МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационных систем

ОТЧЕТ

по практической работе №5
по дисциплине «Объектно-ориентированное программирование»
Тема: Распределенное приложение, включающее клиентскую и серверную части, для вычислений над квадратной матрицей на множестве рациональных чисел

Студенты гр. 1361 _____ Горбунова Д.А.
_____ Кравцов И.Ю.
Преподаватель _____ Егоров С.С.

Санкт-Петербург
2024

ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ

Тема работы: создать распределенное приложение, включающее клиентскую и серверную части, взаимодействующие посредством сетевого обмена сообщениями.

Исходные данные:

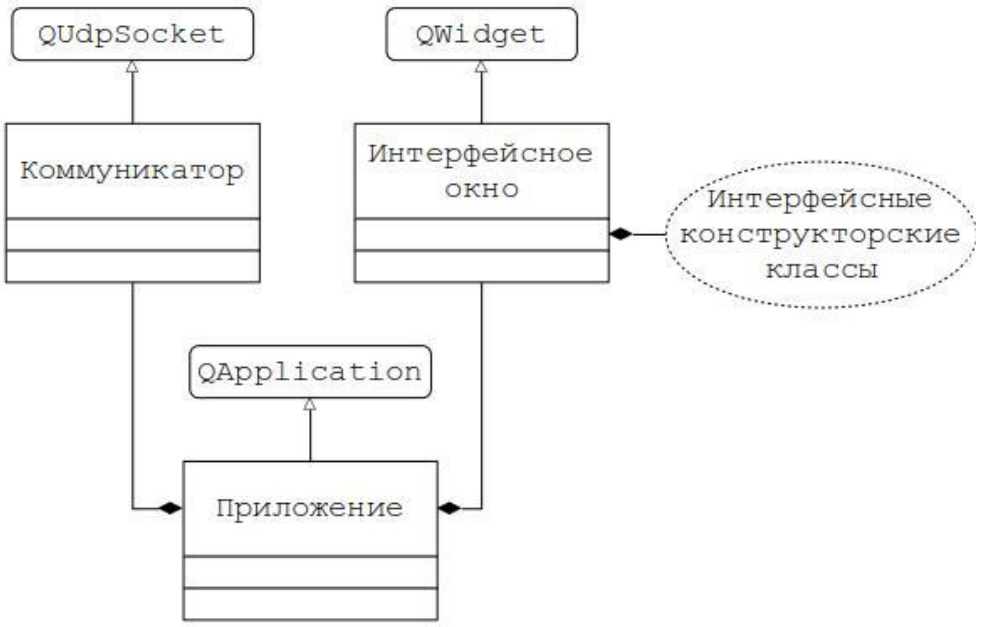

Рис.1 – Диаграмма классов клиентской части работы 5

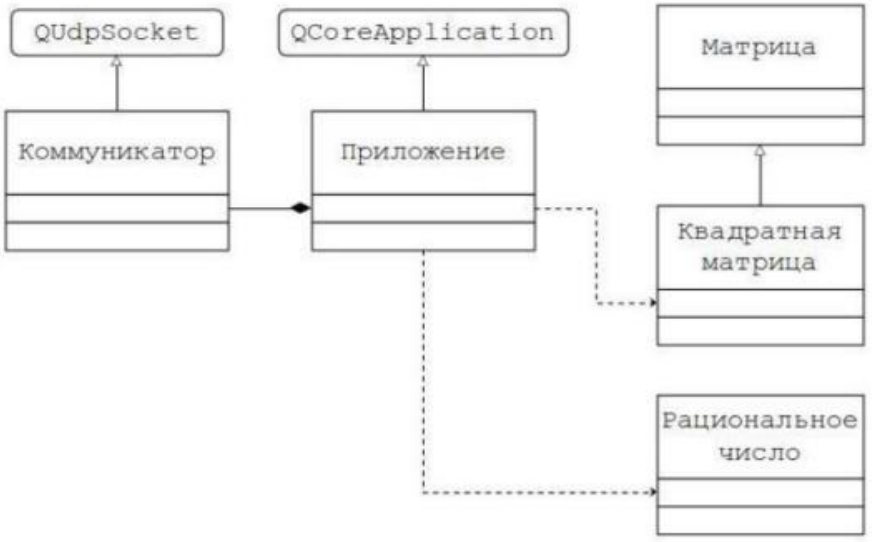

Рис.2 – Диаграмма классов серверной части работа 5

Описание работы:

Создать распределенное приложение, включающее клиентскую и серверную части, взаимодействующие посредством сетевого обмена сообщениями.

Клиентские и серверные части представляют собой приложения, реализованные в работе №4.

Отличие заключается в том, что класс «Квадратная матрица» делается наследников класса «Матрица» с произвольным числом строк и столбцов (отношение обобщения). Это влечет за собой перенос некоторых атрибутов и методов класса «Квадратная матрица» в родительский класс «Матрица», что и предстоит сделать в этой работе.

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

СОДЕРЖАНИЕ

1. СПЕЦИФИКАЦИЯ РАЗРАБОТАННЫХ КЛАССОВ

Классы проекта серверного приложения:

1.1. Класс `Matrix`

Атрибуты класса

a) `Int row, int column`

Область видимости: private

Назначение: хранят в себе строки и столбцы матрицы.

b) `vector<vector<number>> m_values;`

Область видимости: private

Назначение: хранит в себе массив матрицы.

Методы класса:

a) `Matrix();`

Область видимости: public

Назначение: Конструктор по умолчанию

b) `Matrix (int rows, int cols);`

Область видимости: public

Назначение: Конструктор с параметрами

c) `void generate_1();`

Область видимости: public

Назначение: метод заполнения матрицы в виде единичной

d) `void print_screen();`

Область видимости: public

Назначение: метод вывода матрицы в консоль

e) `number determinant() const;`

Область видимости: public

Назначение: метод вычисления определителя

f) `void transpose();`

Область видимости: public

Назначение: метод транспонирования метрицы (был переписан)

g) `void CinMatrix(int n);`

Область видимости: public

Назначение: метод записывания матрицы в память

h) `int rows() const;`

Область видимости: public

Назначение: метод получения количества строк

i) `int cols() const;`

Область видимости: public

Назначение: метод получения количества столбцов

j) `int rank();`

Область видимости: public

Назначение: метод вычисления ранга

k) `void print_screen(std::stringstream& ss) const;`

Область видимости: public

Назначение: метод вывода матрицы в строковую переменную

l) `void setMatrixValue(int row, int col, int num, int det);`

Область видимости: public

Назначение: метод записи по элементно матрицу

m) `void setDimension(int dimension);`

Область видимости: public

Назначение: метод изменение размера матрицы

n) `number getElement(int row, int col);`

Область видимости: public

Назначение: метод получения единичного элемента в матрицы

1.2 Класс SquareMatrix

Атрибуты класса:

Отсутствуют

Методы класса:

a) `determinantHelper`

Область видимости: private

Назначение: Этот метод используется внутри класса SquareMatrix для вычисления определителя матрицы.

b) `determinant`

Область видимости: public

Назначение: Этот метод используется для вычисления определителя квадратной матрицы.

c) `transpose`

Область видимости: public

Назначение: Этот метод используется для транспонирования квадратной матрицы.

d) `rank`

Область видимости: public

Назначение: Этот метод используется для вычисления ранга квадратной матрицы.

2. ДИАГРАММА КЛАССОВ

Диаграмма классов представлена на рисунке 2 и 3. На них обозначены атрибуты классов, их методы и операторы. Знаками «+» обозначены методы, атрибуты, имеющие общедоступный тип доступа и операторы; соответственно «-» – приватный. Также через двоеточие указаны типы возвращаемых значений, где они есть. В круглых скобках представлены типы параметров соответствующих методов, операторов, конструкторов. На рисунке 2 показана диаграмма классов клиентской части, а на рисунке 3 диаграмма классов серверной части.

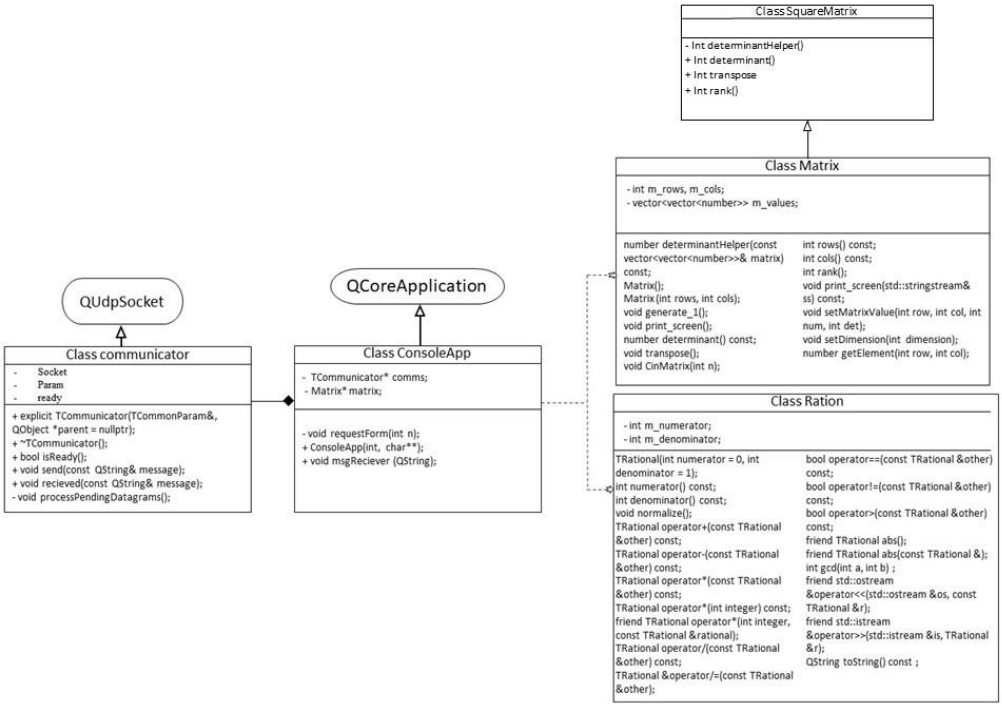

Рисунок 4 – Диаграмма классов со стороны сервера

3. ОПИСАНИЕ КОНТРОЛЬНОГО ПРИМЕРА

Исходные данные для контрольного примера:

$$A = \begin{pmatrix} \frac{1}{2} & \frac{3}{4} \\ \frac{7}{5} & \frac{8}{3} \end{pmatrix}$$

Чтобы найти транспонированную матрицу поменяем строки столбцы матрицы местами:

$$\mathbf{A^T} = \begin{pmatrix} \frac{1}{2} & \frac{7}{5} \\ \frac{3}{4} & \frac{8}{3} \end{pmatrix}$$

Далее ищем определитель матрицы:

$$\det A = \begin{vmatrix} \frac{1}{2} & \frac{3}{4} \\ \frac{7}{5} & \frac{8}{3} \end{vmatrix} = \frac{1}{2} * \frac{8}{3} - \frac{7}{5} * \frac{3}{4} = \frac{4}{3} - \frac{21}{20} = \frac{17}{60}$$

Т.к. $\det A \neq 0$, то линейно зависимых строк или столбцов нет $\Rightarrow$ $\text{rank } A = 2$.

4. РАБОТА ПРОГРАММЫ НА КОНТРОЛЬНЫХ ПРИМЕРАХ

При запуске приложения, программа выводит на экран доступные варианты работы с матрицей (рисунок 3).

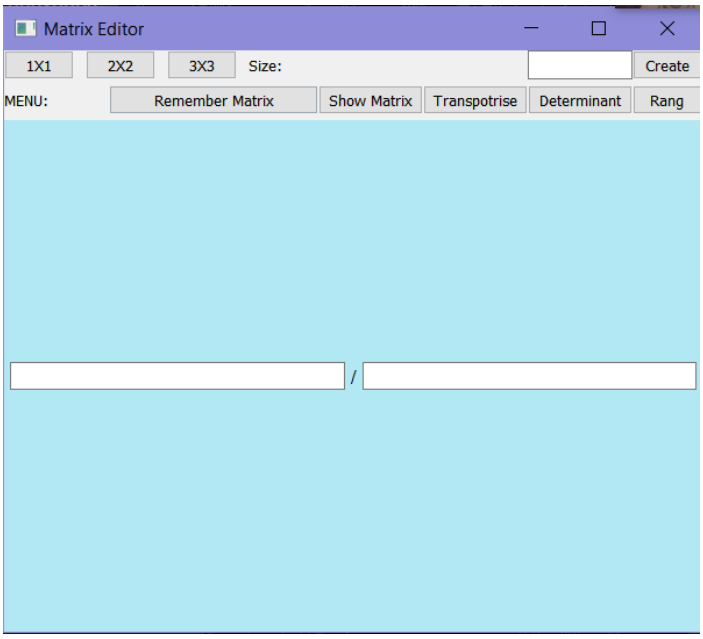

Рисунок 3 – Меню

Далее производим ввод пользовательской матрицы в приложении.

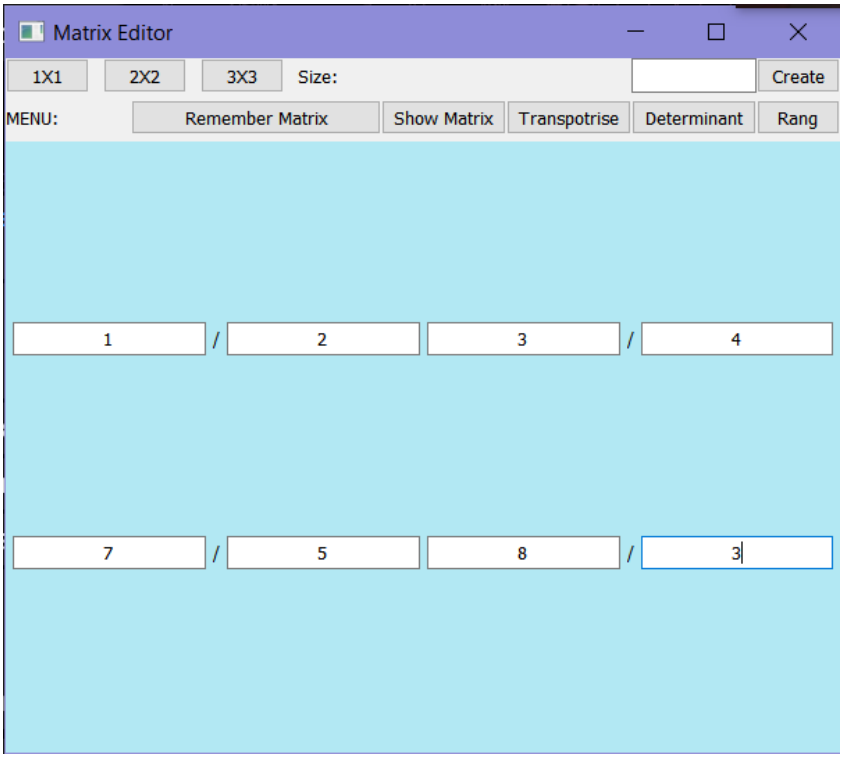

Рисунок 4 – Ввод пользовательской матрицы

После того как мы ввели коэффициенты, можем, нажав кнопку “Remember

Matrix”, записать данную матрицу в память. Приложение выведет следующее сообщение (рисунок 5).

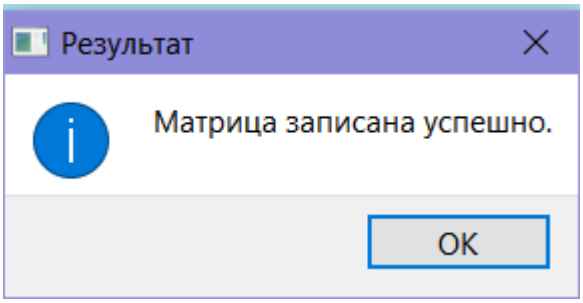

Рисунок 5 – Результат кнопки «Remember Matrix»

После того мы определили матрицу, ее можно вывести, нажав кнопку «Show Matrix» (рисунок 6).

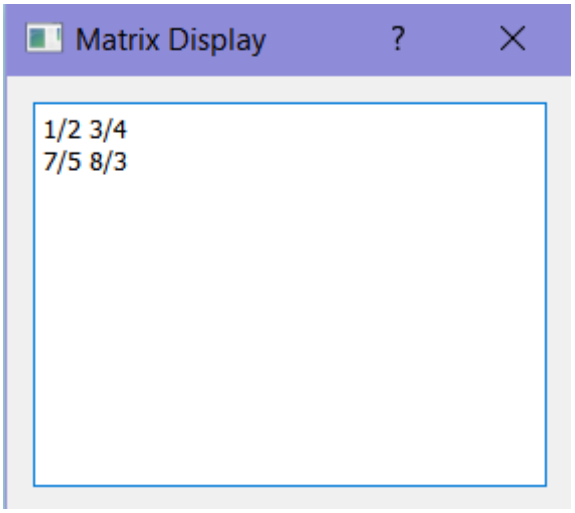

Рисунок 6 – Вывод матрицы.

После того, как у нас матрица задана, мы производим над ней вычисления. Сначала вычислим транспонированную матрицу, нажав кнопку“Transpotrise” (рисунок 7).

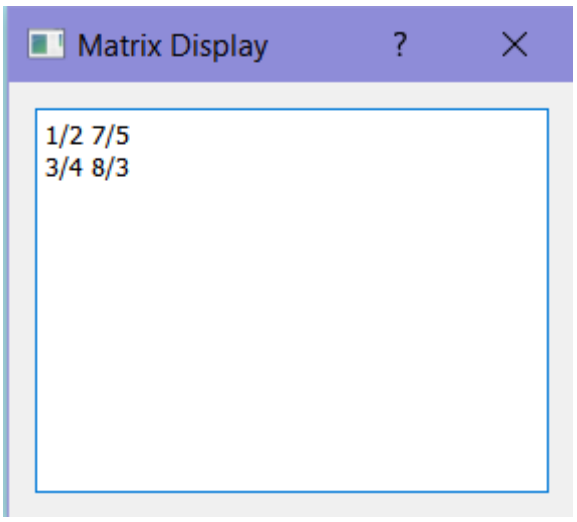

Рисунок 7 – Транспонирование матрицы.

Далее вычисляем ранг матрицы, нажав кнопку “Rang” (рисунок 8).

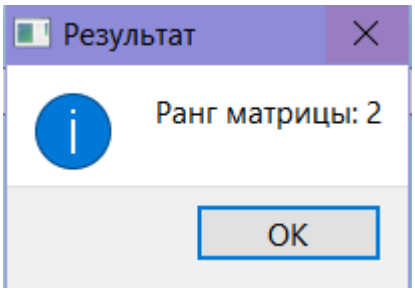

Рисунок 8 – Ранг матрицы.

Если же выбрать кнопку “Determinant, то программа вычислит определитель матрицы (рисунок 9).

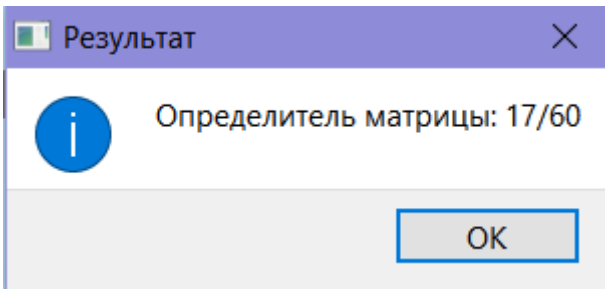

Рисунок 9 – Определитель матрицы.

Для завершения программы достаточно нажать на крест в верхнем правом углу.

При каждом действии сервер в ответ на запрос выводит сообщение об отправке ответа.

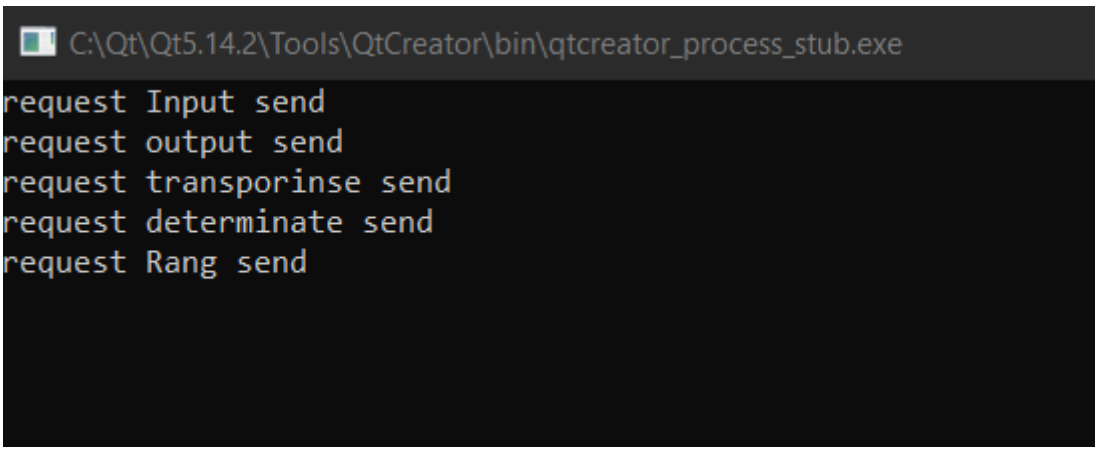

Рисунок 10 – результат работы сервера в консольном приложении

ВЫВОДЫ ПО ВЫПОЛНЕННОЙ РАБОТЕ

В результате выполнения данной практической работы на языке программирования C++ было написано GUI приложение, а также серверная часть приложения, предназначенное для заданных вычислений над квадратной матрицей, заданной на множестве рациональных чисел.

В ходе выполнения работы были реализованы следующие класса:

• SquareMatrix

• matrix

Были использованы области видимости классов, с объяснением использования.

Была определены диаграммы классов и написаны, соответствующие по структуре, приложение и серверная часть.

Для тестирования правильности работы приложения, был приведен контрольный пример и рассчитаны соответствующие значения. В результате тестирования все расчеты оказались верны.