**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра ИС**

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Объектно-ориентированное программирование»**

|  |  |  |
| --- | --- | --- |
| Студенты гр.1376 |  | Байлук А.А.  Сухарева В.А. |
| Преподаватель |  | Егоров С.С. |

Санкт-Петербург

2024

**1. Задание на практическую работу.**

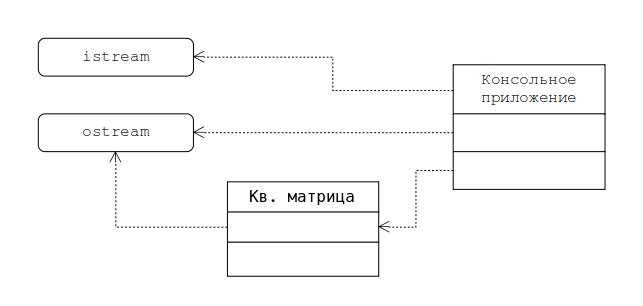


Рис.1 – Диаграмма классов работы №1

Создать консольное приложение согласно представленной на рис.1 диаграмме классов, предназначенное для заданных вычислений над квадратной матрицей, заданной на **множестве вещественных чисел**. Для этого необходимо специфицировать пользовательские классы "Консольное приложение" и "Квадратная матрица", т.е. задать атрибуты и методы указанных классов, а также распределить их по существующим областям видимости. Спецификация классов и реализация их методов должна обеспечивать реализацию отношений, указанных на диаграмме классов. **В отчете представить аргументированное обоснование своего выбора**.

Приложение должно включать основной модуль (функция main), модуль «application» и модуль «matrix».

В **основном модуле** консольного приложения (*для языка С++ - это модуль с функцией main*) должен создаваться объект класса "Консольное приложение" и вызываться его метод, который предоставляет пользователю **меню команд** приложения.

Модуль **«application»** должен содержать спецификацию класса "Консольное приложение" и реализацию его методов. Один из его методов должен выводить в консоль меню команд приложения, включающее:

- команду, инициирующую ввод с консоли значений, задающих объект матрицы (до ввода в программе должна быть задана матрица по умолчанию);

- команду, инициирующую расчет определителя матрицы и вывод результатов расчета;

- команду, инициирующую формирования транспонированной матрицы и ее вывода в консоль;

- команду, инициирующую расчет ранга матрицы и вывод результатов расчета;

- команду, инициирующую представление в консоль текущего объекта матрицы;

- команду выхода из приложения.

Модуль **«matrix»** должен содержать спецификацию класса "Квадратная матрица" и реализацию его методов, необходимых для достижения цели разрабатываемого приложения. Описание класса должно использовать вместо типа double (вещественное число, заданное в условии) абстрактный тип ***number***, описание которого должно задаваться в отдельном заголовочном файле number.h с помощью оператора **typedef double number** (для С++).

**ВАЖНО.** Основное требование к реализации класса **«matrix»** заключается в том, что она должна быть инвариантна (одна и та же для различных вариантов использования) ко множеству применимых объектов. Здесь задано множество определения элементов матрицы как вещественное, а может быть и множества целых, комплексных, рациональных и т.д. Для различных множеств при сохранении функциональности реализация класса должна быть одна и та же. Если для различных множеств менять реализацию, то теряется смысл ООП как парадигмы.

Требуется реализовать и отладить программу, удовлетворяющую сформулированным требованиям и заявленным целям. Разработать контрольные примеры и оттестировать на них программу. Оформить отчет, сделать выводы по работе.

**2. Спецификация классов**

Класс **Matrix –** класс программы, предназначенный для работы с матрицей. Методы присутствующие в класс: заполнение матрицы с выбором ее размера, определение определителя, транспонирование и вывод транспонированной матрицы, определение ранга, вывод матрицы на экран.

SetMatrix – публичный метод, создает матрицу размера заданного пользователем и предоставляет возможность заполнить матрицу пользователю.

FindDet – публичный метод, ссылающийся на приватный рекурсивный метод вычисления определителя.

TransposeMatrix – публичный метод, транспонирующий и выводящий матрицу.

Rang – публичный метод, вычисляющий ранг матрицы.

Show – публичный метод, выводящий матрицу.

matrix - приватный атрибут типа number \*\*, хранит значения элементов матрицы;

size - приватный атрибут типа int, хранит значения размера матрицы;

chr – приватный атрибут типа char, используется вместе с функцией \_getch() для перехода между состоянием программы.

SetCursorPosition – приватный метод, задающий положение курсора в командной строке.

findDet\_private – приватный метод, рекурсивно вычисляющий определитель.

clearMemory – приветный метод, освобождающий память при работе с функцией findDet\_private.

Wait – приватный метод, ожидающий нажатия клавиши для перехода программы в другое состояние.

Класс **Application –** класс программы предназначенный для взаимодействия пользователя с программой. При запуске программы пользователя перемещает в пользовательское меню(UserSpot), где он начинает работу с матрицей.

UserSpot – публичный метод, направляющий пользователя в меню взаимодействия с матрицей.

SetCursorPosition – приватный метод, задающий положение курсора в командной строке.

UpCur – приватный метод, поднимающий курсор выше.

DownCur – приватный метод, опускающий курсор.

Menu – приватный метод, выводящий меню в консоль.

mrtx – приватный атрибут, созданный для взаимодействия пользователя с матрицей.

cursorX – приватный атрибут, хранящий значения курсора по оси Ох.

cursorY – приватный атрибут, хранящий значения курсора по оси Оу.

chr – приватный атрибут типа char, используется вместе с функцией \_getch() для перехода между состоянием программы.

**3. Диаграмма классов.**

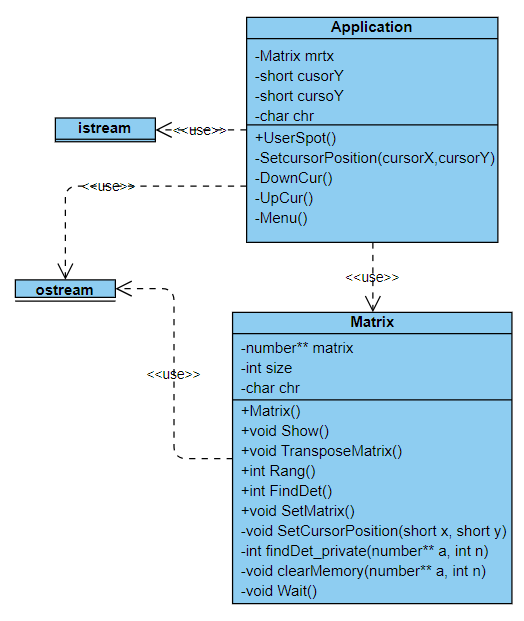


Рис.2 – Диаграмма классов

**4. Описание контрольных примеров с исходными и ожидаемыми данными.**

**Пример №1**

Данные вводимые пользователем mrtx =

Ожидаемые данные: rank(mrtx) = 3, det = 2.

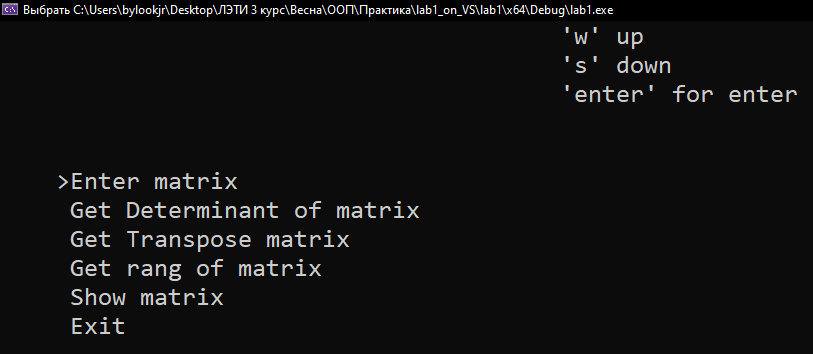


Рис.3 – Начальный экран приложения, курсор на первой вкладке.

Переходим во вкладку «Enter Matrix». Вводим значения размера квадратной матрицы и заполняем ее. Для того чтобы выйти из вкладки необходимо нажать «е».

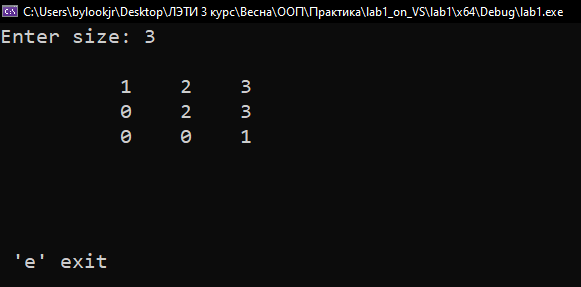


Рис.4 – Заполненная вкладка «Enter Matrix».

Вернувшись на начальный экран переходим во вкладку «Get Determinant of matrix».

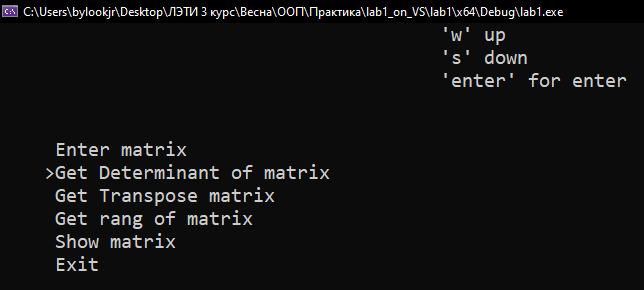


Рис.5 – Начальный экран приложения, курсор на второй вкладке.

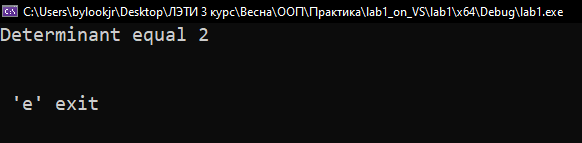


Рис.6 – Результат работы второй вкладки.

Вернувшись на начальный экран переходим во вкладку «Get Transpose matrix».

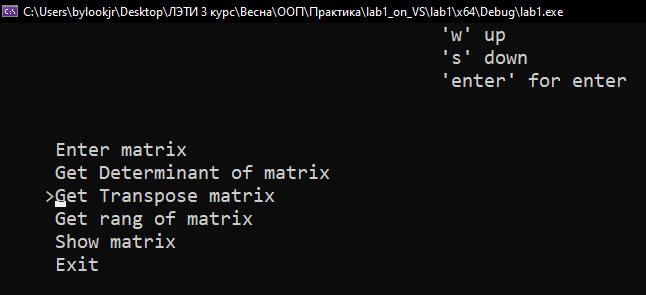


Рис.7 – Начальный экран приложения, курсор на третьей вкладке.

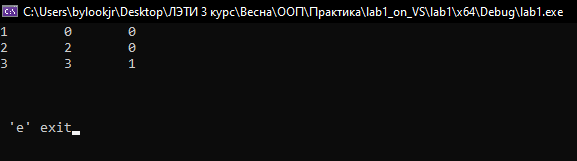


Рис.8 – Результат работы третьей вкладки.

Вернувшись на начальный экран переходим во вкладку «Get rang of matrix».

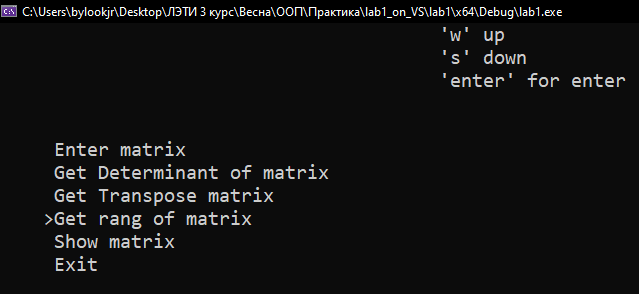


Рис.9 – Начальный экран приложения, курсор на четвертой вкладке.

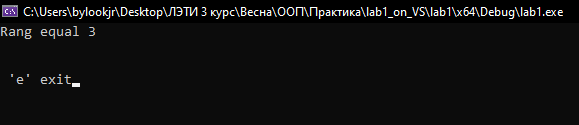


Рис.10 – Результат работы четвертой вкладки.

Вернувшись на начальный экран переходим во вкладку «Show matrix».

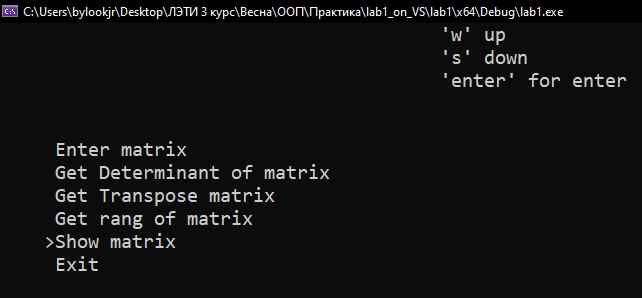


Рис.11 – Начальный экран приложения, курсор на пятой вкладке.

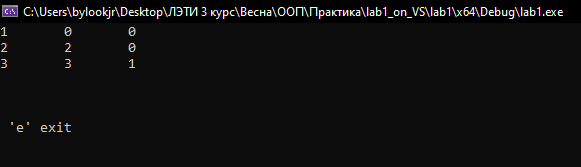


Рис.12 – Результат работы пятой вкладки.

Вернувшись на начальный экран переходим во вкладку «Exit», нажимая ‘enter’ приложение заканчивает свою работу.

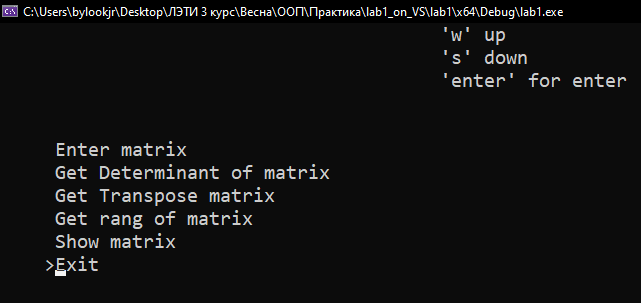


Рис.13 - Начальный экран приложения, курсор на пятой вкладке.

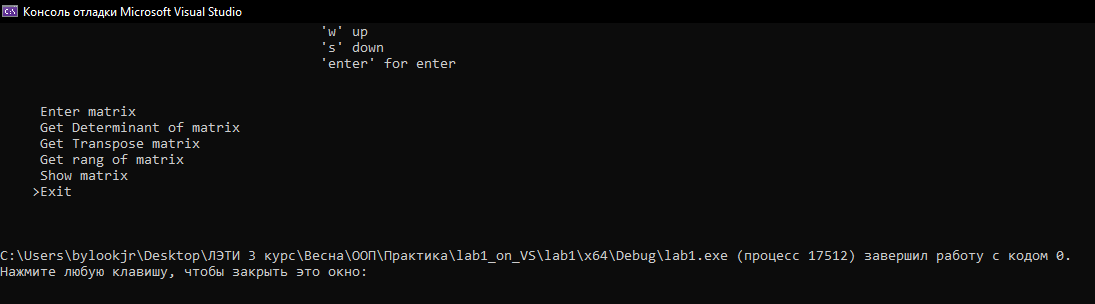


Рис.14 – Результат работы шестой вкладки, выход из приложения.

**Вывод**: в ходе данной практической работы было создано консольное приложение для работы с матрицей, были специфицированы классы Application и Matrix, реализованы их методы, была построена диаграмма классов.