МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**«Структура хранения матриц специального вида»**

**Выполнил:** студент группы 381706-1

Голубева А. С.

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

аспирант каф. МОСТ ИИТММ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г.

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc343339)

[2. Постановка задачи 4](#_Toc343340)

[3. Руководство пользователя 5](#_Toc343341)

[4. Руководство программиста 6](#_Toc343342)

[4.1 Описание структуры программы 6](#_Toc343343)

[4.2 Описание структур данных 6](#_Toc343344)

[4.3 Описание алгоритмов 8](#_Toc343345)

[5. Заключение 10](#_Toc343346)

[6. Литература 11](#_Toc343347)

# **Введение**

Матричные обозначения широко распространены в современной математике и её приложениях. Матрица – полезный аппарат для исследования многих задач теоретической и прикладной математики.

Помимо матриц общего вида, для которых наиболее естественной и наиболее часто используемой представляется программная реализация в виде двумерного массива, в математических приложениях выделяются различные матрицы специальных видов (треугольные, диагональные, …). Для таких матриц предпочтительно создание собственных способов хранения и обработки, учитывающих специфику их структуры, и потому более эффективных.

Матрица – в математике – прямоугольная таблица каких-либо элементов aij, состоящая из m строк и n столбцов. Набор элементов матрицы (a(1,1), a(2,2), …, a(n,n)) называется главной диагональю. Верхнетреугольной называется матрица, в которой все элементы под главной диагональю равны нулю.

Определение матрицы возможно также через понятие Вектор. Вектор – в математике – набор ai состоящий из n элементов. Тогда Матрица из m строк и n столбцов может быть определена как Вектор из n элементов, где каждый элемент, в свою очередь, является вектором из m элементов.

Целью данной лабораторной работы является разработка структуры хранения верхнетреугольных матриц.

# **Постановка задачи**

В рамках лабораторной работы ставится задача создания программных средств, поддерживающих эффективное хранение матриц специального вида (верхнетреугольных) и выполнение основных операций над ними:

1. Сложение, вычитание
2. Умножение
3. Деление
4. Сравнение (проверка на равенство и неравенство)
5. Присваивание одной матрицы другой
6. Ввод, вывод

Выполнение работы предполагает решение следующих задач:

1. Реализация шаблонного класса TVector.
2. Реализация шаблонного класса TMatrix.
3. Реализация класса TException для обработки исключений.
4. Реализация тестов и обеспечение их работоспособности.
5. Пример использования и обеспечение его работоспособности.

# **Руководство пользователя**

Рассмотрим пример работы программы.

Пользователю предлагается ввести размер матриц, затем требуется ввести сами матрицы. На экран выводится результат сложения, вычитания, умножения и деления этих матриц.

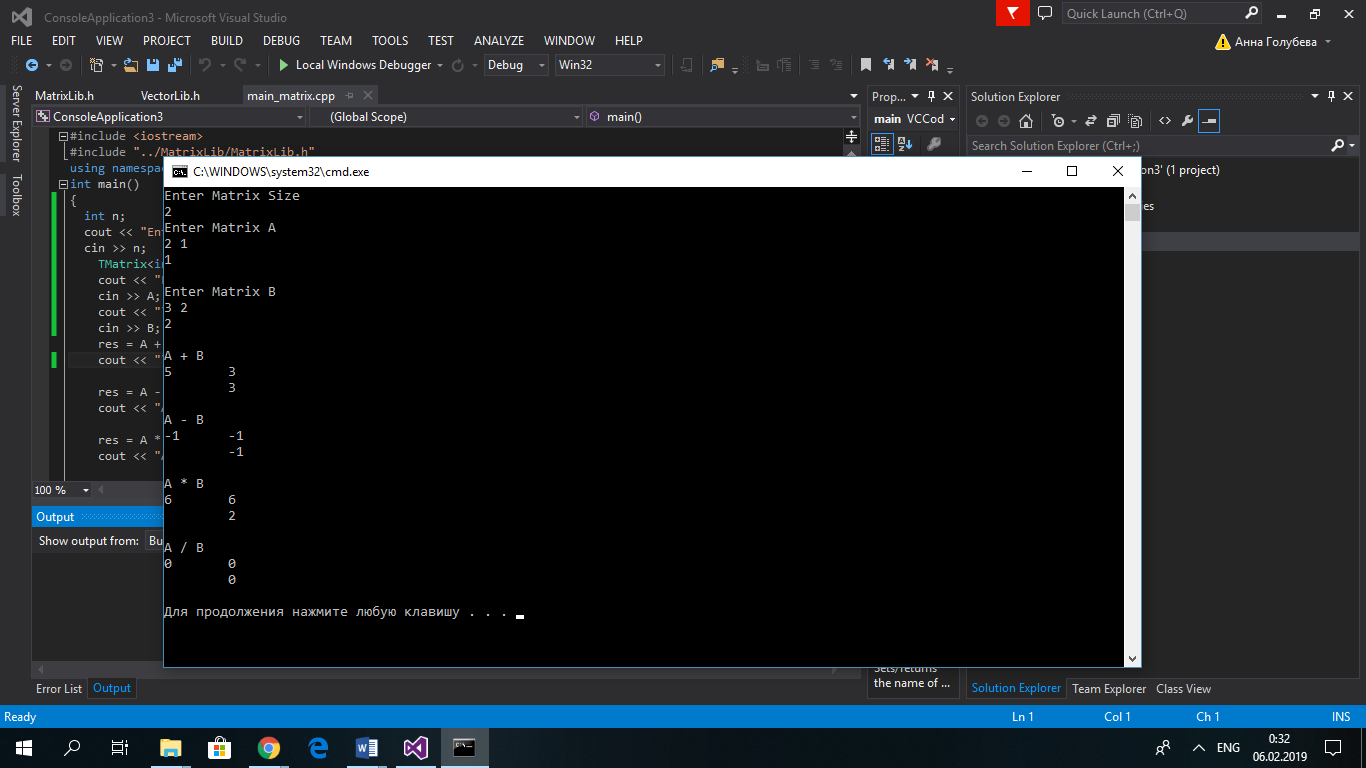


Рисунок 1. Пример работы программы

# **Руководство программиста**

## **4.1 Описание структуры программы**

Программа состоит из следующих модулей:

* Модуль VectorLib – содержит заголовочный файл VectorLib.h, в котором определяется интерфейс шаблонного класса вектор TVector и находится реализация методов этого класса.
* Модуль MatrixLib. Содержит файл MatrixLib.h, в котором определяется интерфейс шаблонного класса матрица TMatrix и находится реализация его методов.
* Модуль Vector – содержит пример использования класса TVector
* Модуль Matrix – содержит пример использования класса TMatrix.
* Модуль VectorTest – содерит файл test.cpp, в котором реализован набор тестов для класса TVector.
* Модуль MatrixTest – содержит файл test.cpp, в котором реализован набор тестов для класса TMatrix.
* Модуль Exception – содержит заголовочный файл Exception.h, в котором реализован класс исключений, и файл Exception.cpp.

## **4.2 Описание структур данных**

**Класс TVector**

Класс TVector – шаблонный класс.

**Protected:**

int size; - размер вектора

T \*Vector; - указатель на область памяти для хранения вектора

**Public:**

TVector<T>(int size = 0); - Конструктор инициализации, принимает размер вектора.

TVector<T>(const TVector <T> &v); - Конструктор копирования, принимает ссылку на объект класса TVector.

virtual ~TVector<T>(); - Деструктор.

int getSize(); - Метод, который возвращает размер вектора.

virtual T& operator[](int i); - Метод, возвращающий i-й элемент.

bool operator==(const TVector<T> &v) const; - Метод, принимающий ссылку на объект класса TVector и проверяющий два вектора на равенство.

bool operator!=(const TVector &v) const; - Метод, принимающий ссылку на объект класса TVector и проверяющий два вектора на неравенство.

TVector& operator= (const TVector<T> &v); - Метод, принимающий ссылку на объект класса TVector и приравнивающий исходный вектор к полученному.

TVector operator++(); - Инкремент.

TVector operator++(int);

TVector operator--(); -Декремент.

TVector operator--(int);

TVector operator+(const TVector<T> &v); - Принимает ссылку на объект класса TVector, возвращает сумму векторов.

TVector operator-(const TVector<T> &v); - Принимает ссылку на объект класса TVector, возвращает разность векторов.

T operator\*(const TVector<T> &v); - Принимает ссылку на объект класса TVector, возвращает

число, равное произведению векторов.

TVector operator\*(T h); - Возвращает вектор, равный произведению исходного вектора на число h.

Ввод-вывод:

template <class FriendT> friend istream& operator>>(istream &in,

TVector<FriendT> &v);

template <class FriendT> friend ostream& operator<<(ostream &out, const

TVector<FriendT> &v);

**Класс TMatrix**

Класс TMatrix- шаблонный класс, является наследником TVector.

**Public:**

TMatrix(int n = 10); - Конструктор инициализации, принимает размер матрицы.

TMatrix(const TMatrix &A); - Конструктор копирования, принимает ссылку на объект класса TMatrix.

TMatrix(const TVector<TVector<T> > &A);- Конструктор копирования, принимает ссылку на объект класса

bool operator==(const TMatrix &A);- Принимает ссылку на объект класса TMatrix и проверяющий две матрицы на равенство.

bool operator!=(const TMatrix &A); - Принимает ссылку на объект класса TMatrix и проверяющий две матрицы на неравенство.

TMatrix& operator= (TVector<TVector<T> > &A); - Принимает ссылку на объект класса TVector<TVector<T> > и приравнивающий исходный вектор к полученному.

TMatrix operator+ (TMatrix &A); - Принимает ссылку на объект класса TMatrix, возвращаетсумму матриц.

TMatrix operator- (TMatrix &A); - Принимает ссылку на объект класса TMatrix,, возвращает разность матриц.

TMatrix operator\*(TMatrix<T> &A);- Принимает ссылку на объект класса TMatrix, возвращает произведение матриц.

TMatrix operator/ (TMatrix<T> &A);- Принимает ссылку на объект класса TMatrix, возвращает результат деления исходной матрицы на полученную.

Ввод-вывод:

template <class FriendT> friend istream& operator>>(istream &istr, TMatrix<FriendT> &A);

template <class FriendT> friend ostream & operator<<(ostream &ostr, const TMatrix<FriendT> &A);

## **4.3 Описание алгоритмов**

Класс TMatrix является потомком класса TVector, поэтому для работы большинства методов требуется вызвать соответствующие методы из класса TVector.

Рассмотрим алгоритмы умножения и деления матриц.

**Умножение матриц**

Произведением матриц А и B является такая матрица C, у которой элемент , стоящий в *i*-ой строке и *j*-ом столбце, равен сумме произведений элементов *i*-ой строки матрицы А на соответствующие элементы *j*-го столбца матрицы B.

Перегрузка оператора умножения реализована с помощью трех циклов:

* по строкам первой матрицы;
* по столбцам второй матрицы;
* по элементам текущего столбца второй матрицы.

**Деление матриц**

Результатом деления матриц A и B является такая матрица C = A\*, где - обратная матрица к матрице B.

При нахождении обратной матрицы выполняются следующие шаги.

1. Записываем матрицу (обязательно квадратную) и дополняем (расширяем) ее справа единичной того же размера.
2. Приводим левую к специальному ступенчатому виду, выполняя элементарные преобразования над всей расширенной (включая правую).
3. В результате справа получим искомую обратную матрицу.

# **Заключение**

В результате лабораторной работы была разработана программа, поддерживающая эффективное хранение верхнетреугольных матриц и выполнение основных операций над ними.

Успешно выполнены тесты, проверяющие работоспособность методов классов TVector и TMatrix. Приведен пример, демонстрирующий работу с матрицами.

# **Литература**

1. Лабораторный практикум. Составители: Барышева И.В., Мееров И.Б., Сысоев А.В., Шестакова Н.В. Под редакцией Гергеля В.П. Учебно-методическое пособие. – Нижний Новгород: Нижегородский госуниверситет, 2017. – 105с.

URL: <http://www.unn.ru/books/met_files/Pract_ADS.pdf>

(Дата обращения: 06.02.2019)

1. Википедия: свободная электронная энциклопедия.

URL:<https://ru.wikipedia.org/wiki/%D0%A3%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86> (Дата обращения: 06.02.2019)