего правого, как на рисунке). Продемонстрируйте выполнение кода на нескольких размерностях массива. Пример:

1	1	1	1	1	1	1
1	0	0	0	0	0	1
1	0	1	1	1	0	1
1	0	1	0	1	0	1
1	0	1	0	1	0	1
1	0	0	0	1	0	1
1	1	1	1	1	0	1

4. Реализуйте клеточный автомат Джона Конвея на **ограниченной** плоскости по классическим и оригинальным (своим) правилам. Продемонстрируйте работу клеточного автомата на примере устойчивой фигуры (фигуры, которые остаются неизменными), а затем на этой же фигуре с измененными правилами, и на развитии колоний клеток, сгенерированных в случайном порядке. Понаблюдайте за изменениями в эволюции. Реализовать работу графического отображения клеточного автомата можно таким образом, чтобы живые клетки изображались единицами, а мёртвые – нулями, либо живые – нулями, а мёртвые пробелами, либо другим возможным вариантом. Каждое новое поколение выводится на очищенное окно консоли через некоторый промежуток времени, например, 250 мс. Таким образом, получается непрерывная анимация.

Задание №6.2 Многоалфавитное шифрование методом сложной замены (подстановки). Дешифрование текста.

Вариант 1

а) систему шифрования Виженера для преобразования исходного текста на английском языке;

б) систему шифрования Виженера для преобразования зашифрованного текста в исходный.

Вариант 2

а) систему шифрования Виженера для преобразования исходного текста на русском языке;

б) систему шифрования Виженера для преобразования зашифрованного текста в исходный.

Для всех вариантов сгенерировать таблицы Виженера, пользуясь соответствующей кодировкой.

Теоретическая часть

Квадрат Виженера или таблица Виженера, так же известная как tabula recta, может быть использована для зашифрования и расшифрования. В XVI веке французский дипломат Блез де Виженер предложил модификацию шифра замен, которая впоследствии получила его имя. В данном шифре ключ задается фразой из d букв. Ключевая фраза подписывается с повторением под сообщением. Букву шифротекста необходимо находить на пересечении столбца, определяемого буквой открытого текста, и строки, определяемой буквой ключа:

$$Vig_d(m_i)=(m_i+k_{i \bmod d})(\mathrm{mod}\ n),$$

где m_i, k_i, $Vig_d(m_i)$- порядковые номера в алфавите очередных символов открытого текста, ключа и шифротекста соответственно. Обратное преобразование выглядит следующим образом:

$$Vig_d^{-1}(m_i)=(m_i-k_{i \bmod d}+n)(\mathrm{mod}\ n).$$

Шифр Виженера для латинского алфавита использует таблицу 1.

Шифр Виженера для русского алфавита использует таблицу 2:

Таблица 1 – Таблица Виженера для латинского алфавита

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	Z	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	X	A	B	S
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	S	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	V	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	S	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	S	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	G	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Таблица 2 – Таблица Виженера для русского алфавита

Ключ	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э
0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э
1	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю
2	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я
3	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а
4	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б
5	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в
6	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г
7	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д
8	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е
9	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж
10	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з
11	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и
12	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й
13	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к
14	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л
15	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м
16	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н
17	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о
18	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
19	у	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р
20	ф	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с
21	х	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т
22	ц	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у
23	ч	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф
24	ш	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х
25	щ	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц
26	ь	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч
27	ы	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш
28	ъ	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
29	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь
30	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы
31	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ы	ъ

Пример использования шифра Виженера приведен на рис. 1 (алфавит дополнен пробелом, порядковый номер которого принят за 34, соответственно, мощность алфавита n=34).

ключевая фраза – ‘ключ’
d=4

Порядковый № буквы в алфавите	26	10	22	18	34	3	10	8	6	15	6	18	1
Открытый текст	ш	и	ф	р		в	и	ж	е	н	е	р	а
Порядковый № буквы в алфавите	12	13	32	25	12	13	32	25	12	13	32	25	12
Ключевая фраза	к	л	ю	ч	к	л	ю	ч	к	л	ю	ч	к
Порядковый № буквы в алфавите	4	23	20	9	12	16	8	33	18	28	4	9	13
Шифротекст	г	х	т	з	к	о	ж	я	р	ь	г	з	л

Рисунок 1 - Пример шифрования с использованием шифра Виженера

Принципиальным отличием данного шифра от других является то, что он относится к классу многоалфавитных алгоритмов – как нетрудно заметить, одной и той же букве шифротекста могут соответствовать различные символы открытого текста в зависимости от того, каким символом ключа они были замаскированы (в приведенном примере буква ‘г’ шифрует в одном случае букву ‘ш’, а в другом – букву ‘е’). Это делает бессмысленным подсчет частоты встречаемости символов в шифротексте. Для криптоанализа шифра Виженера можно использовать *метод Казиски*.

В середине XIX века немецкий математик Казиски предложил определять длину парольной фразы по расстоянию между одинаковыми фрагментами шифротекста. Допустим, найдены два одинаковых фрагмента шифротекста, расстояние между которыми составляет 20 символов. Это может означать, что два одинаковых фрагмента открытого текста были зашифрованы с одной и той же позиции ключа. Это позволяет предположить, что парольная фраза имеет длину 4, 5, 10 или 20 символов. Узнав (или угадав) длину парольной фразы l, можно осуществить частотный криптоанализ шифротекста для выборки каждого l -го символа шифротекста.

Для увеличения длины парольной фразы и затруднения криптоанализа можно применить составной шифр Виженера, который представляет собой многократное шифрование по Виженеру с различными парольными фразами. Он имеет уравнение

$$Vig^*(m_i)=(m_i + k_{i \bmod dk} + l_{i \bmod dl} + \dots + s_{i \bmod ds}) \pmod n,$$

где k_i, l_i, ..., s_i – символы различных парольных фраз. Для повышения криптостойкости парольные фразы должны иметь различные взаимно простые периоды dk, dl, ..., ds. Это существенно затруднит для криптоаналитика поиск величины периода повторения общей парольной фразы, которая будет равна произведению длин всех парольных фраз.

Задание №6.3 Методы решения систем линейных алгебраических уравнений

Задание:

Из табл.6.3.1 выбрать данные для системы линейных уравнений. Найти решение этой системы прямым и приближенным методами с точностью до $\varepsilon=10^{-3}$. Варианты нечетные решить систему уравнений методом Гаусса с выбором главного элемента, четные варианты – методом Холесского.

Исходная система линейных уравнений:

$$\begin{cases} Mx_1 - 0.04x_2 + 0.21x_3 - 18x_4 = -1.24, \\ 0.25x_1 - 1.23x_2 + Nx_3 - 0.09x_4 = P, \\ -0.21x_1 + Nx_2 + 0.8x_3 - 0.13x_4 = 2.56, \\ 0.15x_1 - 1.31x_2 + 0.06x_3 + Px_4 = M. \end{cases}$$

Таблица 6.3.1 - Данные для исходной системы линейных уравнений по вариантам

№ варианта	M	N	P	№ варианта	M	N	P
1	-0.77	0.16	1.12	9	-1.13	0.14	0.87
2	0.93	0.07	-0.84	10	0.91	-0.23	-1.04
3	-1.14	-0.17	0.95	11	-0.88	0.1	0.91
4	1.08	0.22	-1.16	12	1.25	-1.14	-1.09
5	0.87	-0.19	1.08	13	0.79	0.18	-0.86
6	-1.21	0.2	0.88	14	-1.19	-0.21	1.21
7	1.09	-0.16	0.84	15	0.89	0.12	-1.15
8	0.89	0.08	-1.21	16	1.08	0.22	-1.16

Примерный порядок выполнения работы

1. **Составьте методику решения системы линейных уравнений прямым методом согласно варианту.**
2. **Составить программу решения системы уравнений, вывести результаты прямого и обратного хода метода.**
3. **Преобразовать систему к каноническому виду с выполнением условия сходимости итерационной последовательности.**
4. **Составьте методику решения системы уравнений любым итерационным методом.**
5. **Составьте программу решения системы уравнений с точностью до ε, выводящую результаты в таблицу:**

N	x_1	x_2	x_3	x_4	ε_n
…	…	…	…	…	…

6. **Найдите все корни системы уравнений и выпишите их с верными знаками.**

Теоретическая часть

1. Постановка задачи

Рассмотрим линейную неоднородную задачу для систем линейных алгебраических уравнений, которая записывается в виде:

$$Ax = b \tag{1}$$

где A – действительная матрица , b – вектор столбец. X – вектор неизвестных, принадлежат R^n – n-мерному евклидовому пространству.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned} \tag{2}$$

Требуется найти решение $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ из R^n системы (2), подстановка которого в (2) приводит к верному равенству $Ax^* = b$.

<u>Замечание!</u>

Из линейной алгебры известно, что решение задачи (2) существует и единственно, если детерминант матрицы A отличен от нуля.

Особенности задачи:

а) линейна и неоднородна;

б) количество неизвестных равно количеству уравнений;

в) количество n для некоторых практических задач велико;

г) трудно найти обратную матрицу при больших n.

Характер задачи и точность получаемого решения в большой степени зависят от ее обусловленности, являющейся важным математическим понятием, влияющим на выбор метода решения.

Более строго обусловленность задачи характеризуется числом обусловленности $\nu(A) = \|A\| \cdot \|A^{-1}\|$, где $\|A\|$ – норма матрицы A, $\|A^{-1}\|$ – норма обратной матрицы. Чем больше это число, тем хуже обусловленность системы. В качестве нормы A можно принять число, являющееся максимальным из сумм (по модулю) элементов строк этой матрицы.

!!! Реализация хорошей или плохой обусловленности в задачах напрямую связана с численной устойчивостью и неустойчивостью.

В численном анализе используются два класса численных методов решения систем линейных уравнений:

Прямые методы, позволяющие найти решение за определенное число операций. К прямым методам относятся: метод Гаусса и его модификации, метод LU – разложения и др.

Итерационные методы, основанные на использовании повторяющегося процесса и позволяющие получить решение в результате последовательных приближений. Операции, входящие в повторяющийся процесс, составляют итерацию. К итерационным методам относятся: метод простых итераций, метод Зейделя и др.

2. Прямые методы решения системы линейных уравнений

2.1 Метод Гаусса

Метод Гаусса состоит в исключении слагаемых системы путем ее равносильного преобразования. Метод разбивается на две совокупности операций, которые разбиваются условно на прямой и обратный ход.

а) Прямой ход состоит в исключении элементов, расположенных ниже элементов, соответствующих главной диагонали матрицы А. Матрица А преобразуется к верхнетреугольному виду с единицами на главной диагонали.

б) Обратный ход, из матрицы А* определяем последовательно $x_n,\dots,x_1$.

Надо решить систему алгебраических уравнений $Ax = b$:

$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1$

…

$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n$

Нужно преобразовать к треугольной матрице.

Сделаем последовательно:

<u>1 шаг:</u>

$$x_1 = \frac{(-1)}{a_{11}}(a_{12}x_2 + \dots + a_{1n}x_n) + \frac{b_1}{a_{11}}$$

Исключаем из x_1 всех уравнений по симплекс – правилу:

$$\begin{bmatrix} p & q \\ r & s \end{bmatrix} \to \begin{bmatrix} p & q \\ 0 & s - \dfrac{qr}{p} \end{bmatrix}$$

где p - ведущий элемент; q - элемент ведущей строки; r - элемент ведущего столбца; s - произвольный элемент.

Ведущий элемент выбирается по главной диагонали матрицы; из произвольного элемента вычитается произведение элементов ведущей строки на ведущий столбец, деленное на ведущий элемент.

При i=1 имеем $a_{ij}^{(1)} = a_{ij} - \dfrac{a_{1j}a_{i1}}{a_{11}}$ $b_i^{(1)} = b_1 - \dfrac{a_{i1}b_1}{a_{11}}$, j=i,…,n

В общем случае, $$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \frac{a_{ik}^{(k-1)}a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}},$$

$$b_i^{(k)} = b_k - \frac{a_{ik}b_k}{a_{kk}},$$

где k-номер шага $i=k+1,\dots n-1,\ j=i,2\dots n$ (3)

Получим

$$\begin{pmatrix} a_{11} & a_{12} & & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & & \cdots & \\ 0 & 0 & a_{33}^{(2)} & & \vdots \\ 0 & 0 & 0 & \ddots & \\ 0 & 0 & 0 & 0 & a_{nn}^{(n-1)} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(1)} \\ \\ \vdots \\ b_n^{(n-1)} \end{pmatrix}$$

Обратный ход, начиная с последнего уравнения, последовательно определяем x_n, x_{n-1}, …, x_1:

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}}$$

$$x_k = \frac{b_k^{(k-1)} - \sum_{j=k+1}^{n} a_{kj}^{(k-1)} x_j}{a_{kk}^{(k-1)}} \quad (4)$$

Замечание!

На основе прямого хода путем перемножения ведущих элементов вычисляется определитель матрицы A.

Изложенный метод имеет ограничение, связанное с тем, что ведущие элементы $a_{11}, a_{22}, \ldots, a_{nn}$ и т.д. должны быть отличны от нуля и не должны быть малыми по модулю, поскольку погрешности вычислений будут большими.

Таким образом, порядок последовательности исключения неизвестных может сильно сказаться на результатах расчетов. Уменьшить опасность такого рода позволяют модифицированные варианты метода Гаусса. Наиболее надежным является метод исключения с выбором главного элемента.

В методе Гаусса-Жордана (с выбором главного элемента) в качестве ведущих элементов выбираются максимальное по модулю a_{ii} путем перебора этих элементов по столбцу, соответствующему этому ведущему элементу или по всем столбцам.

Метод Гаусса с выбором главного элемента по столбцу

Перед исключением x_1 отыскивается $\max|a_{i1}|$ по i. Допустим, максимум соответствует $i=i_0$. Тогда первое уравнение в исходной системе меняем местами с i_0 уравнением. После этого осуществляем первый шаг исключения. Затем перед исключением x_2 из оставшихся уравнений отыскиваем $\max |a_{i2}|$, где $2<=i<=n$, осуществляется соответствующая перестановка уравнений.

Можно показать, что условие диагонального преобладания остается справедливым после каждого шага исключений в процессе приведения матрицы к треугольному виду, т.е.

$$\left|a_{ii}^{(k)}\right| > \sum_{\substack{j=k \\ i\neq j}}^{n} \left|a_{ij}^{(k)}\right|, \text{ для всех } k=1,2,\ldots,n-1. \quad (5)$$

Это означает, что перед каждым исключением очередной неизвестной главный элемент будет находиться в «нужной позиции».

Рассмотренные модификации метода Гаусса позволяют, как правило, существенно уменьшить влияние погрешности округления на результаты расчетов.

Метод Гаусса с выбором главного элемента.

$$\begin{vmatrix} a_{11} & \ldots & 0 & \ldots & a_{1n} \\ & & \vdots & & \\ \vdots & & a_{pq} & & \vdots \\ & & \vdots & & \\ a_{n1} & \ldots & 0 & \ldots & a_{nn} \end{vmatrix}$$

При первом ходе производится выбор наибольшего по модулю элемента и выполняется перестановка строк или столбцов. Это повышает точность вычисления. Затем проводится преобразование элементов по мнемоническому правилу, т.е. запоминается главная строка, главный столбец обнуляется, получаем новую матрицу с меньшим числом строк и столбцов. Вновь определяем главный элемент. Преобразование проводим до тех пор, пока не останется строка из двух элементов, т.о. получаем треугольную матрицу, далее начинается обратный ход (4).

2.2 Метод Холесского

Метод опирается на возможность представления матрицы в виде произведения двух треугольных матриц:

A=LU.

L - верхняя треугольная матрица; *U* - нижняя треугольная матрица.

Решение системы сводится к последовательному решению двух простых систем с треугольными матрицами.

Замечание. Всякую квадратную матрицу A, имеющую отличные от нуля $|a_{11}|$, $\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$,... главные миноры можно представить в виде LU-разложения, причем это разложение будет единственным.

Из соотношения *A=LU* следует:

$$\begin{bmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & \dots & \dots & l_{nn} \end{bmatrix} \cdot \begin{bmatrix} 1 & u_{12} & u_{13} & \dots & u_{1n} & u_{1n+1} \\ 0 & 1 & u_{23} & \dots & u_{2n} & u_{2n+1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & u_{nn+1} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n+1} \\ & & \dots & \\ & & \dots & \\ & & \dots & \\ a_{n1} & a_{n2} & \dots & a_{nn+1} \end{bmatrix}$$

Получим для первой строки и столбца:

$$l_{i1} = a_{i1} \qquad i = 1,2,\dots,n$$

$$u_{ij} = \frac{a_{1j}}{l_{11}} \qquad j = 2,3,\dots,n+1$$

Аналогично для i>=j и i<j:

$$l_{ik} = a_{ik} - \sum_{j=1}^{k-1} l_{ij} u_{jk} \qquad i = 1,2,\dots,n$$

$$u_{kj} = \frac{a_{kj} - \sum_{m=1}^{k-1} l_{km} u_{mj}}{l_{kk}} \qquad j = k+1,\dots,n+1 \tag{6}$$

Обратный ход:

$$x_n = u_{nn+1}$$

$$x_i = u_{in+1} - \sum_{k=i+1}^{n} u_{ik} x_k \qquad i = n-1,\dots,1 \tag{7}$$

<u>Замечание!</u>

При большом числе уравнений прямые методы решения систем линейных алгебраических уравнений становятся трудно реализуемыми на компьютере, прежде всего из-за сложности хранения промежуточных результатов и операций с большими матрицами.

Существуют различные способы представления матрицы А в виде

$A=A_1*A_2* \dots *A_m$,

Где каждая A_i имеет удобную форму для решения системы линейных уравнений. Как правило m<=5. Тогда в результате решения последовательности систем

$A_1b_1=b_0$, $A_2b_2=b_1$, …, $A_mb_m=b_{m-1}$ можно найти искомое решение.

3. Итерационные методы решения систем линейных уравнений

Альтернативой прямым методам являются итерационные, основанные на многократном уточнении $x^{(0)}$ – приближенного решения задачи Ax=в. Верхним индексом в скобках обозначается номер итерации (совокупности повторяющихся действий).

3.1 Метод трехточечной прогонки

Общая постановка задачи.

Дана система линейных алгебраических уравнений с трехдиагональной матрицей с диагональным преобладанием.

$$a_i x_{i-1} - b_i x_i + c_i x_{i+1} = f_i,\ a_1 = c_n = 0, i = 1,2,\ldots,n \tag{8}$$

Требуется найти решения $x=(x_1,x_2,\ldots,x_n)$ системы.

Приводимый ниже алгоритм носит название метода трехточечной прогонки и является специальным случаем метода исключения Гаусса. Рассмотрим следующую линейную систему:

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & \ldots & 0 \\ a_2 & b_2 & c_2 & 0 & \ldots & 0 \\ 0 & a_3 & b_3 & c_3 & \ldots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \ldots & 0 & 0 & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}. \tag{9}$$

Чтобы начать исключение, разделим первое уравнение этой системы на диагональный элемент b_1 и используем обозначения $p_1 = \dfrac{c_1}{b_1}$ и $q_1 = \dfrac{d_1}{b_1}$. Предположим, что мы исключили все нулевые поддиагональные элементы в первых $\boldsymbol{i-1}$ строках. В этом случае система преобразуется к виду:

$$\begin{bmatrix} 1 & p_1 & 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & p_2 & 0 & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & & & & \vdots \\ 0 & \ldots & 0 & 1 & p_{i-1} & 0 & \ldots & 0 \\ 0 & \ldots & 0 & a_i & b_i & c_i & \ldots & 0 \\ \vdots & & & & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \ldots & 0 & 0 & 0 & 0 & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{i-1} \\ x_i \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{i-1} \\ d_i \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}. \tag{10}$$

Теперь, чтобы исключить поддиагональный элемент $\boldsymbol{a_i}$ в $\boldsymbol{i}$-й строке, умножим $\boldsymbol{(i-1)}$– ю строку на a_i и вычтем ее из $\boldsymbol{i}$-й строки. В результате i-я строка нашей системы преобразуется к виду

$$(b_i - a_i p_{i-1})x_i + c_i x_{i+1} = d_i - a_i q_{i-1}.$$

Чтобы получить единицу на главной диагонали матрицы, разделим i-ю строку на коэффициент $(b_i - a_i p_{i-1})$. В результате для элементов $\boldsymbol{p_i}$ и $\boldsymbol{q_i}$ в окончательном виде i-й строки получаем следующие формулы:

$$\begin{aligned} p_i &= \frac{c_i}{b_i - a_i p_{i-1}},\ i = 2,\ldots,n-1, \quad p_1 = \frac{c_1}{b_1}, \\ q_i &= \frac{d_i - a_i q_{i-1}}{b_i - a_i p_{i-1}},\ i = 2,\ldots,n, \quad q_1 = \frac{d_1}{b_1}. \end{aligned} \tag{11}$$

Продолжая исключение, получаем систему уравнений с двухдиагональной матрицей вида

$$\begin{bmatrix} 1 & p_1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & p_2 & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ 0 & \ldots & 0 & 1 & p_{n-2} & 0 \\ 0 & \ldots & 0 & 0 & 1 & p_{n-1} \\ 0 & \ldots & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{n-2} \\ q_{n-1} \\ q_n \end{bmatrix}. \tag{12}$$

Это позволяет записать рекуррентные формулы для вычисления неизвестных x_i

$$\begin{aligned} x_n &= q_n, \\ x_i &= p_i x_{i+1} + q_i,\ i = n-1,\ldots,1. \end{aligned} \tag{13}$$

В виду опасности обращения знаменателя в ноль возникает вопрос об устойчивости решения, таким условием является диагональное преобладание в исходной матрице: $|b_i| > |a_i| + |c_i|$ для всех i. В этом случае $|p_i|<1$

Док-во методом индукции:

Пусть $|p_{i-1}|<1$, тогда

$$|p_i| = \frac{|c_i|}{|b_i - p_{i-1} a_i|} \le \frac{|c_i|}{\left||b_i| - |p_{i-1}||a_i|\right|} < \frac{|c_i|}{\left||b_i| - |a_i|\right|} < \frac{|c_i|}{|c_i|} = 1 .$$

Утверждение доказано. Из того, что $|b_i - p_i a_i| > |c_i| >= 0$, то есть знаменатель в формулах отличен от нуля.

Отметим, что метод прогонки относится к экономичным методам. Экономичными называются методы, для которых число неизвестных требуемых арифметических операций пропорционально числу неизвестных. Этот метод требует порядка 8n операций, метод Гаусса порядка n^3 операций.

3.2 Метод простой итерации

Для решения систему линейных уравнений *Ax=b* приводим к каноническому виду:

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ &\qquad\qquad \ldots \\ &a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{aligned} \tag{14}$$

Выразим $x_1, x_2, \ldots, x_n$:

$$x_1 = \frac{b_1}{a_{11}} - \frac{a_{12}x_2}{a_{11}} - \ldots - \frac{a_{1n}x_n}{a_{11}}$$
$$\ldots$$
$$x_n = \frac{b_n}{a_{nn}} - \frac{a_{12}x_2}{a_{nn}} - \ldots - \frac{a_{nn-1}x_{n-1}}{a_{nn}}$$

Таким образом, получим *x=Cx+f* , где

$$C = \begin{pmatrix} 0 & -\frac{a_{12}}{a_{11}} & \ldots & -\frac{a_{1n}}{a_{11}} \\ \ldots & 0 & \ldots & \ldots \\ \ldots & \ldots & 0 & \ldots \\ -\frac{a_{12}}{a_{nn}} & \ldots & -\frac{a_{nn-1}}{a_{nn}} & 0 \end{pmatrix}, \quad f = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \ldots \\ \frac{b_n}{a_{nn}} \end{pmatrix}, \quad x^{(0)} = \begin{bmatrix} x_1^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix} . \qquad (15)$$

Итерационный процесс запишется виде:

$$x^{(k+1)} = Cx^{(k)} + f \qquad (16)$$

Замечание!

Процесс называется параллельным итерированием, так как для вычисления (к+1)-го приближения всех неизвестных учитываются вычисленное ранее их к-е приближение. Начальное приближение $x^{(0)}$ выбирается произвольно.

Возникают вопросы об условии сходимости, и какова погрешность. Ответ на вопрос о сходимости дают следующие две теоремы

Теорема 1. *Для сходимости итераций (16) к решению системы (14) достаточно, чтобы в какой-либо норме выполнялось условие* $\|c\| \le q < 1,$ *тогда независимо от выбора* $x^{(0)}$ *выполняется*

$$\left\|x^{(k)} - x^*\right\| \le q^k \left\|x^{(0)} - x^*\right\|,$$ *где x* - точное решение (14).*

Доказательство:

Из (14) имеем x*=Cx*+f .

Вычитаем из (16) $x^{(k)}-x^*=C(x^{(k-1)} - x^*)$.

Получим оценку погрешности

$$\left\|x^{(k)} - x^*\right\| \le \|P\| \cdot \left\|x^{(k-1)} - x^*\right\| \le q\left\|x^{(k-1)} - x^*\right\| \le \ldots \le q^k \left\|x^{(0)} - x^*\right\|$$

очевидно, что при q<1 $\lim\limits_{k\to\infty} x^{(k)} = x^*$.

Замечание!

Условие теоремы, как достаточное, является завышенным.

Сходящийся процесс обладает свойством самоисправляемости.

Условие сходимости выполняется если в матрице А диагональные элементы преобладают. Иначе модули диагональных коэффициентов в каждом уравнении системы больше суммы модулей недиагональных коэффициентов. Чем меньше величина нормы матрицы С, тем быстрее сходится метод.

Теорема 2. *Для сходимости итераций (16) к решению системы (14) необходимо и достаточно, чтобы все собственные значения матрицы С по абсолютной величине были меньше 1.*

Замечание!

Хотя теорема 2 дает более общее условие сходимости метода простых итераций, однако ею воспользоваться сложнее, так как нужно вычислить границы собственных значений.

$$\sum_{j=1}^{n} |c_{ij}| \le \alpha < 1 \qquad i = 1,2,\ldots,n$$
$$\sum_{i=1}^{n} |c_{ij}| \le \beta < 1 \qquad j = 1,2,\ldots,n \tag{17}$$

Для того, чтобы данный процесс сходился, необходимо, чтобы норма матрицы C была меньше 1 или выполнялись следующие условия:

$$|x_i - x_i^{(k)}| \le \frac{\alpha}{1-\alpha} \max_{j=1,\ldots,n} |x_j^{(k)} - x_j^{(k-1)}|$$
$$|x_i - x_i^{(k)}| \le \frac{\beta}{1-\beta} \sum_{j=1}^{n} |x_j^{(k)} - x_j^{(k-1)}| \tag{18}$$

Метод работает лучше, если диагональные элементы значительно превосходят остальные.

Метод имеет ряд преимуществ:

- погрешность округления сказывается значительно меньше, чем в методе Гаусса;

- метод самоисправляющийся, отдельные ошибки при определении очередного вектора могут рассматриваться как новый начальный вектор;

- метод удобен в разрешенных матрицах (в коэффициентах стоит много нулей);

- процесс легко программируется.

Количество операций для выполнения этого метода приблизительно равно $(2n^2)k$, где k количество приближений. Если допустимая погрешность достигается при $k < \frac{n}{3}$, то метод итераций становится предпочтительнее метода Гаусса.

Кроме того, методы итераций могут оказаться предпочтительнее с точки зрения устойчивости вычислений.

Методика решения задачи

Преобразовать систему $Ax=b$ к виду $x=cx+f$

Задать начальное приближение $x^{(0)}$ и точность ε. Положить $k=0$.

Вычислить приближение по формуле $x^{(k+1)}=cx^{(k)}+f$

Если выполнено условие $\left\| x^{(k+1)} - x^{(k)} \right\| < \varepsilon$, то процесс завершить и положить $x^*=x^{(k+1)}$, иначе $k=k+1$ перейти к пункту 3.

Метод Зейделя

Является модифицированным методом простой итерации, дает лучшую сходимость и экономит память. При нахождении (k+1)- приближения сразу же используются найденные значения k компонент приближения с меньшим номером.

$$x_1^{(k)} = \frac{1}{a_{11}}(f_1 - a_{12}x_2^{(k-1)} - a_{13}x_3^{(k-1)} - \ldots - a_{1n}x_n^{(k-1)})$$

$$x_2^{(k)} = \frac{1}{a_{22}}(f_2 - a_{21}x_1^{(k)} - a_{23}x_3^{(k-1)} - \ldots - a_{2n}x_n^{(k-1)}) \tag{19}$$

$$x_m^{(k)} = \frac{1}{a_{mm}}(f_m - a_{m1}x_1^{(k)} - \ldots - a_{mm-1}x_{m-1}^{(k)} - a_{mm+1}x_{m+1}^{(k-1)} \ldots - a_{mn}x_n^{(k-1)}).$$

Замечание!

1) Метод Зейделя гарантированно сходится:

а) выполняется условие диагонального преобладания, является достаточным условием, но не является необходимым;

б) для матриц симметричных и положительно определенных.

2) В одинаковых условиях метод Зейделя сходится примерно в два раза быстрее метода простых итераций. Метод Зейделя может сходиться, если расходится метод простых итераций и наоборот. Также является само-исправляющимся, удобным в пользовании на ЭВМ, экономит память.