РАЗРАБОТКА И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МАТРИЧНОГО КАЛЬКУЛЯТОРА

Сабирова Резеда Рафисовна

студент, Набережночелнинский институт (филиал) ФГАОУ ВО К(П)ФУ, РФ, г. Набережные Челны

При решении прикладных задач математики приходится сталкиваться не только с действительными или комплексными числами, но и с такими объектами как матрицы. Поиск решений систем линейных уравнений, разрешимость обыкновенных дифференциальных уравнений, большой раздел задач аналитической геометрии и линейной алгебры – все эти вопросы подразумевают под собой работу с матрицами. Собственно, для того чтобы избежать неудобств при решении этих задач и создается данное приложение.

Целью работы является создание графического приложения «Матричный калькулятор». Задачи, необходимые для ее достижения:

1. Анализ предметной области;
2. Проектирование системы классов;
3. Выбор языка программирования и среды разработки;
4. Программная реализация классов;
5. Разработка оконного приложения для тестирования классов;
6. Тестирование системы классов.

Матрица – множество чисел, образующих прямоугольную таблицу, которая содержит m-строк и n-столбцов [1]. Для обозначения матрицы используется надпись:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \end{pmatrix}$$

Под матрицей далее будет подразумеваться обыкновенный двумерный массив чисел, каковым, по сути, матрица и является.

В приложении «Матричный калькулятор» реализованы класс матрицы Matrix, с помощью которого будут выполняться основные действия над матрицами, и класс главного окна приложения MainWindow. Программа «Матричный калькулятор» написана на языке

Сабирова Р.Р. Разработка и программная реализация матричного калькулятора // Студенческий форум: электрон. научн. журн. 2017. № 21(21). URL: https://nauchforum.ru/journal/stud/21/29107 (дата обращения: 03.09.2025).

программирования C++. В качестве среды программирования используется QtCreator.

Объектами-членами класса ***Matrix*** являются указатель на динамическую память и размеры массива. В ходе работы для данного класса реализованы следующие методы:

· конструкторы и деструктор;

· функции доступа к объектам-членам класса;

· методы перегрузки операторов;

· функция умножения матриц;

· функция для вычисления определителя матрицы;

· функция для транспонирования матрицы.

Класс имеет приватные и общедоступные объекты-члены и функции-члены (методы). Для хранения компонентов матрицы используется динамический массив элементов типа double. Для создания объекта предусмотрены несколько конструкторов: конструктор по умолчанию, конструкторы с параметрами, конструктор копирования и деструктор. Для выполнения множества матричных операций перегружены операторы: с присваиванием (=, +=), сложения (+), умножения (*), логические (==, !=) и т.п.

Для реализации графического интерфейса создан класс главного окна приложения. В качестве класса, базового для класса окна приложения ***MainWindow***, при создании проекта используется класс QMainWindow.

Для разработки GUI используется редактор QtDesigner. QtDesigner представляет собой визуальный редактор-конструктор окон пользовательского интерфейса [2]. Заготовка будущего окна сделана в этом редакторе. В заготовку помещены виджеты. Для каждого виджета, включая само окно, выполнена настройка их параметров. Для сконструированного окна редактор сгенерировал текстовый файл, содержащий описание всех элементов управления, самого окна, включая совокупность их параметров.

Структурно окно приложения состоит из следующих компонентов: меню для сохранения матрицы в файл, загрузки матрицы из файла и выхода из программы; поля для ввода и вывода данных; кнопки для выбора операций с матрицами. Главное окно приложения представлено на рисунке 1.

Для класса окна приложения MainWindow были определены следующие открытые функции:

1. 1. Конструктор и деструктор. В конструкторе динамически создаются массивы. В деструкторе они удаляются.

2. Важными методами класса являются функции mtx2table и table2mtx, т.к. они используются во многих других функциях и слотах. Функция mtx2table осуществляет вывод матрицы в QTableWidget. Далее приведено описание функции:

```
void MainWindow::mtx2table(Matrix*&M,QTableWidget*&TW)
{
TW->clear();//очищается виджет
TW->setRowCount(M->getN());//задается кол-во ячеек,которые берутся из класса
TW->setColumnCount(M->getM());//матрицы
for(inti=0;i<M->getN();i++)
for(intj=0;j<M->getM();j++)
{
TW->setItem(i,j,newQTableWidgetItem(QString::number(M-
>getElement(i,j))));//задаются элементы
}
```

3. Функция table2mtx является обратной к вышеописанной. Она берет данные из виджета и копирует в класс матрицы.

4. Метод mtxSwap предназначен для обмена матрицы A и B местами.

5. Для копирования данных из одной матрицы в другую, и в виджеты используется функция mtxCopy.

6. Функция loadMtx нужна для загрузки матрицы из файла.

7. Функция saveMtx похожа на метод loadMtx, различие в том, что в ней файл открывается в режиме только для записи, а метод serialize берется с аргументами (out,true), в результате происходит сохранение матрицы в файл.

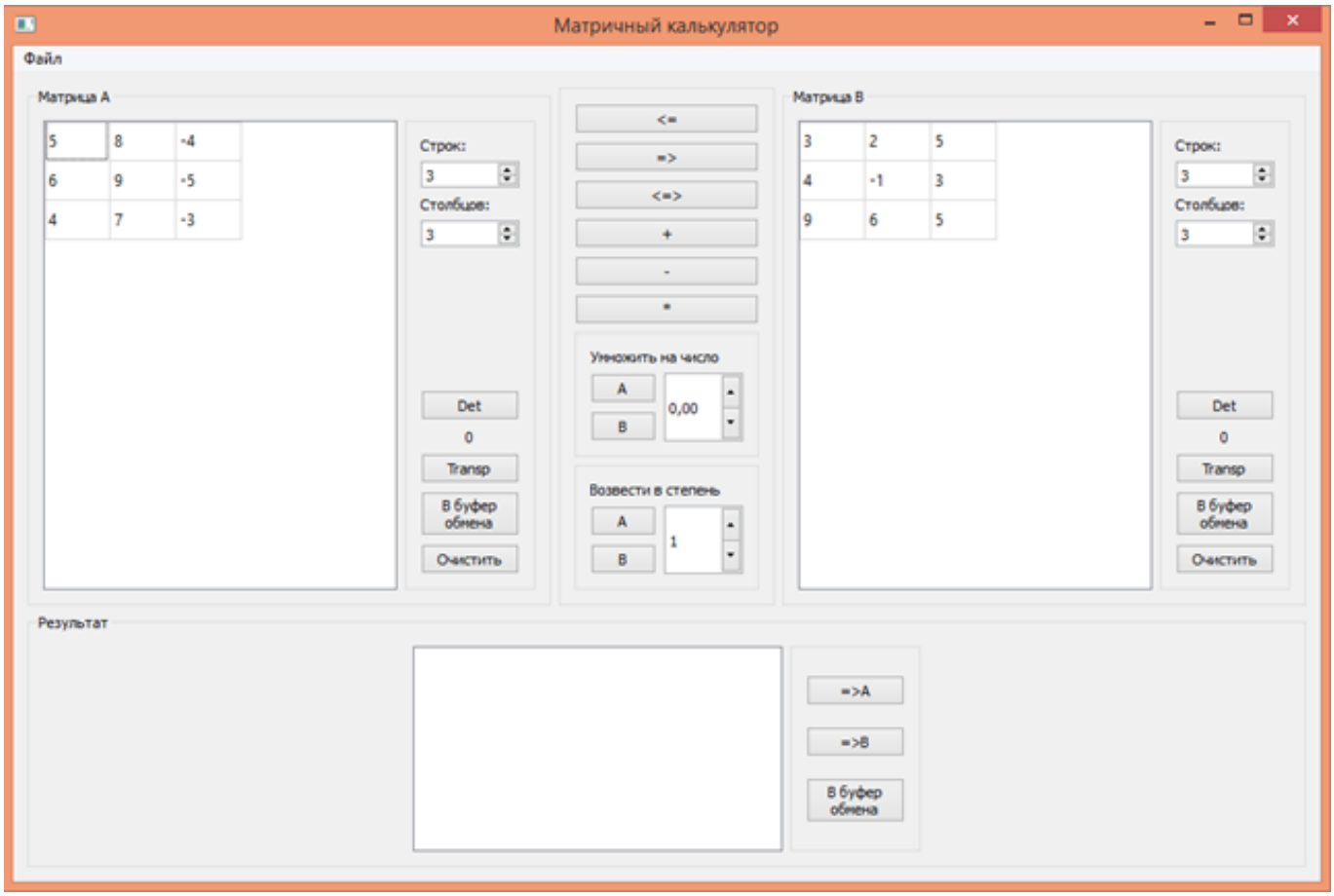

Рисунок 2.1. Главное окно приложения «Матричный калькулятор»

8. Для операций сложения, вычитания, умножения матриц требуется, чтобы матрицы имели соответствующие размеры. Чтобы избежать ошибок в программе, была создана функция update(). Если размеры матриц не подходят, то этот метод блокирует кнопки соответствующих операций.

9. В программе реализована возможность копирования матриц в буфер обмена. Для этого нужно преобразовать матрицу в строку. Функция mtx2string осуществляет это действие:

```
QString MainWindow::mtx2string(Matrix*&mtx)const
{
QString tmp;//Создается объект класса QString
tmp="["; //В начало строки ставится символ"["
for(inti=0;i<mtx->getN();i++)
{
for(intj=0;j<mtx->getM();j++)//В цикле происходит
{                                          // запись элементов матрицы в строку
tmp+=QString::number(mtx->getElement(i,j));
if(j!=mtx->getM()-1)
tmp+="";
}
if(i!=mtx->getN()-1)
tmp+=";";//Если ряд заканчивается,ставится символ";"
}
tmp+="]";//Когда все элементы записаны в строку,квадратные скобки закрываются
return tmp;//Возвращается символьная строка
}.
```

Фундаментальный класс виджетов – QWidget. Унаследован от QObject, следовательно может использовать механизм сигналов и слотов. Поэтому, для данного приложения были определены слоты.

Например, методы on_spH1_valueChanged и on_spW1_valueChanged меняют размеры матрицы A, on_pbDet1_clicked считает определитель матрицы A, on_pbTransp1_clicked транспонирует матрицу A. Для таких же действии с матрицей B созданы аналогичные функции.

Для того, чтобы убедиться в правильности работы созданного приложения, необходимо протестировать его. Для проверки корректности работы проведены несколько тестов. В ходе работы приложения не было выявлено ошибок. Поэтому можно считать, что получен корректно работающий матричный калькулятор.

Список литературы:

1. Кочетков, Е.С. Линейная алгебра: Учебное пособие / Е.С. Кочетков, А.В.Осокин. – М.: Форум, 2012.- 397с.

2. Липпман С. Б., Лажойе Ж. Язык программирования C++. Вводный курс: Пер. с англ. — 3-е изд. — М.: ДМК, 2001. — 1104 с.