|  |  |  |
| --- | --- | --- |
|  | ***Федеральное агентство по рыболовству***  ***Федеральное государственное бюджетное образовательное учреждение***  ***высшего образования***  ***«Астраханский государственный технический университет»***  **Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована DQS**  **по международному стандарту ISO 9001:2015** | |
| Институт информационных технологий и коммуникаций  Направление подготовки 09.03.01 Информатика и вычислительная техника  Профиль «Автоматизированные системы обработки информации и управления»  Кафедра «Автоматизированные системы обработки информации и управления» | | |
| **КУРСОВОЙ ПРОЕКТ**  **Учебно-демонстрационная программа**  **«Решение системы линейных уравнений методом Гаусса»**  по дисциплине «Программирование и информатика» | | |
| Допущен к защите  «\_\_» \_\_\_\_\_\_\_\_20\_\_г.  Руководитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Оценка, полученная на защите «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_» | | Проект выполнил  обучающийся группы ДИНРб-11\2 Щербаков Е. В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Руководитель  ст. преп. Толасова В.В. |
| Члены комиссии:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев В.В.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Куркурин Н.Д.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Толасова В.В. | |  |

Астрахань – 2019

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ**

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | Кафедра «Автоматизированные системы  обработки информации и управления» |
| Заведующий кафедрой  к.т.н., доцент  С.В. Белов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_201 \_\_ г. |

**ЗАДАНИЕ**

**на выполнение курсового проекта**

Обучающийся ***Щербаков Егор Валериевич***

Группа ***ДИНРб-11\2***

Дисциплина ***Программирование и информатика***

Тема ***Учебно-демонстрационная программа «Решение системы линейных уравнений методом Гаусса»***

Дата получения задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_\_\_г.  
Срок представления обучающимся КП на кафедру «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_\_\_г.

Руководитель ***ст. преподаватель***\_\_\_\_\_\_\_\_\_ ***Толасова В.В.*** «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_\_\_г.

должность, степень, званиеподписьФИО

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***Щербаков Е.В.*** «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_\_\_г.

подпись ФИО

**Задачи**

Разработка программного продукта, который

* предоставляет пользователю учебный материал по заданной теме, записанный в файл.
* предоставляет демонстрационную часть.
* предоставляет пользователю тестирование, которое запрашивает ответы на 5/10/15, случайно выбранных из базы, теоретических вопросов.
* уведомляет пользователя о совершении ошибки сразу после ответа на задание в тестировании.
* сообщает результаты тестирования в виде зачёт/незачёт (без указания текста вопросов, в котором была сделана ошибка).
* хранит базу вопросов в файле.
* позволяет редактировать файлы, содержащие теоретический материал

**Список рекомендуемой литературы**

1 Эллайн А. C++ От ламера до программера – СПб.: Питер, 2015. – 480 с.

2 Булычева Ю.В., Васильева Т.В. Учебное пособие для студентов «Алгебра. Часть 2». – Астрахань, 2010. – 87 с. (Глава 5. Системы линейных уравнений).

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | К заданию на курсовой проектпо дисциплине  «Программирование и информатика» |
| Заведующий кафедрой  к.т.н., доцент  С.В. Белов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_г. |

**КАЛЕНДАРНЫЙ ГРАФИК**

курсового проектирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Разделы, темы и их содержание, графический материал | Дата сдачи | Объем, % |
| 1 | Выбор темы | 22.01.2019 | 1 |
| 2 | Техническое задание | 03.02.2019 | 3 |
| 3 | Разработка модели, проектирование системы   * *введение,* * *технический проект,* * *программа и методика испытаний,* * *литература* | 30.03.2019 | 25 |
| 4 | Программная реализация системы   * *работающая программа,* * *рабочий проект* * *скорректированное техническое задание (при необходимости)* | 30.04.2019 | 40 |
| 5 | Тестирование и отладка системы, эксперименты   * *работающая программа с внесёнными изменениями,* * *окончательные тексты всех разделов*   ***окончательные тексты***   * *введение* * *технический проект* * *рабочий проект* * *программа и методика испытаний* | 14.05.2019 | 50 |
| 6 | Компоновка текста  Подготовка презентации и доклада   * *пояснительная записка* * *презентация* * *электронный носитель с текстом пояснительной записки, исходным кодом проекта, презентацией и готовым программным продуктом* | 25.05.2019 | 59 |
| 7 | Защита курсового проекта | 28.05.2019-01.06.2019 | 60-100 |

С графиком ознакомлен «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_г.

Щербаков Е.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, обучающийся группы ДИНРб-11

(фамилия, инициалы, подпись)

График курсового проектирования выполнен

без отклонений / с незначительными отклонениями / со значительными отклонениями

нужное подчеркнуть

Руководитель курсового проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преподаватель Толасова В.В.

подпись, ученая степень, звание, фамилия, инициалы

**СОДЕРЖАНИЕ**

[Введение 5](#_Toc26274554)

[1 Технический проект 6](#_Toc26274555)

[1.1 Анализ предметной области 6](#_Toc26274556)

[1.1.1 Понятие системы линейных алгебраических уравнений 6](#_Toc26274557)

[1.1.2 Метод Гаусса 6](#_Toc26274558)

[1.1.3 Проверка знаний 7](#_Toc26274559)

[1.1.4 Административные функции 8](#_Toc26274560)

[1.2 Технология обработки информации 8](#_Toc26274561)

[1.2.1 Формат данных 10](#_Toc26274562)

[1.2.2 Алгоритм считывания теории 11](#_Toc26274563)

[1.2.3 Алгоритм вывода теории 11](#_Toc26274564)

[1.2.4 Алгоритм считывания примеров 12](#_Toc26274565)

[1.2.5 Алгоритм отображения одной матрицы 12](#_Toc26274566)

[1.2.6 Алгоритм вывода примеров 13](#_Toc26274567)

[1.2.7 Алгоритм вывода вопроса и ответов 13](#_Toc26274568)

[1.2.8 Алгоритм заполнения вектора с вопросами 15](#_Toc26274569)

[1.2.9 Алгоритм тестирования 15](#_Toc26274570)

[1.3 Входные данные 16](#_Toc26274571)

[1.4 Системные требования 16](#_Toc26274572)

[2 Рабочий проект 17](#_Toc26274573)

[2.1 Общие сведения о работе системы 17](#_Toc26274574)

[2.2 Функциональное назначение программного продукта 17](#_Toc26274575)

[2.3 Инсталляция и выполнение программного продукта 17](#_Toc26274576)

[2.4 Описание программы 18](#_Toc26274577)

[2.5 Разработанные меню и интерфейсы 25](#_Toc26274578)

[2.6 Сообщения системы 30](#_Toc26274579)

[Программа и методика испытаний 31](#_Toc26274580)

[Заключение 34](#_Toc26274581)

[Список использованных источников 35](#_Toc26274582)

[Приложение 1 Техническое задание 36](#_Toc26274583)

[Приложение 2 База вопросов программы 39](#_Toc26274584)

# ВВЕДЕНИЕ

Во время обучения на технических специальностях студентам приходится сталкиваться с системами линейных уравнений. У систем линейных уравнений есть множество различных методов решения, одним из которых является метод Гаусса.

Поскольку линейные уравнения являются важной темой в алгебре и не только, крайне важно уметь решать их. Однако не всем студентам удаётся в полной мере освоить линейные уравнения, поскольку не весь доступный студентам материал по этой теме бывает полезен или, по крайней мере, не усугубляет ситуацию со знаниями студента.

Так же сложно найти и готовые тесты, которые помогут студентам в освоении данной темы и отточат их навыки, поэтому создание программы, совмещающей в себе эти два качества является важным в наше время.

Целью создания учебно-демонстрационной программы «Решение систем линейных уравнений методом Гаусса» является автоматизация процесса обучения студентов по указанной теме.

Назначение программы – повышение успеваемости и качества знаний студентов, снижение нагрузки на преподавателя.

# 1 ТЕХНИЧЕСКИЙ ПРОЕКТ

## Анализ предметной области

### Понятие системы линейных алгебраических уравнений

Система линейных алгебраических уравнений — система уравнений, каждое уравнение в которой является линейным — алгебраическим уравнением первой степени.

Общий вид системы линейных алгебраических уравнений (формула 1.1):

|  |  |
| --- | --- |
| E:\Папка_Егора\Bandicam\bandicam 2019-11-21 16-28-28-651 - копия.jpg | (1.1) |

Система линейных алгебраических уравнений может быть представлена в матричной форме как (формула 1.2):

|  |  |
| --- | --- |
| E:\Папка_Егора\Bandicam\bandicam 2019-11-21 16-30-33-209 - копия.jpg | (1.2) |

Число aij – элемент матрицы, расположенной на пересечении i-той строки и j-того столбца.

### Метод Гаусса

Ме́тод Га́усса — классический метод решения системы линейных алгебраических уравнений. Это метод последовательного исключения переменных, когда с помощью элементарных преобразований система уравнений приводится к равносильной системе треугольного вида, из которой последовательно, начиная с последних (по номеру), находятся все переменные системы.

|  |  |
| --- | --- |
| E:\Папка_Егора\Bandicam\bandicam 2019-11-21 16-39-18-817.jpg | (1.3) |

E:\Папка_Егора\Bandicam\bandicam 2019-11-21 16-41-30-133.jpg

Если хотя бы одно число E:\Папка_Егора\Bandicam\bandicam 2019-11-21 16-45-19-234 - копия.jpg где *i* > *r*, то рассматриваемая система несовместна, т.е. у неё нет ни одного решения.

Перенесём свободные переменные за знаки равенств и поделим каждое из уравнений системы на свой коэффициент при самом левом неизвестном *(E:\Папка_Егора\Bandicam\bandicam 2019-11-21 17-04-02-089 - копия.jpg* , *i=1*,…,*r*, где *i* — номер строки) (формула 1.4):

|  |  |
| --- | --- |
| C:\Users\Egor\Desktop\bandicam 2019-11-21 17-05-55-718 - копия.jpg | (1.4) |
| C:\Users\Egor\Desktop\bandicam 2019-11-21 17-05-55-718.jpg |  |

где *i* = 1,…,*r*, *k* = *i* + 1,…,*n*.

Если свободным переменным системы (1.4) придавать все возможные значения и решать новую систему относительно главных неизвестных снизу-вверх (то есть от нижнего уравнения к верхнему), то мы получим все решения этой системы. Так как эта система получена путём элементарных преобразований над исходной системой (1.3), то по теореме об эквивалентности при элементарных преобразованиях системы (1.3) и (1.4) эквивалентны, то есть множества их решений совпадают.

### Проверка знаний

Студент начинает изучение темы «Решение систем линейного уравнения методом Гаусса» с того, что получает теоретический материал по теме и рассматривает решение учебных примеров. Для проверки знаний темы пользователю предлагается пройти тестирование, которое имеет уровень сложности (выбираемый пользователем и определяющий количество вопросов).

Программа запрашивает ответ на вопрос у пользователя и сравнивает его с эталонным. Если пользователь ответил правильно, то счётчик тестирования увеличивается на единицу, если неправильный, то выводится правильный ответ и счётчик тестирования уменьшается на единицу.

Тестирование будет продолжаться, пока пользователь не ответит на все вопросы или не допустит больше 50% ошибок. По итогу на экран выведется счёт, полученный в результате прохождения теста. Таким образом, пользователь будет знать примерный уровень своего понимания данной темы.

### Административные функции

Для сохранения корректной работоспособности программы, все теоретические данные зашифрованы в бинарных файлах. Поскольку такие файлы невозможно редактировать стандартными средствами операционной системы, в программе должен быть предусмотрен административный режим, защищённый паролем. После корректного ввода пароля (можно считать три попытки ввода достаточным количеством) пользователь должен иметь следующие возможности: изменить читаемый файл для указанной темы (что позволяет проверить читабельность файла внутри программы); внести корректировки в выбранный файл; перезаписать внутри программы вектор с данными, если файл был изменён вне программы (в случае, если оригинальный файл был заменён другим файлом с тем же именем); посмотреть всю информацию о прочитанном файле без полноценного чтения данного файла (для всех файлов: имя файла; для файлов с текстом: количество листов и строк в каждом листе; для файлов с вопросами: количество вопросов в файле; для файлов с матрицами примеров: количество примеров, количество матриц в каждом примере); полностью перезаписать выбранный файл (новые данные могут быть введены вручную или скопированы с .txt файла); а также сменить пароль для входа в административный режим.

## Технология обработки информации

Анализ предметной области показал, что программа рассчитана на одного пользователя.

На рисунке 1.1 показана диаграмма вариантов использования для обычного пользователя.

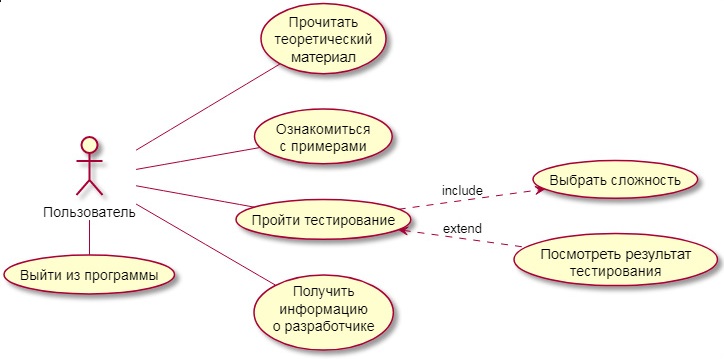


Рисунок 1.1 – Диаграмма вариантов использования для обычного пользователя

Пользователь должен иметь возможность получить учебный материал, пройти тестирование для проверки знаний, наглядный пример вычисления, дать ответ на задание, а также получить результат тестирования.

На рисунке 1.2 показана диаграмма вариантов использования для администратора.

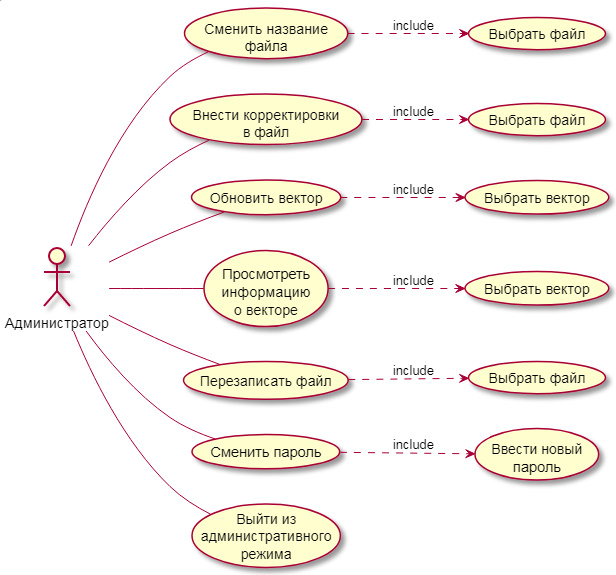


Рисунок 1.2 – Диаграмма вариантов использования для администратора

Администратор должен иметь возможность изменить файл для чтения, внести изменения в любой файл, используемый в программе, перезаписать любой вектор внутри программы, посмотреть информацию о уже прочитанном файле (в формате вектора), полностью перезаписать любой файл, используемый в программе, поменять пароль для входа в административный режим, а так же выйти из данного режима, чтобы продолжить работу в качестве пользователя.

### Формат данных

Учебный материал содержится в бинарных файлах.

Бинарный файл с базой вопросов заполнен структурами типа:

* Вопрос
* Первый вариант ответа (в файле первый ответ всегда правильный)
* Второй вариант ответа
* Третий вариант ответа

Бинарный файл с матрицами примеров заполнен структурами типа:

* Матрица размером 3х4
* Номер строки, в которой находится подсвечиваемый (изменяемый) элемент
* Номер столбца, в котором находится подсвечиваемый (изменяемый) элемент

### Алгоритм считывания теории

Считывание учебного материала происходит следующим образом:

Объявить переменную **main\_vector**: вектор векторов, состоящих из строк.

Объявить переменную **single\_list**: вектор строк.

Открыть на чтение файл “Theory.bin/Examples.bin”.

Прочитать из файла переменную **temp\_string**: строка символов.

Пока **temp\_string[temp\_string.size() - 1] != '#'** делать:

| Если **temp\_string[0] == '\t'**, то:

| | Добавить вектор **single\_list** справа к вектору **main\_vector**.

| | Очистить вектор **single\_list**.

| Добавить строку **temp\_string** справа к вектору **single\_list**.

| Прочитать из файла переменную **temp\_string**.

Удалить из строки **temp\_string** предпоследний элемент.

Добавить символ **‘\n’** справа к строке **temp\_string**.

Добавить строку **temp\_string** справа к вектору **single\_list**.

Добавить вектор **single\_list** справа к вектору **main\_vector**.

Закрыть файл “Theory.bin/Examples.bin”.

Конец цикла.

### Алгоритм вывода теории

Вывод страниц учебного материала происходит следующим образом:

Прочитать теорию с помощью алгоритма 1.2.2 в вектор векторов **main\_vector**.

Присвоить **i = 0**.

Инициализировать переменную **true\_size** = **main\_vector.size()**.

Объявить переменную **user\_s\_choice**: символ.

Пока **user\_s\_choice != 'e'** делать:

| Присвоить **counter = 0**.

| Пока **counter != main\_vector[i].size()**:

| | Отобразить **main\_vector[i][counter]**.

| | **counter**++.

| Конец цикла.

| Запросить у пользователя ввод переменной **user\_s\_choice**.

| Если **user\_s\_choice** = ‘b’ и **i** != 0: Уменьшить **i** на 1.

| Иначе если **user\_s\_choice** = ‘f’ и **i** != **true\_size**: Увеличить **i** на 1.

| Иначе если **user\_s\_choice** = ‘e’: Конец цикла.

### Алгоритм считывания примеров

Считывание теории для примеров происходит следующим образом:

Прочитать текстовый теоретический материал для примеров с помощью алгоритма 1.2.2 в переменную **main\_vector**.

Объявить переменную **examples\_vector**: вектор векторов матриц 3х4.

Открыть на чтение файл “Numb\_ex.bin”.

Прочитать из файла переменную **quantity\_of\_examples**.

Присвоить **i = 0**.

Пока **i < quantity\_of\_examples** делать:

| Проитать из файла переменную **quantity\_of\_matrix**.

| Присвоить **j = 0**.

| Пока **j** < **quantity\_of\_matrix** делать:

| | Считать **examples\_vector[i][j]**;

| | **j**++;

| Конец цикла.

Конец цикла.

Закрыть файл “Numb\_ex.bin”.

### Алгоритм отображения одной матрицы

Дано: matrix – матрица размером 3х4, row – номер строки с выделяемым элементом, col – номер столбца с выделяемым элементом.

Присвоить **i = 0**;

Пока **i** < 3 делать:

| Присвоить **j = 0**.

| | Пока **j** < 4 делать:

| | Если **i = row** и **j = col**:

| | | Установить зелёный цвет текста.

| | | Вывести **matrix[i][j]**.

| | | Установить белый цвет текста.

| | Иначе:

| | | Вывести **matrix[i][j]**.

| Конец цикла.

Конец цикла.

### Алгоритм вывода примеров

Вывод примеров для демонстрации происходит следующим образом:

Считать теорию с помощью алгоритма 1.2.4.

Присвоить **i = 0**.

Пока **i** < **main\_vector.size()** делать:

| Присвоить **counter = 0**.

| Пока **counter != main\_vector[i].size()**:

| | Отобразить **main\_vector[i][counter]**.

| | **counter**++.

| Конец цикла.

| **i**++;

| Присвоить **j = 0**.

| Пока **j < examples\_vector[i].size()**:

| | Вызвать алгоритм 1.2.5, передав в качестве параметра **examples\_vector[i][j]**.

| | **j++**;

| Конец цикла.

| Присвоить **counter = 0**.

| Пока **counter != main\_vector[i].size()**:

| | Отобразить **main\_vector[i][counter]**.

| | **counter**++.

| Конец цикла.

| **i**++;

Конец цикла.

### Алгоритм вывода вопроса и ответов

Вывод вопроса с ответами для тестирования происходит следующим образом:

Дано: четыре строки — **Question**, **answer\_one**, **answer\_two**, **answer\_three**.

Инициализировать переменную **number** случайным числом из диапазона **<0, 2>**.

Вывести **Question**.

Eсли **number = 0**:

| Вывести **answer\_one**.

| Инициализировать переменную **number** случайным числом из диапазона **<0, 1>**.

| Если **number = 0**:

| | Вывести **answer\_two**.

| | Вывести **answer\_three**.

| Иначе:

| | Вывести **answer\_three**.

| | Вывести **answer\_two**.

| Конец если.

Иначе если **number** = 1:

| Вывести **answer\_two**.

| Инициализировать переменную **number** случайным числом из диапазона **<0, 1>**.

| Если **number = 0**:

| | Вывести **answer\_one**.

| | Вывести **answer\_three**.

| Иначе:

| | Вывести **answer\_three**.

| | Вывести **answer\_one**.

| Конец если.

Иначе:

| Вывести **answer\_three**.

| Инициализировать переменную **number** случайным числом из диапазона **<0, 1>**.

| Если **number = 0**:

| | Вывести **answer\_one**.

| | Вывести **answer\_two**.

| Иначе:

| | Вывести **answer\_two**.

| | Вывести **answer\_one**.

| Конец если.

Конец если.

### Алгоритм заполнения вектора с вопросами

Заполнение вектора с вопросами происходит следующим образом:

Дано: вектор **main\_vector**, каждая ячейка которого содержит четыре строки; количество вопросов **difficulty**.

Присвоить **i = 0**.

| Инициализировать переменную **number** случайным числом из диапазона **<0,**

**| main\_vector.size() - 1>**.

| Добавить элемент **main\_vector[number]** в вектор **question\_vector**.

| Удалить элемент по индексу **number** из вектора **main\_vector**.

Конец цикла.

Вернуть **question\_vector**.

### Алгоритм тестирования

Тестирование происходит следующим образом:

Дано: вектор **main\_vector**, каждая ячейка которого содержит четыре строки.

Объявить вектор **question\_vector**.

Запросить у пользователя ввод переменной **input\_dif** (символ).

Если **input\_dif = ‘l’**, то: размер вектора **question\_vector** будет определён как 5, переменная **max\_false** как 3.

Иначе если **input\_dif = ‘m’**, то: размер вектора **question\_vector** будет определён как 10, переменная **max\_false** как 5.

Иначе: размер **question\_vector** вектора будет определён как 15, переменная **max\_false**

как 8.

Заполнить вектор **question\_vector** случайными вопросами из заранее подготовленных алгоритмом 1.2.8.

Присвоить **i = 0**.

Пока **i < question\_vector.size()** и **false\_counter < max\_false**:

| Отобразить вопрос с помощью алгоритма 1.2.7.

| Считать ответ пользователя.

| Если ответ пользователя верен, то к счётчику **true\_counter** прибавляется 1.

| Иначе к счётчику **false\_counter** прибавляется 1.

Конец цикла.

## Входные данные

Входные данные:

* выбор пункта в главном меню.
* ответ пользователя на вопросы в тестировании.
* бинарный файл, содержащий текст – учебный материал.
* бинарный файл, содержащий текст – вопросы к тестированию.
* бинарный файл, содержащий матрицы 3х4 – примеры.

## Системные требования

* Intel - совместимый процессор с частотой не менее 1,6 ГГц;
* не менее 512 МБ ОЗУ;
* не менее 20 MБ свободного места на диске;
* дисковод CD-ROM/DVD-ROM, монитор, клавиатура, мышь.

Операционная система: Windows XP(x86) с пакетом обновления 3(SP3). Среда

разработки – CodeBlocks 17.12.

# 2 РАБОЧИЙ ПРОЕКТ

## 2.1 Общие сведения о работе системы

Программный продукт разработан в интегрированной среде CodeBlocks 17.12 на языке C++. Программа работает под управлением операционной системы Windows XP (x86) Professional (SP3) и более поздними. Стартовый модуль для CodeBlocks – Course\_project.cbp.

## 2.2 Функциональное назначение программного продукта

Разработанный программный продукт предназначен для отработки навыков решения линейных уравнений с помощью метода Гаусса. Программа имеет следующие функциональные возможности:

* предоставляет пользователю учебный материал по заданной теме, записанный в файл;
* предоставляет демонстрационную часть;
* предоставляет пользователю тестирование, которое запрашивает ответы на 5/10/15, случайно выбранных из базы, теоретических вопроса;
* уведомляет пользователя о совершении ошибки сразу после ответа на задание в тестировании;
* сообщает результаты тестирования в виде зачёт/незачёт (без указания текста вопроса, в котором была сделана ошибка);
* хранит базу вопросов в файле;
* позволяет редактировать теоретические данные с помощью административных функций.

Программа имеет следующие функциональные ограничения:

* теоретический материал может быть записан только в бинарные файлы;
* в конце файла с теоретическим текстовым материалом должен быть символ “#”;
* матрицы для решения примеров не могут быть иного размера, чем 3х4.

## 2.3 Инсталляция и выполнение программного продукта

Для выполнения программы необходимо:

1. Скопировать на жесткий диск компьютера папку Release, содержащую приложение COURSE\_PROJECT.exe и бинарные файлы: Theory.bin, Tests.bin, Examples.bin, Numb\_ex.bin, Password.bin.
2. Запустить на выполнение COURSE\_PROJECT.exe.

## 2.4 Описание программы

Программа состоит из 7 заголовочных файлов (табл. 2.1).

Таблица 2.1 – Список заголовочных файлов программы

|  |  |
| --- | --- |
| **Заголовочный файл** | **Назначение** |
| menus.h | Содержит функции для вывода главного меню и меню с информацией о разработчиках |
| Theory.h | Содержит функции для работы с теоретическим материалом |
| Examples.h | Содержит функции для работы с материалом для примеров |
| Tests.h | Содержит функции для работы с тестами |
| administrative.h | Содержит административные функции |
| Colors.h | Содержит функции для изменения цвета текста в консоли |
| cin\_fixer.h | Содержит вспомогательные функции для предотвращения некорректного ввода |

Ниже представлены функции, используемые в выполнении этих модулей:

В таблице 2.2 приведено описание модуля menus.cpp используемого в программе.

Таблица 2.2 – Функции модуля menus.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void border\_line(const int length) | Вывод разделительной черты |
| void main\_menu(short unsigned int& MM) | Отображение главного меню |
| void about\_developers() | Вывод информации о разработчике |

В таблице 2.3 приведено описание модуля Theory.cpp используемого в программе.

Таблица 2.3 – Функции модуля Theory.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void read\_one\_line(ifstream& reading\_file, string& temp\_string) | Считывание одной строки |
| bool read\_one\_list(ifstream& reading\_file, single\_list& temp\_vector) | Считывание одного листа |
| bool read\_theory(const string& name\_of\_file, all\_theory& main\_vector) | Считывание всей теории из файла |

Продолжение таблицы 2.3

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| bool correct\_input\_ch(const char& choice, const unsigned int& i, const unsigned int& true\_size) | Проверка на корректный ввод пользователем символа при чтении теории |
| void display\_theory(const all\_theory& main\_vector) | Отображение теории |

В таблице 2.4 приведено описание модуля Examples.cpp используемого в программе.

Таблица 2.4 – Функции модуля Examples.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void display\_matrix(const disp\_mtr& matr) | Отображение матрицы |
| single\_example read\_one\_example(ifstream& read\_file) | Считывание одного примера |
| bool read\_examples(const string& name\_of\_file, all\_examples& ex\_vec) | Считывание всех примеров из файла |
| void display\_single\_list(const single\_list& list\_of\_theory) | Отображение одного листа с теорией |
| void display\_single\_example(const single\_example& exmp) | Отображение одного примера |
| void display\_examples(const all\_theory& main\_vector, const all\_examples& ex\_vec) | Отображение всех примеров |

В таблице 2.5 приведено описание модуля Tests.cpp используемого в программе.

Таблица 2.5 – Функции модуля Tests.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| bool read\_tests(tests& main\_vector, const string& file\_name) | Считывание тестов с файла |
| tests create\_test\_questing(tests main\_vector, const short unsigned int difficulty) | Создание временного вектора для тестирования |
| short unsigned int display\_question(const Quest& question) | Отображение вопроса со случайно перемешанными ответами |
| bool testing\_cycle(const char difficulty, const tests& question\_vector) | Цикл работы с заданиями |
| void start\_testing(const tests& main\_vector, const bool& is\_theory\_vec\_filled) | Основная функция запуска тестов |

В таблице 2.6 приведено описание модуля administrative.cpp используемого в программе.

Таблица 2.6 – Функции модуля administrative.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| bool check\_correct\_password\_input(const string& real\_pass, const string& input\_pass) | Проверка введённого пароля |
| bool start\_administrative() | Включение административного режима |
| bool check\_correct\_bin\_file\_name\_input(const string str) | Проверка правильности ввода имени бинарного файла |
| string change\_name\_of\_file(const string file\_name) | Смена имени файла |
| short unsigned int administrative\_menu() | Административное меню программы (отображение самого меню) |
| bool change\_bin(const string &name) | Перезапись в бинарный файл пароля |
| void change\_password() | Смена пароля для входа в административный режим |
| char get\_correct\_symbol\_for\_p5() | Проверка правильности ввода символа для пунктов 5,7 административного меню |
| char submenu\_for\_changing\_th\_vector(const bool& is\_examples) | Подменю для пунктов 5,7 административного меню |
| void out\_single\_list(all\_theory& main\_vector, const unsigned int& number\_of\_list) | Отображение одного листа с теорией для пунктов 5,7 административного меню |
| void change\_single\_string(all\_theory& main\_vector, const unsigned int& number\_of\_list, const unsigned int& number\_of\_string) | Редактирование выбранной строки для пунктов 5,7 административного меню |
| void start\_correcting\_string(all\_theory& main\_vector, const unsigned int& number\_of\_list) | Ввод номера строки пользователем и её изменение для пунктов 5,7 административного меню |
| void change\_single\_list(all\_theory& main\_vector) | Корректировка одного листа для пунктов 5,7 административного меню |
| char get\_choice\_for\_p5(const unsigned int& i, const unsigned int& true\_size) | Выбор пользователя для движения по листу для пунктов 5,7 административного меню |
| void change\_all\_theory(all\_theory& main\_vector) | Чтение всей теории для пунктов 5,7 административного меню |

Продолжение таблицы 2.6

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| bool rewrite\_bin\_theory\_file(const all\_theory& main\_vector, const string& name\_of\_file) | Перезапись файла с теорией для пунктов 5,7 административного меню |
| void change\_theory\_vector(all\_theory& main\_vector, const string& name\_of\_file, const bool& is\_examples) | Редактирование вектора с теорией для пунктов 5,7 административного меню |
| char get\_correct\_symbol\_for\_p6() | Проверка правильности ввода символа для пункта 6 административного меню |
| char submenu\_for\_changing\_tests\_vector() | Подменю для пункта 6 административного меню |
| void out\_single\_question(tests& main\_vector, const unsigned int& number\_of\_question) | Отображение одного вопроса для пункта 6 административного меню |
| string change\_single\_string(const string original\_str) | Редактирование выбранной строки для пункта 6 административного меню |
| short unsigned int get\_correct\_choice\_for\_p6() | Получение корректного ответа пользователя для пункта 6 административного меню |
| void choice\_for\_changing(tests& main\_vector, const unsigned int& number\_of\_question) | Выбор пункта для корректировки для пункта 6 административного меню |
| void change\_single\_question(tests& main\_vector) | Редактирование одного вопроса для пункта 6 административного меню |
| void change\_all\_tests(tests& main\_vector) | Отображение и редактирование всех вопросов для пункта 6 административного меню |
| bool rewrite\_bin\_tests\_file(const tests& main\_vector, const string& name\_of\_file) | Перезапись файла с тестами для пункта 6 административного меню |
| void change\_tests\_vector(tests& main\_vector, const string& name\_of\_file) | Редактирование вектора с тестами для пункта 6 административного меню |
| char get\_correct\_symbol\_for\_p8() | Проверка правильности ввода символа для пункта 8 административного меню |
| char submenu\_for\_changing\_numb\_ex\_vector() | Подменю для пункта 8 административного меню |

Продолжение таблицы 2.6

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void change\_glowing\_element(all\_examples& main\_vector, const unsigned int& number\_of\_example, const unsigned int& number\_of\_matrix) | Смена подсвечиваемого элемента для пункта 8 административного меню |
| void manual\_matrix\_input(all\_examples& main\_vector, const unsigned int& number\_of\_example, const unsigned int& number\_of\_matrix) | Ручной ввод одной матрицы для пункта 8 административного меню |
| void change\_single\_matrix\_element(all\_examples& main\_vector, const unsigned int& number\_of\_example, const unsigned int& number\_of\_matrix) | Смена выбранного элемента в матрице для пункта 8 административного меню |
| short unsigned int submenu\_for\_changing\_single\_matrix() | Подменю для редактирования одной матрицы для пункта 8 административного меню |
| void change\_single\_matrix(all\_examples& main\_vector, const unsigned int& number\_of\_example, const unsigned int& number\_of\_matrix) | Редактирование одной матрицы для пункта 8 административного меню |
| void choose\_matrix\_to\_change(all\_examples& main\_vector) | Выбор матрицы для редактирования для пункта 8 административного меню |
| void change\_single\_matrix\_for\_cycle(all\_examples& main\_vector, const unsigned int& number\_of\_example, const unsigned int& number\_of\_matrix) | Редактирование одной матрицы (функция для цикла) для пункта 8 административного меню |
| bool rewrite\_bin\_examples\_file(const all\_examples& main\_vector, const string& name\_of\_file) | Перезапись файла с матрицами примеров для пункта 8 административного меню |
| void change\_all\_matrix(all\_examples& main\_vector) | Чтение и редактирование всех матриц для пункта 8 административного меню |
| void change\_nu\_ex\_vector(all\_examples& main\_vector, const string& name\_of\_file) | Редактирование вектора с матрицами примеров для пункта 8 административного меню |
| void clearing\_vector(all\_theory& main\_vector, string& name\_of\_file, bool& is\_vec\_filled) | Сброс вектора с теорией и заполнение его заново (п.9,11) |

Продолжение таблицы 2.6

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void clearing\_vector(tests& main\_vector, string& name\_of\_file, bool& is\_vec\_filled) | Сброс вектора с тестами и заполнение его заново (п.10) |
| void clearing\_vector(all\_examples& main\_vector, string& name\_of\_file, bool& is\_vec\_filled) | Сброс вектора с матрицами примеров и заполнение его заново (п.12) |
| void see\_info(const all\_theory& main\_vector, const string& name\_of\_file, const bool& is\_vec\_filled) | Отображение данных по вектору с теорией (п.13) |
| void see\_info(const tests& main\_vector, const string& name\_of\_file, const bool& is\_vec\_filled) | Отображение данных по вектору с тестами (п.14) |
| void see\_info(const all\_examples& main\_vector, const string& name\_of\_file, const bool& is\_vec\_filled) | Отображение данных по примерам (п.15) |
| bool check\_correct\_txt\_file\_name\_input(const string& str) | Проверка правильности ввода имени текстового файла |
| string get\_correct\_txt\_file\_name() | Получение корректного имени .txt файла |
| string get\_correct\_bin\_file\_name() | Получение корректного имени .bin файла |
| bool check\_correct\_opening(string& file\_name, ifstream& file\_stream) | Проверка корректности открытия файла с заданным именем (входной файловый поток, .txt файл) |
| bool check\_correct\_opening(string& file\_name, ofstream& file\_stream) | Проверка корректности открытия файла с заданным именем (выходной файловый поток, .bin файл) |
| void cycle\_of\_rewriting(const string& bin\_file\_name, ifstream& in\_file) | Цикл перезаписи из .txt файла в .bin файл |
| void cycle\_of\_rewriting(ifstream& in\_file, ofstream& out\_file) | Цикл перезаписи из .txt в .bin (новый выходной файл) |
| void rewrite\_txt\_to\_bin(string& name\_of\_theory) | Перезапись .txt файла в .bin файл |
| string cin\_file\_name() | Ввод имени бинарного файла |
| void txt\_to\_bin\_str\_cycle(ifstream& in\_file, ofstream& out\_file) | Цикл перезаписи из .txt в .bin для файла со структурами вопросов |
| void from\_txt\_to\_bin\_struct(string& bin\_file\_name) | Перезапись .txt файла в .bin файл в структурном формате |

Продолжение таблицы 2.6

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| Quest get\_struct\_var() | Ввод одного вопроса с ответами |
| void manual\_input\_in\_bin\_struct(string& file\_name) | Запись .bin файла вручную в структурном формате |
| void write\_struct\_bin(const string& name\_of\_tests) | Запись .bin файла со структурами вопросов |
| bool error\_of\_opening(const string& name\_of\_file) | Сообщение об ошибке открытия |
| disp\_mtr get\_matrix\_from\_txt(ifstream& in\_file) | Получение одной матрицы из .txt файла |
| void from\_txt\_to\_matrix\_file(const string& name\_of\_bin\_file) | Перезапись txt-bin (матрицы) |
| disp\_mtr get\_matrix\_manual() | Прямой ввод матрицы |
| void manual\_matrix\_input\_for\_bin(const string& name\_of\_bin\_file) | Прямой ввод матриц |
| void write\_matrix\_bin(const string& name\_of\_num\_ex) | Запись .bin файла с матрицами |
| void administrative\_menu\_main(all\_theory& main\_theory\_vector, tests& main\_tests\_vector, all\_theory& main\_examples\_vector, all\_examples& num\_examples\_vector, string& name\_of\_theory, string& name\_of\_tests, string& name\_of\_th\_ex, string& name\_of\_nu\_ex, bool& admin\_mode, bool& is\_theory\_vec\_filled, bool& is\_tests\_vec\_filled, bool& is\_examples\_vector\_filled, bool& is\_numb\_ex\_vector\_filled) | Административное меню (код работы) |

В таблице 2.7 приведено описание модуля cin\_fixer.cpp используемого в программе.

Таблица 2.7 – Функции модуля cin\_fixer.cpp

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| bool fix\_cin() | Защита от некорректного ввода |
| char cin\_choice() | Получение корректного символа (y/n) |
| char cin\_choice\_ex() | Получение корректного символа (y/n/e) |

Файл Colors.cpp содержит одну функцию:

void SetColor(int text, int background) – устанавливает цвет текста и консоли.

## 2.5 Разработанные меню и интерфейсы

После запуска программы на выполнение в консольном окне появится сообщение о запуске (рис.2.1), которое позволяет нажать на любую клавишу и войти в меню.

C:\Users\Egor\Desktop\Новая папка (2)\bandicam 2019-11-23 10-04-01-733.jpg

Рисунок 2.1 – Титульный лист для входа в меню

После нажатия, в консольном окне появится главное меню (рис.2.2), которое позволяет выбрать одно из шести дальнейших действий: переход к изучению теоретической части, переход к просмотру демонстрации практической части, переход к выполнению теста, доступ к административному режиму, отображение информации о разработчике и выход из программы. Пример приведён на рис. 2.2.

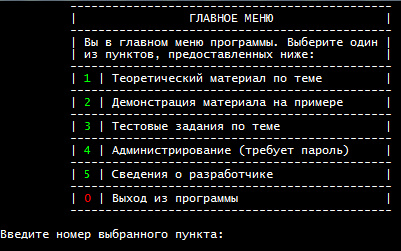


Рисунок 2.2 – Главное меню

При выборе первого пункта пользователь перейдет к просмотру теоретического материала. Также программа выведет запрос на дальнейшие действия. Выход в главное меню доступен пользователю в любой момент, как и пролистывание назад.

Пример вывода теории предоставлен на рис. 2.3.

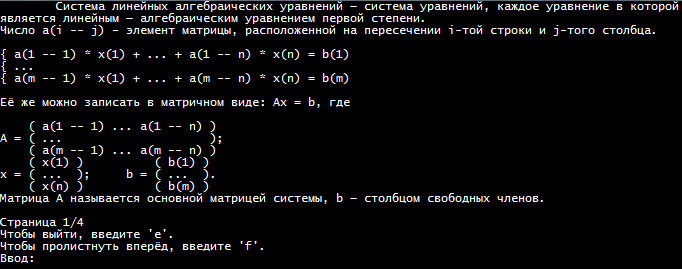


Рисунок 2.3 – Консоль программы с учебным материалом и выбором дальнейших действий

При выборе второго пункта в меню пользователь перейдёт к демонстрации практической части, где он сможет получить знания по решение систем линейных уравнений методом Гаусса. Пример представлен на рис. 2.4.

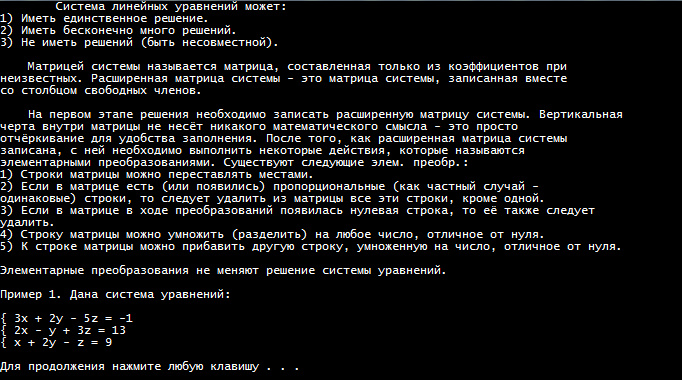


Рисунок 2.4 – Демонстрация практической части

После того, как пользователь ознакомится с предоставленной теоретической информацией и примером, программа отобразит решение системы в виде матриц, подсвечивая при этом изменённый элемент. Пользователь в любой момент может остановить чтение примера. Пример представлен на рис. 2.5.

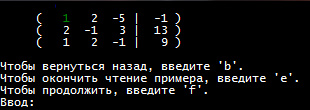


Рисунок 2.5 – Консоль программы с демонстрационным материалом

Выбрав третий пункт в главном меню, пользователь может пройти тестирование. Пользователь получит 5/10/15 теоретических заданий на знание учебного материала, сложность выбирает он сам. Пример представлен на рис. 2.6:

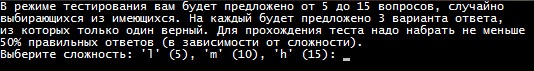


Рисунок 2.6 – Начало тестирования

Если пользователь ошибется при решении, будет выведено сообщение об ошибке и счётчик правильных/неправильных ответов. Пример представлен на рис. 2.7.

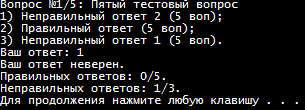


Рисунок 2.7 – Сообщение об ошибке пользователя

Если же пользователь ответит верно, то ему будет сообщено об этом и программа выведет счётчик правильных/неправильных ответов. Пример на рис. 2.8:

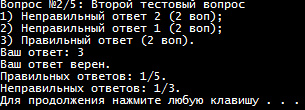


Рисунок 2.8 – Сообщение о правильном ответе пользователя

Тренировка будет продолжаться до тех пор, пока пользователь не даст ответ на все задания или не допустит больше 50% ошибок, после чего пользователь увидит результаты своей тренировки и после нажатия на клавишу он сможет вернуться в меню. Пример успешного окончания тестирования приведён на рис. 2.9:

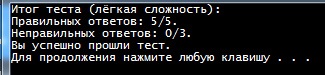


Рисунок 2.9 – Результаты успешного тестирования

Пример неуспешного окончания тестирования приведён на рис. 2.10:

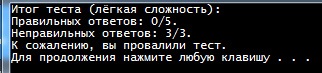


Рисунок 2.10 – Результаты неуспешного тестирования

Если пользователь выберет 4 пункт в меню, то ему будет предложено ввести пароль для входа в административный режим. Пример приведён на рис. 2.11:

C:\Users\Egor\Desktop\Новая папка (2)\bandicam 2019-11-27 10-59-34-245.jpg

Рисунок 2.11 – ввод пароля

В случае, если за три попытки пользователь не введёт правильно пароль, то программа сообщит ему об этом. Пример приведён на рис. 2.12:

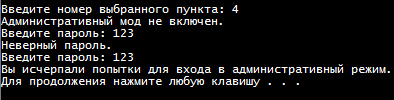


Рисунок 2.12 – сообщение о неудачном вводе

В случае успешного ввода пароля пользователь попадёт в административное меню программы. Пример приведён на рис. 2.13:

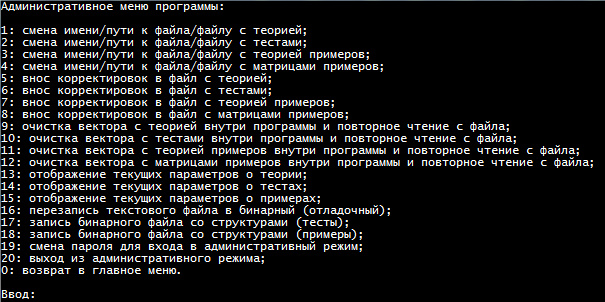


Рисунок 2.13 – административное меню программы

При выборе пунктов 1-4 администратору будет выведено имя файла (в зависимости от выбранного пункта) и просьба подтвердить ввод нового имени. Пример приведён на рис. 2.14:

C:\Users\Egor\Desktop\Новая папка (2)\bandicam 2019-11-27 11-00-48-868.jpg

Рисунок 2.14 – вывод имени файла и подтверждения замены

При выборе пунктов 5-8 администратору будет выведено подменю редактирования файла (в зависимости от выбранного пункта). Пример приведён на рис. 2.15:

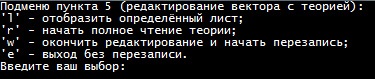


Рисунок 2.15 – вывод подменю выбранного пункта

При выборе пунктов 13-15 администратору будет выведена информация о прочитанных данных из файла (в зависимости от выбранного пункта). Пример приведён на рис. 2.16:

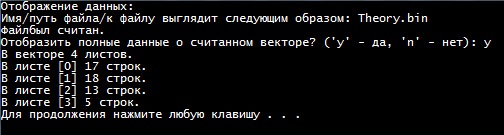


Рисунок 2.16 – вывод информации о считанном векторе

Если пользователь выберет 5 пункт в меню, то ему будет отображена информация о разработчике программы. Пример приведён на рис. 2.17:

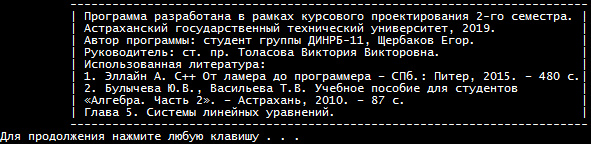


Рисунок 2.17 – Информация о разработчике

## 2.6 Сообщения системы

В таблице 2.12 приведены сообщения системы.

Таблица 2.12 – Сообщения системы

|  |  |  |
| --- | --- | --- |
| **№ п\п** | **Сообщение** | **Причина возникновения,  способ устранения** |
| 1 | Количество правильных ответов: | Количество правильных ответов |
| 2 | Количество неправильных ответов: | Количество неправильных ответов |
| 3 | Ошибка открытия файла. Будет совершён возврат в главное меню. | При попытке открытия не был обнаружен файл. Необходимо проверить целостность программы и возможность доступа к файлу. |
| 4 | Вы исчерпали попытки для входа в административный режим. | Пользователь ввёл неправильный пароль для административного режима 3 раза. Для появления возможности заново ввести пароль необходимо перезапустить программу. |

В случае появления других сообщений следует обратиться к разработчику.

# ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

**3.1 Проверка работоспособности модуля просмотра учебного материала**

1. Запустить программу на выполнение. Появится титульный лист (см. рис. 2.1).
2. Нажать любую клавишу. Убедиться, что появилось меню (см. рис. 2.2).
3. Выбрать пункт первый, убедиться, что предложен учебный материал по теме “Решение систем линейных уравнений методом Гаусса” (см. рис.2.3).
4. Нажать клавишу “f”. Убедиться, что на экран выведена следующая страница. Повторить пункт 4 до тех пор, пока переход к следующей странице станет недоступен.
5. Нажать клавишу “b”. Убедиться, что на экран выведена предыдущая страница. Повторить пункт 5, до тех пор, пока переход к предыдущей странице станет недоступен.
6. После выполнения пунктов 4 и 5, нажать клавишу “e”. Убедиться, что осуществлён возврат в главное меню.

**3.2 Проверка работоспособности модуля просмотра демонстрационного материала**

1. Нажать любую клавишу. Убедиться, что появилось меню (см. рис. 2.2).
2. Выбрать пункт второй, убедиться, что программа вывела теоретический материал для примеров (см. рис. 2.4).
3. Нажать любую клавишу, убедиться, что программа отобразила корректную матрицу системы из примера (см. рис. 2.5).
4. Нажать клавишу “f” убедиться, что программа перешла к следующей матрице.
5. Нажать клавишу “b” убедиться, что программа вернулась к предыдущей матрице.
6. Нажать клавишу “e”. Убедиться в том, что программа окончила отображение текущего примера и отобразила следующий (см. рис. 2.4, 2.5).
7. Повторять пункты 2, 3 до тех пор, пока демонстрация не будет окончена.
8. Нажать любую клавишу и убедиться, что программа вернулась в главное меню.

**3.3 Проверка работоспособности модуля тестирования**

1. Выбрать пункт третий, убедиться, что программа запустила тестирование.
2. Выбрать один из предоставленных вариантов сложности, убедиться, что количество вопросов соответствует заданной сложности (см. рис. 2.6).
3. Дать неверный ответ на вопрос. Убедиться, что программа осведомила пользователя о неправильном ответе на вопрос (см. рис. 2.7)
4. Ответить неправильно ещё на 2/4/7 вопросов.
5. Получить неудовлетворительный результат тестирования (см. рис. 2.10).
6. Нажать любую клавишу, подтвердить выход в меню. Убедиться, что произошел выход в меню.
7. Повторить пункты 1, 2.
8. Ответить правильно как минимум на 3/6/8 вопросов. Убедиться, что программа сообщает о правильном ответе (см. рис. 2.8).
9. Получить удовлетворительный результат тестирования (см. рис. 2.9).
10. Повторить пункт 6.
11. Выбрать 0 пункт в главном меню, убедиться, что произошел выход из программы.

**3.4 Проверка работоспособности административного модуля**

1. Выбрать пункт четвёртый, убедиться, что программа предложила ввод пароля

(см. рис 2.11).

1. Ввести неправильный пароль 2 раза, убедиться, что программа вернулась в главное меню (см. рис 2.12).
2. Снова выбрать четвёртый пункт, убедиться, что программа не предлагает ввод пароля.
3. Перезапустить программу, ввести правильный пароль, убедиться, что программа отобразила административное меню (см. рис 2.13).
4. Выбрать пункт 1/2/3/4, убедиться, что программа вывела имя файла и запросила подтверждение замены (см. рис 2.14), согласиться на замену, ввести новое имя файла, убедиться, что программа вернулась в административное меню, выйти в главное меню, выбрать пункт 1/2/3 (в зависимости от выбранного ранее), убедиться, что программа прочитала файл с новым именем.
5. Снова выбрать четвёртый пункт, выбрать пункт 5/6/7/8, убедиться, что программа вывела подменю (см. рис 2.15). Внести необходимые изменения, выйти с перезаписью. Вернуться в главное меню, выбрать пункт 1/2/3 (в зависимости от выбранного ранее), убедиться, что программа отобразила изменённый текст. Выйти из программы, запустить её заново, выбрать снова пункт 1/2/3 (в зависимости от выбранного ранее), убедиться, что программа отобразила изменённый текст.
6. Повторить пункт 4, затем выбрать пункт 9/10/11/12 (в зависимости от выбранного ранее), выйти в главное меню, выбрать пункт 1/2/3 (в зависимости от выбранного ранее), убедиться, что программа отобразила текст из выбранного файла.
7. Снова выбрать четвёртый пункт, затем выбрать пункт 13/14/15, убедиться, что программа отобразила правильные данные.
8. Выбрать пункт 16/17/18, указать имя .txt файла, из которого будут прочитаны новые данные, или ввести новые данные вручную. Перезапустить программу, затем выбрать пункт 1/2/3 (в зависимости от выбранного ранее), убедиться, что программа отобразила перезаписанные данные.
9. Снова выбрать четвёртый пункт, затем выбрать пункт 19, ввести новый пароль. Выбрать пункт 20, повторить пункт первый, повторить пункт четвёртый с новым паролем.
10. Выбрать четвёртый пункт, затем выбрать пункт 0, убедиться, что программа вернулась в главное меню, снова выбрать пункт 4, убедиться, что программа не требует пароля для отображения.

# ЗАКЛЮЧЕНИЕ

В результате курсового проектирования разработана учебно-демонстрационная программа «Решение системы линейных уравнений методом Гаусса». Программа предлагает учебный материал, демонстрацию решения и задания на проверку навыков решения, проверяя правильность ответа. Тренировка продолжается до получения ответов на все заданные вопросы или до совершения не более 50% ошибок, по окончании выдаётся сообщение о том, насколько успешно были выполнены задания.

Программа отвечает поставленным требованиям и может быть использована для обучения студентов.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Эллайн А. C++ От ламера до программера – СПб.: Питер, 2015. – 480 с.
2. Булычева Ю.В., Васильева Т.В. Учебное пособие для студентов «Алгебра. Часть 2». – Астрахань, 2010. – 87 с. (Глава 5. Системы линейных уравнений).

# ПРИЛОЖЕНИЕ 1 Техническое задание

**на разработку учебно-демонстрационной программы**

**«Решение системы линейных уравнений методом Гаусса»**

по дисциплине «Программирование и информатика»

Направление 09.03.01 Информатика и вычислительная техника

Исполнитель: студент гр. ДИНРб11 **Щербаков Е.В.**

1. **Назначение, цели и задачи разработки**

**Цель разработки –** автоматизация обучения по теме «Решение системы линейных уравнений методом Гаусса».

**Назначение разработки:**

* + повышение качества знаний пользователей;
  + снижение нагрузки на преподавателя.

**Основные задачи,** решаемые разработчиком в процессе выполнения курсового проекта:

* + анализ предметной области;
  + разработка программного продукта в соответствии с требованиями;
  + документирование проекта в соответствии с установленными требованиями.

1. **Характер разработки:** прикладная квалификационная работа.
2. **Основания для разработки**
   * Учебный план направления 09.03.01 «Информатика и вычислительная техника» 2018 года набора.
   * Рабочая программа дисциплины «Программирование и информатика».
   * Распоряжение по кафедре АСОИУ №\_\_\_\_ от «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2019 г.
3. **Плановые сроки выполнения –** весенний семестр 2018/19 учебного года:

Начало «22» января 2019 г.

Окончание «25» мая 2019 г.

1. **Требования к проектируемой системе**

**5.1 Требования к функциональным характеристикам**

Проектируемая система представляет собой консольное приложение и должна обеспечивать выполнение следующих основных функций:

* предоставляет пользователю учебный материал по заданной теме, записанный в бинарный файл.
* разбивает учебный материал на страницы, которые можно пролистывать вперед и назад.
* предоставляет демонстрационную часть, снабженную комментариями к происходящему процессу, с возможностью просмотреть предыдущие шаги вплоть до первого.
* предоставляет пользователю тестирование, которое представляет собой 5/10/15, случайно выбранных из базы, теоретических вопроса.
* сравнивает каждый ответ пользователя с эталонным, вычисленным программой.
* уведомляет пользователя о совершении ошибки сразу после ответа на задание в тестировании
* сообщает результаты тестирования в виде оценки (без указания текста вопроса, в котором была сделана ошибка)
* хранит базу вопросов в бинарном файле, который редактируется.
* продолжает тестирование пока пользователь не ответит на все задания или не совершит больше 50% ошибок.
* позволяет редактировать файлы с материалом внутри программы.

**5.2 Требования к эксплуатационным характеристикам**

Программа не должна аварийно завершаться при любых действиях пользователя.

Время реакции программы на действия пользователя не должно превышать 10 секунд.

**5.3 Требования к программному обеспечению:**

Средства разработки: интегрированная среда CodeBlocks 17.12, язык C++ (стандарт С++ 11).

Операционная система: Windows XP (x86) с пакетом обновления 3 (SP3).

**5.4 Требования к аппаратному обеспечению:**

Рекомендуемая конфигурация:

* + Intel-совместимый процессор с частотой не менее 1,6 ГГц;
  + не менее 512 МБ ОЗУ;
  + не менее 20 MБ свободного места на диске;
  + дисковод CD-ROM/DVD-ROM.

1. **Стадии и этапы разработки**
2. **Эскизный проект (ЭП)**
   * Анализ предметной области.
   * Подготовка проектной документации.
3. **Технический проект (ТП)**
   * Разработка структур и форм представления данных.
   * Разработка структуры программного комплекса.
   * Подготовка пояснительной записки.
4. **Рабочий проект (РП)**
   * Программная реализация.
   * Тестирование и отладка программы.
   * Подготовка программной и эксплуатационной документации.
5. **Эксплуатация (Э)**

Описание и анализ результатов проведенного исследования.

1. **Требования к документированию проекта**

К защите курсового проекта должны быть представлены следующие документы:

* + Пояснительная записка к курсовому проекту:
  + Презентация доклада.
  + Программа, презентация и пояснительная записка к курсовому проекту на оптическом носителе.

Требования к структуре документов определены соответствующими стандартами ЕСПД.

Требования к оформлению определены соответствующими методическими указаниями.

1. **Порядок контроля и приемки**

Контроль выполнения курсового проекта проводится руководителем поэтапно в соответствии с утвержденным графиком выполнения проекта.

На завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) проекта к защите.

Защита курсового проекта проводится комиссией в составе не менее двух человек, включая руководителя проекта.

В процессе защиты проекта исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует ее программную реализацию.

При выставлении оценки учитывается:

* + степень соответствия представленной разработки требованиям технического задания;
  + качество программной реализации, документации и доклада по теме проекта;
  + соблюдение исполнителем графика выполнения курсового проекта.

1. **Литература**
2. Эллайн А. C++ От ламера до программера – СПб.: Питер, 2015. – 480 с.
3. Булычева Ю.В., Васильева Т.В. Учебное пособие для студентов «Алгебра. Часть 2». – Астрахань, 2010. – 87 с. (Глава 5. Системы линейных уравнений).

# ПРИЛОЖЕНИЕ 2 База вопросов программы

Правильный ответ всегда идёт первым.

**1. Метод Гаусса – это**

1) метод последовательного исключения переменных

2) метод дополнения

3) метод замены переменных

**2. Метод Гаусса заключается в приведении данной системы**

1) к ступенчатому виду

2) к квадратному виду

3) к выделению нулевых элементов

**3. При решении систем уравнений методом Гаусса нельзя:**

1) умножать любой столбец на некоторое число

2) любую строку умножать или делить на некоторое число

3) переставлять местами строки

**4. К «обратному ходу метода Гаусса» относится следующее**

1) «лишние» уравнения исключаются из системы

2) из последнего уравнения определяется самое правое неизвестное

3) составляется матрица свободных членов

**5. Если при выполнении преобразований появились уравнения вида 0\*x1 + 0\*x2 + … + 0\*xn = 0 , то неверно следующее:**

1) неопределённой является и исходная система

2) система не имеет решений

3) число уравнений меньше числа неизвестных

**6. Если все элементы матрицы свободных членов равны нулю, то**

1) ни один из вариантов не является правильным

2) система обязательно имеет решения

3) cистема не имеет решений

**7. Что можно сделать со строкой матрицы?**

1) ни один из вариантов не является правильным

2) умножить её на ноль

3) умножить её на другой столбец

**8. Что относится к элементарным преобразованиям?**

1) умножение/деление строки на любое число, отличное от нуля

2) прибавление другой строки, умноженной на ноль

3) ни один из вариантов не является правильным

**9. В методе Гаусса система записывается с помощью:**

1) расширенной матрицы системы

2) обычной матрицы системы

3) алгебраической матрицы

**10. Дана система**

**{ x + 2y + 3z = 1**

**{ 2x - y + 2z = 6**

**{ x + y + 5z = -1**

**После выполнения элементарных преобразований вторая строка матрицы будет выглядеть следующим образом:**

1) (0; 1; -2 | 2)

2) (1; 2; 3 | 1)

3) (0; 1; -2 | 0)

**11. Дана система**

**{ 8x + 7y + 3z = 18**

**{ -7x - 7y - 4z = -11**

**{ -6x + 5y - 4z = -15**

**После выполнения элементарных преобразований первая строка матрицы будет выглядеть следующим образом:**

1) (1; 3; -1 | 7)

2) (3; 14; -71 | -5)

3) (1; -5; -1 | -5)

**12. Если количество уравнений меньше, чем количество переменных, то это значит, что:**

1) оба ответа верны

2) система может быть несовместной

3) система может иметь бесконечно много решений

**13. Дана система**

**{ 3x + 2y + z = 2**

**{ x - y + 2z = -1**

**{ 2x + 2y + z = 3**

**После выполнения элементарных преобразований вторая строка матрицы будет выглядеть следующим образом:**

1) (0; 1; -1 | 1)

2) (0; -1; 2 | -1)

3) (0; 1; -2 | 1)

**14. Дана система**

**{ x + 2y + 5z = -9**

**{ x - 1y + 3z = 2**

**{ 3x - 6y - 1z = 25**

**После выполнения элементарных преобразований первая строка матрицы будет выглядеть следующим образом:**

1) (1; 2; 5 | -9)

2) (1; -3; -2 | 11)

3) (1; 2; -8 | 2)

**14. Дана система**

**{ u + 2x + 3y - 2z = 1**

**{ 2u - x - 2y - 3z = 2**

**{ 3u + 2x - y + 2z = -5**

**{ 2u - 3x + 2y + z = 11**

**Эта система:**

1) имеет только одно решение

2) имеет бесконечно много решений

3) не имеет решений