

Министерство образования Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Программирование»
на тему «База данных»

20.05.26

О.А.



Выполнил:

студент группы 25BVB4

Волковец А.С.

Принял:

К.И. Токарев А.Н.

Пенза, 2026

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания " 13 " 02.25

Дата защиты проекта " "

Руководитель 

Задание получил " 25 " 02 2026г.

Студент АВ Волковец А.С.

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"



Зав. кафедрой ВТ

М.А. Митрохин

«__» ____ 20__ г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Восковцеву Андрею Сергеевичу Группа 25 ВВВ4

Тема проекта: Разработка программы сложной структуры методом нисходящего программирования.

Исходные данные (технические требования) на проектирование

База данных по студентам. Технические требования: Добавление записей, удаление записей, Редактирование записей, Поиск и сортировка записей, Вывод записей по отпр. критериям, Создание новой базы данных, Сохранение текущей базы данных, Загрузка сохраненной базы данных.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Содержание

Введение.....	5
Постановка задачи	6
Выбор решения.....	7
Описание разработки программы.....	10
Отладка и тестирование.....	12
Описание программы.....	13
Руководство пользователя	31
Заключение	42
Список используемых источников	44

Введение

На сегодняшний день информация – ценный источник знаний, систематизации которой, позволяет оптимизировать работу как человека, так и общества в целом. Компьютерные технологии во многом упрощают нашу жизнь в самых разнообразных сферах нашей деятельности. И именно компьютерные технологии позволяют грамотно взаимодействовать с информацией, обрабатывать её и хранить на компактных носителях.

База данных представляет собой упорядоченную, структурированную совокупность данных, подобранных таким образом, что эти данные могут легко и быстро обрабатываться на ЭВМ.

История баз данных в узком смысле рассматривает базы данных в традиционном (современном) понимании. Эта история начинается с 1955 года, когда появилось программируемое оборудование обработки записей. Программное обеспечение этого времени поддерживало модель обработки записей на основе файлов. Для хранения данных использовались перфокарты.

Оперативные сетевые базы данных появились в середине 1960-х. Операции над оперативными базами данных обрабатывались в интерактивном режиме с помощью терминалов. Простые индексно-последовательные организации записей быстро развились к более мощной модели записей, ориентированной на наборы.

В наше время базы данных являются неотъемлемой частью любой организации, любого предприятия. Динамические сайты используют базу данных чтобы хранить данные пользователя, как и стриминговые сервисы с подписочной моделью. Банки хранят в огромных базах данных данные пользователей.

Данная программа является информационно-поисковой системой и позволяет работать с простейшей базой данных студентов, позволяющей хранить, добавлять, редактировать, удалять информацию о каждом студенте.

Постановка задачи

Необходимо разработать программу для работы с базой данных сотовых студентов. Программа должна иметь интуитивно понятный интерфейс для пользователя. Необходимо определить названия пунктов меню, построить алгоритмы работы с массивами структур для фильтрации и сортировки записей по определенным критериям, изучить функции для работы с файлами, чтобы сохранять базу данных в файл и загружать из файла.

Многомодульность программы. Программа должна быть поделена на логические модули. Это упростит поиск ошибок при отладке и тестировании программы, а также позволит легко расширять функционал программы.

Использование сложных типов данных – массивов, структур, файлов. Это необходимо для более простой и интуитивной обработки данных в коде программы.

Режим работы видеосистемы – текстовый/графический. Для начала необходимо определиться с типом интерфейса и с элементами управления, затем необходимо изучить способы их реализации.

Устройство ввода-вывода – клавиатура и мышь. Необходимо различать и идентифицировать действия, произведенные с их помощью, это облегчит использование программы.

Интерфейс должен быть построен на основе меню. Это необходимо для создания интуитивно понятного интерфейса для пользователя.

Заставка необходима для того, чтобы пользователь, запустивший программу, смог получить достаточную информацию о ней.

Выбор решения

При запуске программы открывается приветственный экран. Далее появляется меню, состоящее из семи пунктов:

1 - Вывести список студентов. На экране в виде форматированной таблицы отображается вся текущая база данных. Поле уникального идентификатора (ID) каждого студента автоматически дополняется ведущими нулями до 4 символов и выравнивается по правому краю (например, вместо 3 выводится 0003). Все текстовые поля выравниваются по левому краю для обеспечения читаемости таблицы.

2 - Добавить нового студента. Запускается пошаговый процесс интерактивного добавления записи:

- **Автоматическая генерация ID:** Идентификатор генерируется программой автоматически в диапазоне от 1 до 9999 случайным образом (`srand(time)`). Программа в цикле проверяет уникальность сгенерированного числа: если такой ID уже существует в базе, генерируется новый, что исключает дубликаты. Максимальная емкость базы данных ограничена 9999 записями.

- **Валидация текстовых полей:** При вводе полей «Фамилия», «Имя», «Отчество» и «Факультет» программа посимвольно проверяет строку. При обнаружении цифр выдается ошибка с предложением повторить ввод. Ввод поля «Группа» допускает использование цифр.

- **Валидация числовых полей:** При вводе полей «Возраст» и «Курс» программа проверяет тип данных (защита от ввода букв), а также проверяет значения на вхождение в логический диапазон (возраст от 16 до 100 лет, курс от 1 до 6). При ошибке запрашивается повторный ввод.

По окончании ввода данные сохраняются в оперативной памяти и автоматически перезаписывают рабочий файл базы данных на диске.

3 - Редактировать данные студента. Пользователь вводит ID записи. Программа находит студента, выводит его текущую анкету и предлагает обновить поля. Изменение полей ФИО, факультета, группы, возраста и курса

происходит под контролем тех же функций валидации, что и при добавлении. Изменение уникального случайного ID заблокировано из соображений целостности базы. Изменения сразу сохраняются в файл.

4 - Удалить студента по ID. Пользователь вводит ID записи, которую необходимо удалить. Программа производит поиск элемента в динамическом массиве. Если запись найдена, она полностью стирается из оперативной памяти с помощью механизма *Erase-Remove*, после чего обновленный вектор мгновенно синхронизируется с файлом на жестком диске (мусорные записи не остаются). Если ID не найден, выдается ошибка.

5 - Поиск и фильтрация. Пользователь переходит в новое подменю, где выбирает один из пяти критериев для фильтрации:

5.1 Поиск по ID. Пользователь вводит идентификатор. Если студент существует, программа выводит его строку, иначе выдается ошибка.

5.2 Поиск по Фамилии. Пользователь вводит фамилию или её часть. Программа выполняет поиск по частичному совпадению и выводит подходящие записи.

5.3 Фильтр по Факультету. Аналогично предыдущему пункту, выборка идет по названию факультета.

5.4 Фильтр по Группе. Выводятся все студенты, чье поле «Группа» совпало с введенным значением.

5.5 Фильтр по Курсу. Фильтрация базы данных по номеру курса.

6 - Ассемблерный расчет (Средний возраст). Программа производит низкоуровневый расчет среднего возраста всех студентов. Вычисления производятся в блоке ассемблерной вставки (`__asm` для x86-архитектуры). В ассемблерный цикл передается указатель на массив данных в памяти и размер структуры. С помощью команд смещения (`offsetof`) и суммирования в регистре EAX рассчитывается общий возраст, который затем делится на количество записей. Результат выводится на экран.

7 - Создать новую БД. Пользователь вводит название для нового файла базы данных с расширением .dat. Программа создает пустой файл на диске, а текущий список студентов в оперативной памяти полностью очищается для работы с новой базой.

8 - Удалить файл БД. Программа производит полное физическое удаление текущего файла базы данных с жесткого диска, а также очищает вектор данных в оперативной памяти.

9 - Загрузить базу данных из файла. Пользователь вводит название существующего файла. Программа проверяет его наличие на диске. Если файл найден, данные считываются в оперативную память, затирая предыдущую рабочую сессию. Если файл не существует, выдается ошибка, а текущие данные в памяти остаются нетронутыми.

0 - Выход. Программа закрывает все потоки, гарантированно перезаписывает текущее состояние базы данных в файл и завершает работу.

Навигация в главном меню осуществляется с помощью ввода цифр от 0 до 9. Система защищена от сбоев при случайном вводе буквенных символов вместо номеров пунктов.

Описание разработки программы

Разработанная программа имеет модульную структуру и состоит из нескольких логических компонентов, которые распределены по файлам проекта и реализовывались в следующем порядке:

1. **Database.h** – заголовочный файл проекта. Содержит определение класса Student с инкапсулированными полями данных (ФИО, факультет, группа, возраст, курс) и прототипы всех функций для работы с базой данных.
2. **curs.cpp** – файл главного модуля программы. Содержит точку входа main(), код приветственного экрана, инициализацию динамического вектора студентов и реализацию интерактивного циклического меню управления.
3. **loadFromFile.cpp (в составе Database.cpp)** – модуль автоматической загрузки базы данных из бинарного файла при старте приложения (инициализация сессии).
4. **printStudentRow.cpp / displayDatabase.cpp (в составе Database.cpp)** – модуль форматированного вывода записей. Отвечает за построение таблицы, принудительное выравнивание ID по правому краю с заполнением ведущими нулями до 4 символов и вывод текстовых полей.
5. **inputValidString.cpp / inputValidInt.cpp (в составе Database.cpp)** – вспомогательные модули валидации ввода. Обеспечивают защиту от «дурака»: блокируют ввод цифр в текстовые поля и букв в числовые поля с контролем логических диапазонов.
6. **addStudent.cpp (в составе Database.cpp)** – пользовательская оболочка для добавления новой записи в оперативную память с подсистемой циклической генерации уникального случайного ID (от 1 до 9999) и проверкой на дубликаты.

7. **saveToFile.cpp (в составе Database.cpp)** – модуль сохранения базы данных. Отвечает за побайтовую запись вектора структур на жесткий диск в бинарном режиме.
8. **deleteStudent.cpp (в составе Database.cpp)** – модуль физического удаления записи из динамического массива на основе механизма *Erase-Remove* с мгновенной перезаписью файла на диске для предотвращения появления мусора.
9. **findStudents.cpp / searchMenu.cpp (в составе Database.cpp)** – модуль поиска и фильтрации данных по пяти различным критериям (ID, фамилия, факультет, группа, курс) с поддержкой частичного совпадения строк.
10. **calculateAverageAge.cpp (в составе Database.cpp)** – расчетно-аналитический модуль. Выполняет подсчет среднего возраста студентов с использованием низкоуровневой ассемблерной вставки (`__asm` под x86-архитектуру) и прямого обращения к памяти через смещения полей (`offsetof`).
11. **loadCustomDatabase.cpp (в составе Database.cpp)** – модуль интерактивного переключения баз данных. Позволяет пользователю загрузить сторонний файл `.dat`, предварительно проверяя его физическое существование на диске.

По порядку реализации модулей можно понять, что использовался **восходящий подход** (Bottom-Up) реализации программы. Этот метод основан на том, что вначале проектируются и кодируются базовые элементы, структуры данных и низкоуровневые функции валидации и работы с памятью/файлами, после чего на их основе выстраивается высокоуровневая логика — пользовательское меню, подсистема фильтрации и интерфейс взаимодействия.

Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2019. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных. Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено огромное количество проблем, связанных с работой с файлами, работой с памятью, работой с полями структуры

Описание программы

Приложение curs.exe является основным модулем программы. При запуске программы происходит вывод заставки-приветствия, после чего пользователю необходимо выбрать пункт меню для дальнейшего использования базы данных. Выход из меню осуществляется после выбора пользователем нужного пункта. Описание состояний программы выполнено в таблице 1.

Таблица 1 - Описание состояний главного меню программы

К л а в и ш и , вызывающие событие	Д е й с т в и е пользователя	Действие программы
1, Enter	В ы б р а н п у н к т «Вывести список студентов»	Запускается модуль форматированного вывода таблицы, отображающий все записи из оперативной памяти на экран.
2, Enter	В ы б р а н п у н к т «Добавить нового студента»	Запускается диалог интерактивного ввода данных, генерируется уникальный ID и происходит добавление записи в БД с автосохранением в файл.
3, Enter	В ы б р а н п у н к т «Удалить студента по ID»	Запускается запрос идентификатора (ID) и последующее физическое удаление записи из памяти и файла базы данных.
4, Enter	В ы б р а н п у н к т « П о и с к и фильтрация»	Открывается новое подменю поиска, предоставляющее выбор критериев для фильтрации студентов.
5, Enter	В ы б р а н п у н к т « Р а с с ч и т а т ь средний возраст студентов»	Запускается расчетно-аналитический модуль с использованием низкоуровневой ассемблерной вставки __asm.
6, Enter	В ы б р а н п у н к т «Создать новую б а з у д а н н ы х (файл)»	Запускается диалог ввода имени для инициализации нового пустого бинарного файла на диске.

Продолжение таблицы 1

7, Enter	В ы б р а н п у н к т «Удалить текущий файл базы данных»	Выполняется физическое удаление рабочего файла .dat с жесткого диска и очистка текущего вектора в памяти.
8, Enter	В ы б р а н п у н к т «Загрузить базу данных из файла»	Запускается диалог выбора и проверки стороннего файла для его последующей полной выгрузки в оперативную память.
9, Enter	В ы б р а н п у н к т «Редактировать данные студента»	Запускается поиск записи по ID и диалог изменения полей существующего студента под контролем модулей валидации.
0, Enter	В ы б р а н п у н к т « С о х р а н и т ь и выйти»	Выполняется финальная синхронизация данных с рабочим файлом и закрытие приложения.

Поскольку пункт меню 4 открывает вложенное интерактивное меню, для него составляется отдельная таблица 2 описания состояний.

Таблица 2 - Описание состояний меню поиска и фильтрации

К л а в и ш и , в ы з ы в а ю щ и е событие	Д е й с т в и е пользователя	Действие программы
1, Enter	Выбран вариант «Поиск по ID»	Программа выполняет точечный поиск и выводит анкету студента с указанным идентификатором.
2, Enter	Выбран вариант « П о и с к п о Фамилии»	Выполняется сканирование базы и вывод всех студентов, чья фамилия полностью или частично совпадает с запросом.
3, Enter	Выбран вариант « Ф и л ь т р п о Факультету»	Программа фильтрует динамический массив и выводит на экран таблицу студентов выбранного факультета.
4, Enter	Выбран вариант « Ф и л ь т р п о	Программа группирует и выводит список студентов, обучающихся в заданной группе.

Продолжение таблицы 2

5, Enter	Выбран вариант «Фильтр по Курсу»	Осуществляется числовой отбор и вывод студентов, соответствующих указанному номеру курса.
----------	----------------------------------	---

Далее на рисунке 1 представлена схема взаимодействия программы

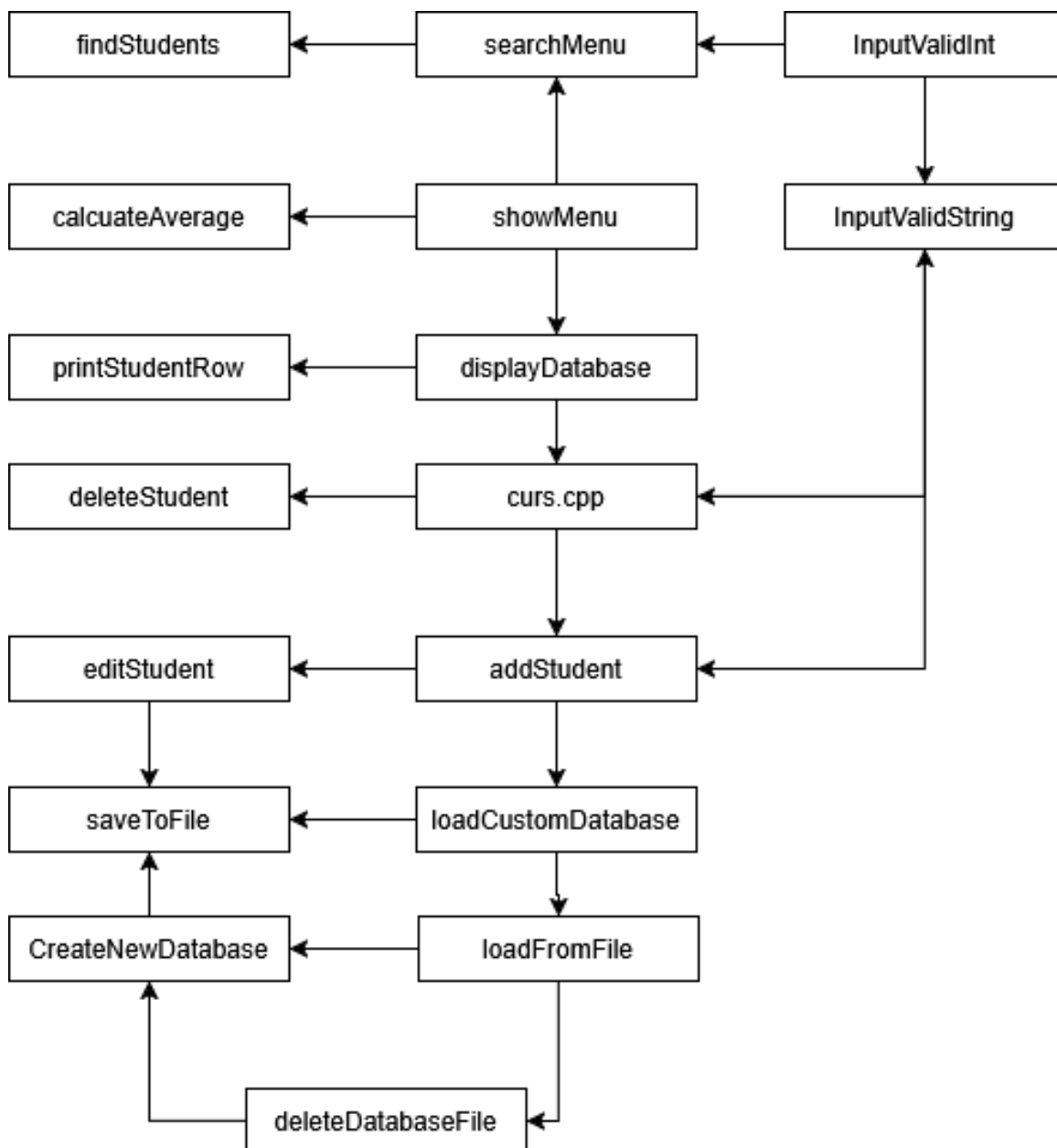


Рисунок 1 – схема взаимодействия программы

Ниже на рисунке 2 представлена схема данных.

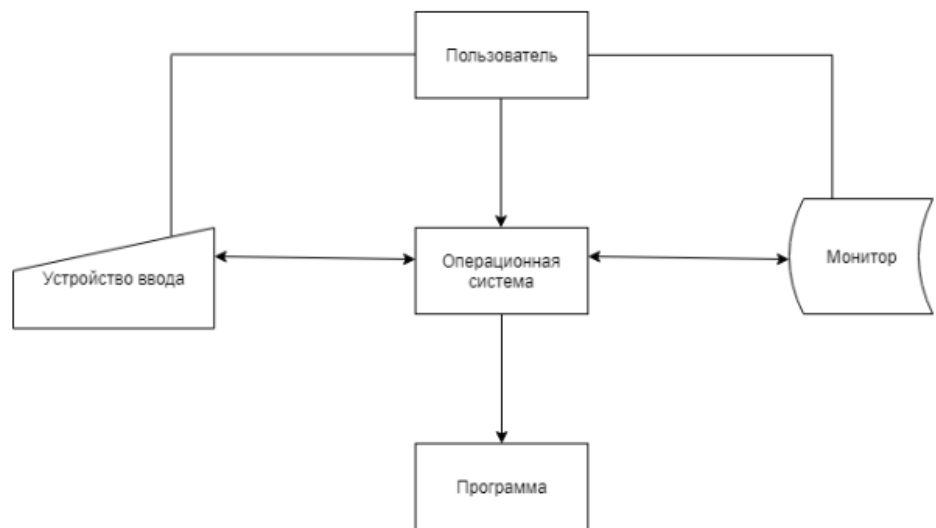


Рисунок 2 – схема данных

Блок-схема функции main представлена на рисунке 2.1.



Рисунок 2.1 - Блок-схема функции main

Блок-схема функции inputValidString представлена на рисунке 2.2.

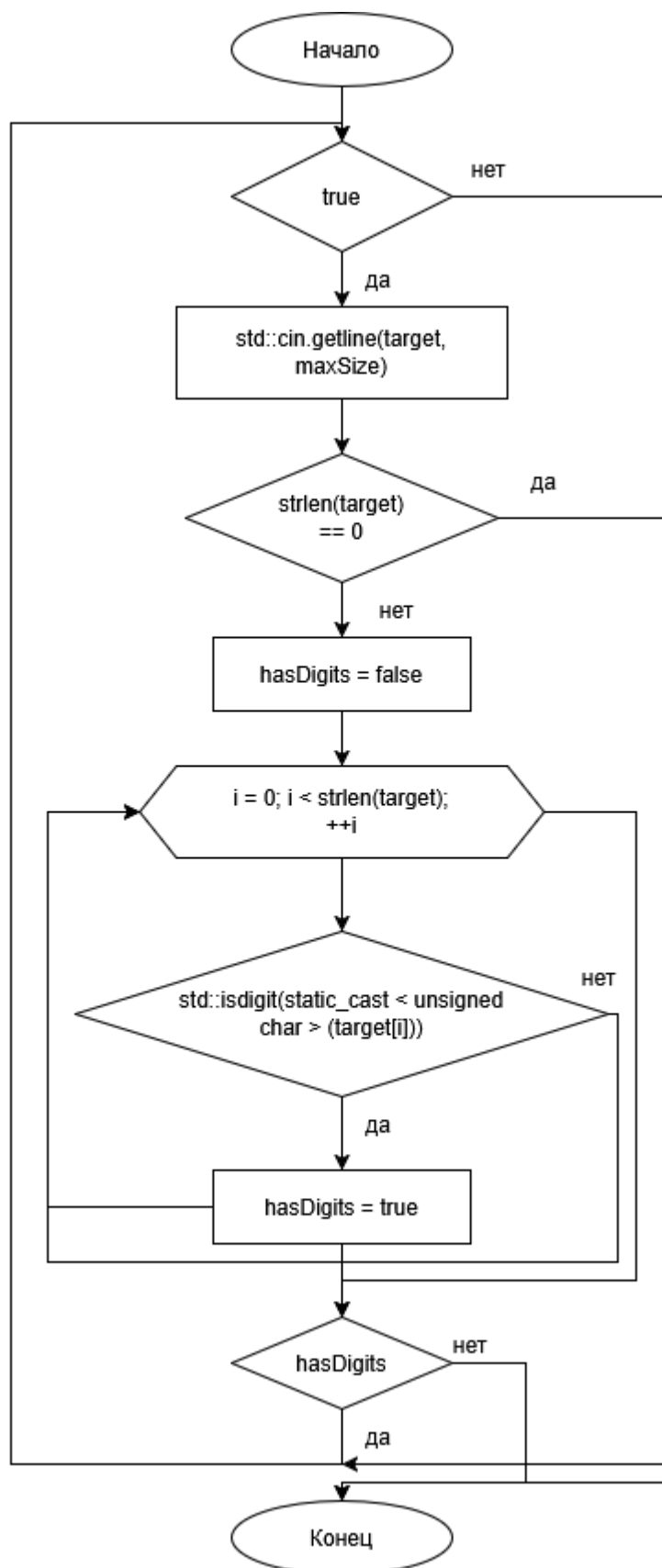


Рисунок 2.2 - Блок-схема функции inputValidString

Блок-схема функции inputValidInt представлена на рисунке 2.3.

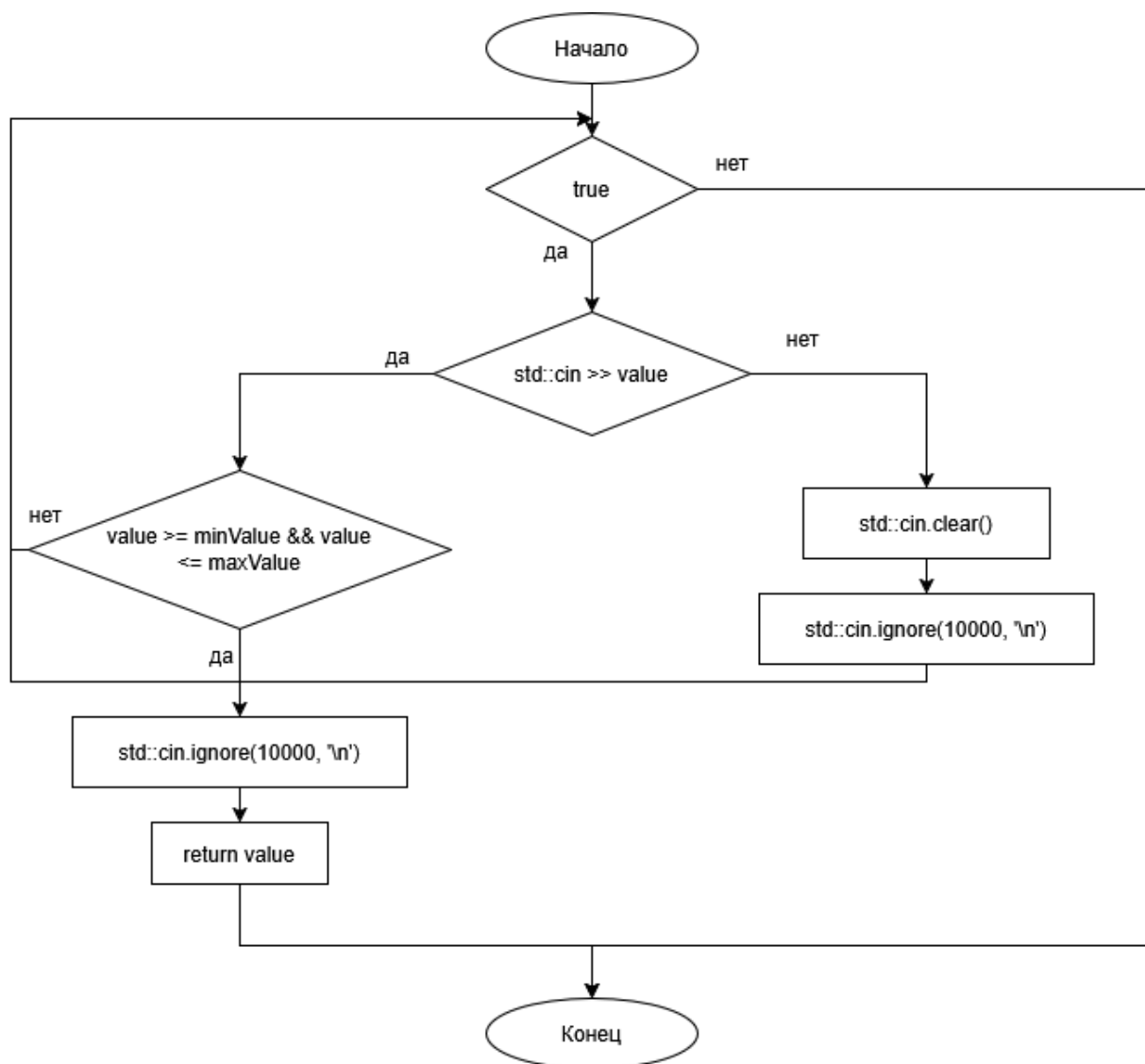


Рисунок 2.3 - Блок-схема функции inputValidInt

Блок-схема функции createNewDatabase представлена на рисунке 2.4.

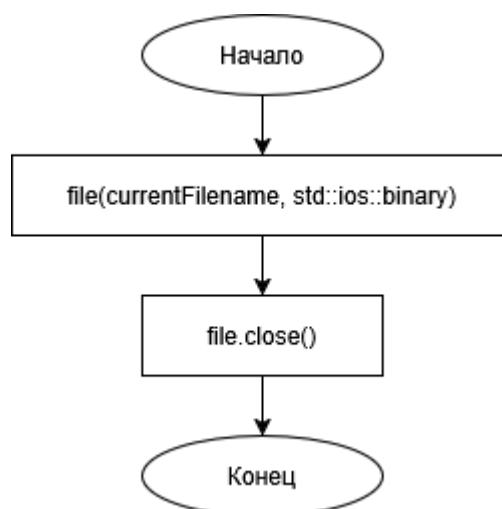


Рисунок 2.4 - Блок-схема функции createNewDatabase

Блок схема функции deleteDatabaseFile Представлена на рисунке 2.5.

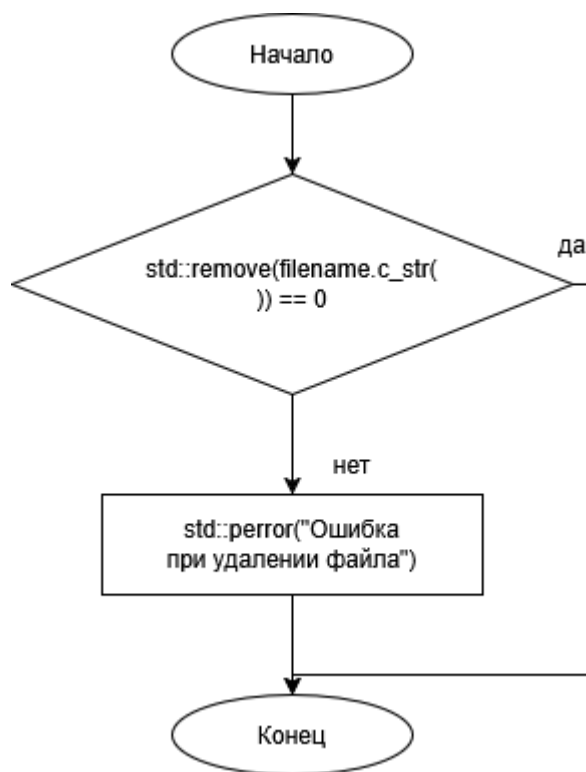


Рисунок 2.5 - Блок-схема функции `deleteDatabaseFile`

Блок-схема функции `saveToFile` представлена на рисунке 2.6.

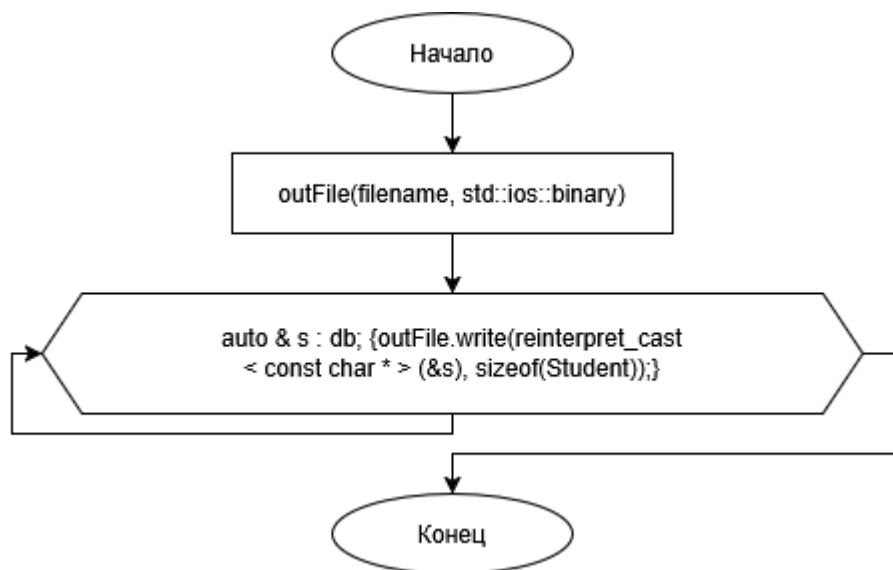


Рисунок 2.6 - Блок-схема функции `saveToFile`

Блок-схема функции `loadFromFile` представлена на рисунке 2.7.

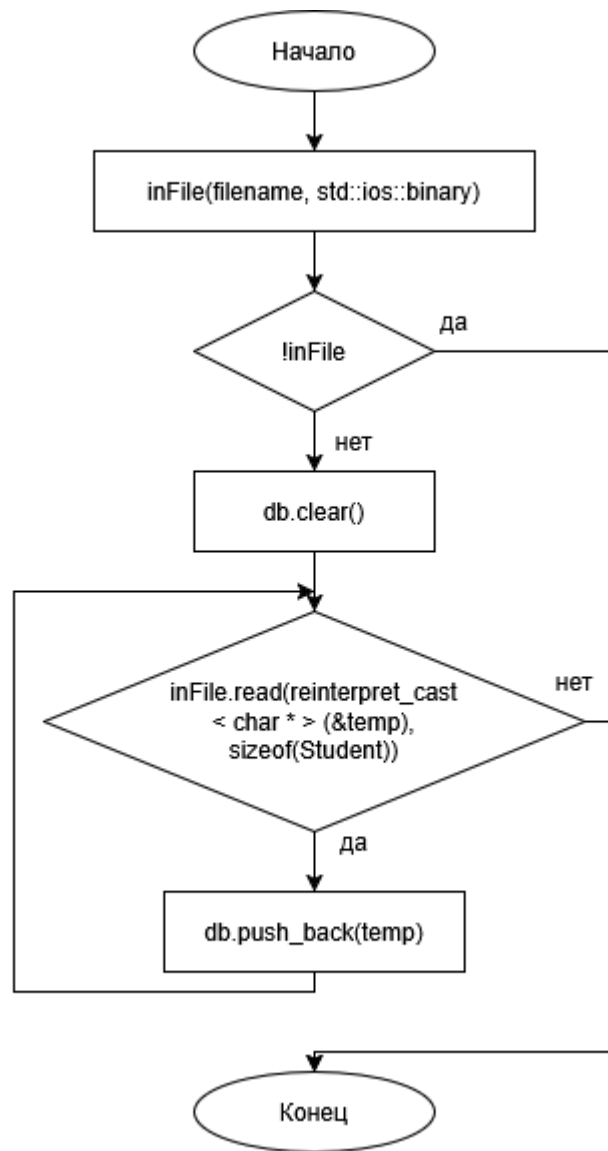


Рисунок 2.7 - Блок-схема функции loadFromFile

Блок-схема функции loadCustomDatabase представлена на рисунке 2.8.

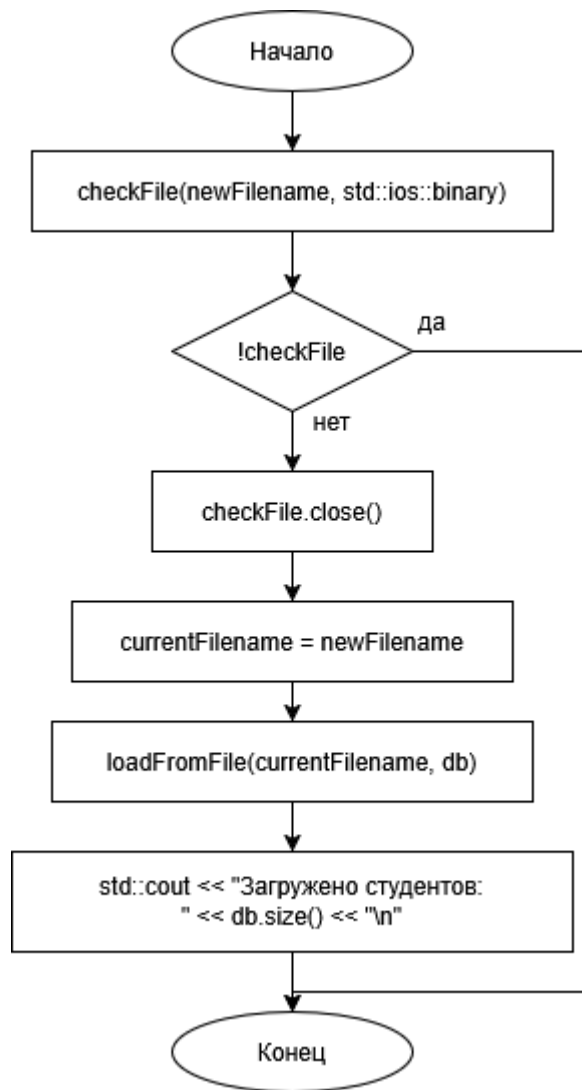


Рисунок 2.8 - Блок-схема функции loadCustomDatabase

Блок-схема функции addStudent представлена на рисунке 2.9.

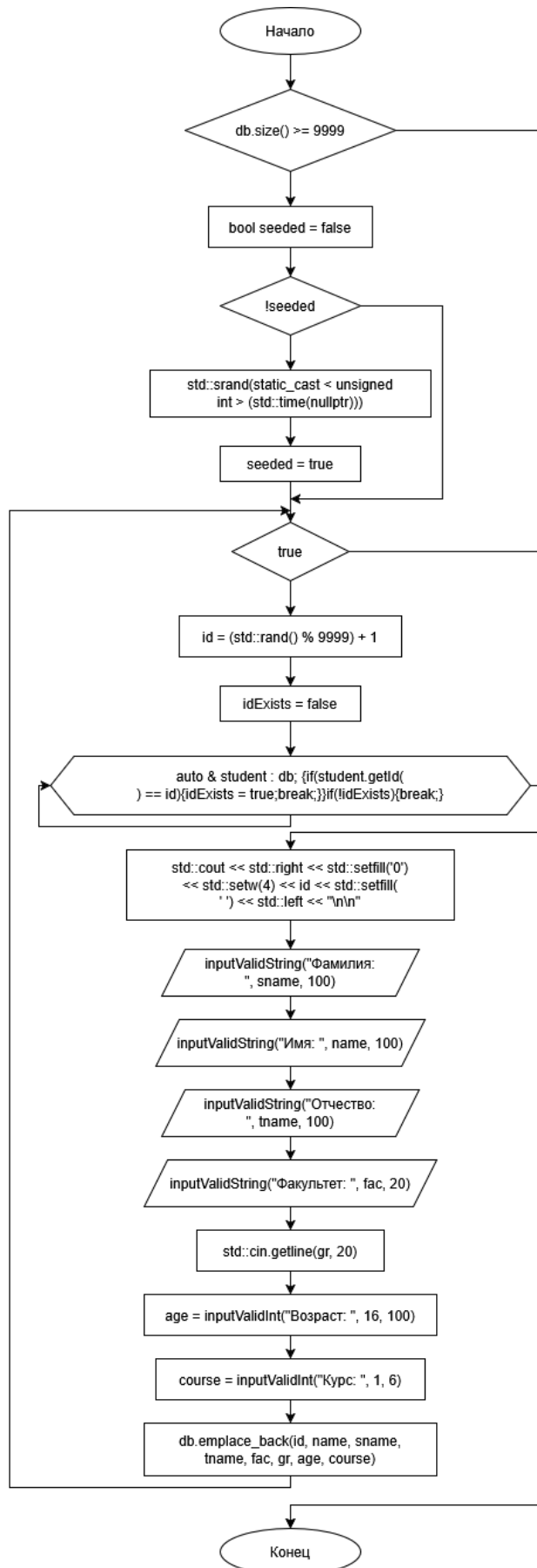


Рисунок 2.9 - Блок-схема функции addStudent

Блок-схема функции editStudent представлена на рисунке 2.10.

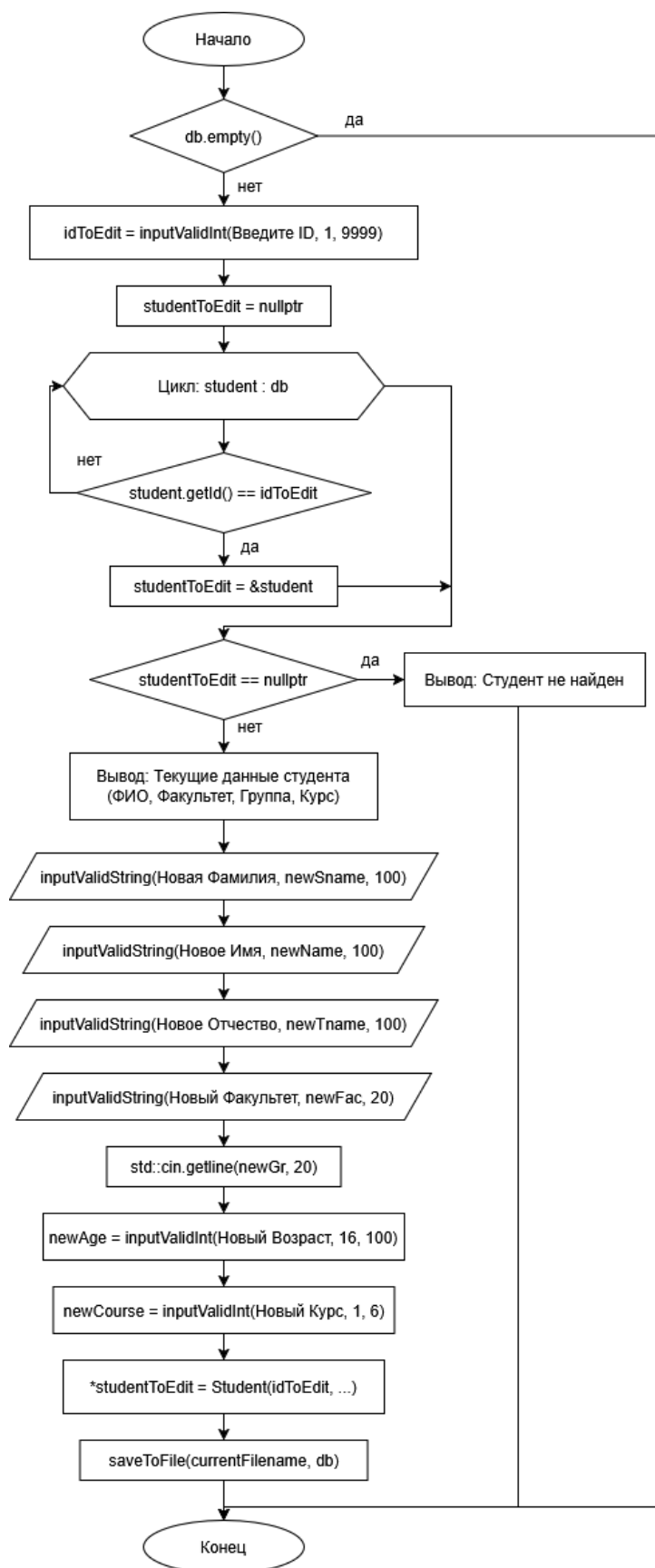


Рисунок 2.10 Блок-схема функции editStudent

Блок-схема функции findStudents представлена на рисунке 2.11.



Рисунок 2.11 - Блок-схема функции findStudents

Блок-схема функции deleteStudent представлена на рисунке 2.12.

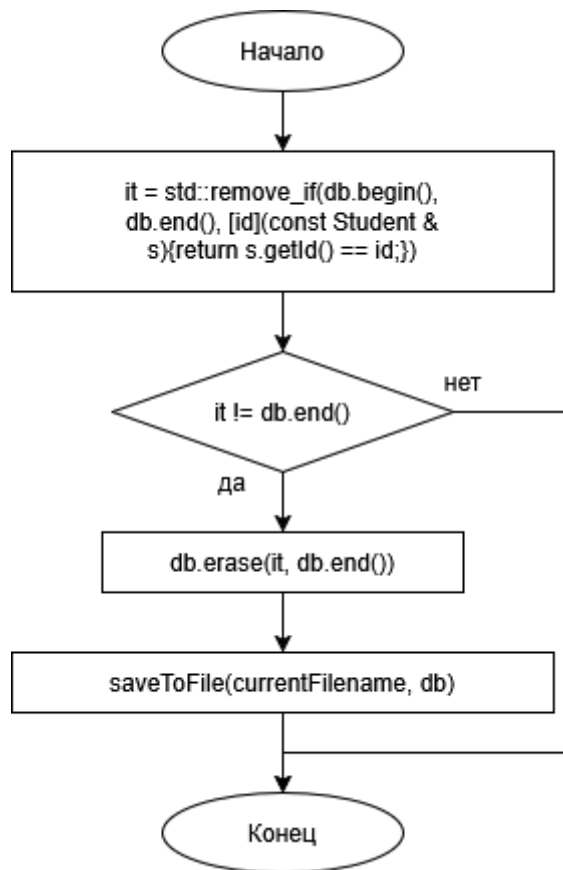


Рисунок 2.12 - Блок-схема функции `deleteStudent`

Блок-схема функции `drawHeader` представлена на рисунке 2.13.

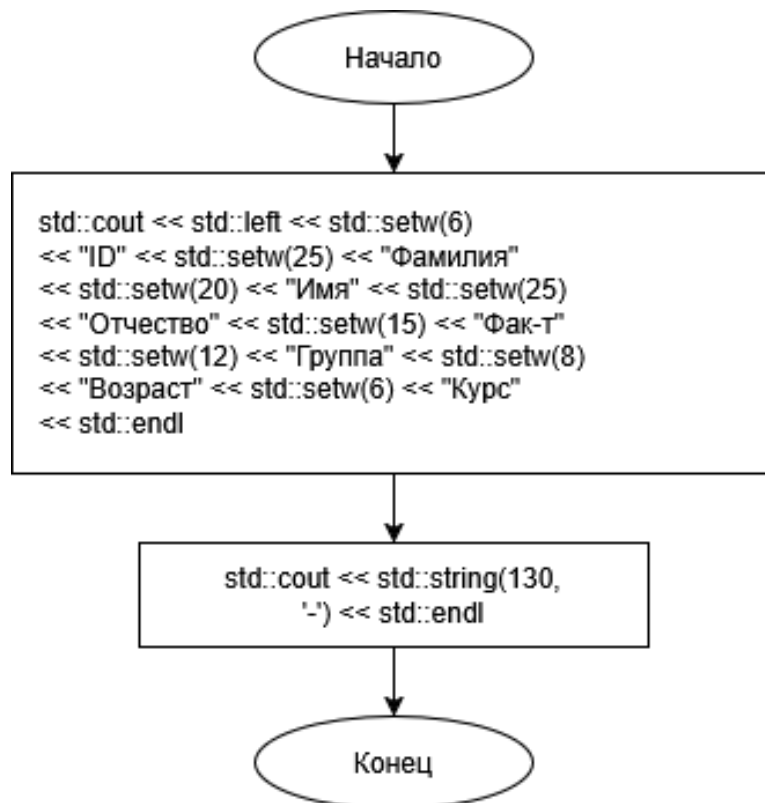


Рисунок 2.13 Блок-схема функции `drawHeader`

Блок-схема функции printStudentRow представлена на рисунке 2.14.

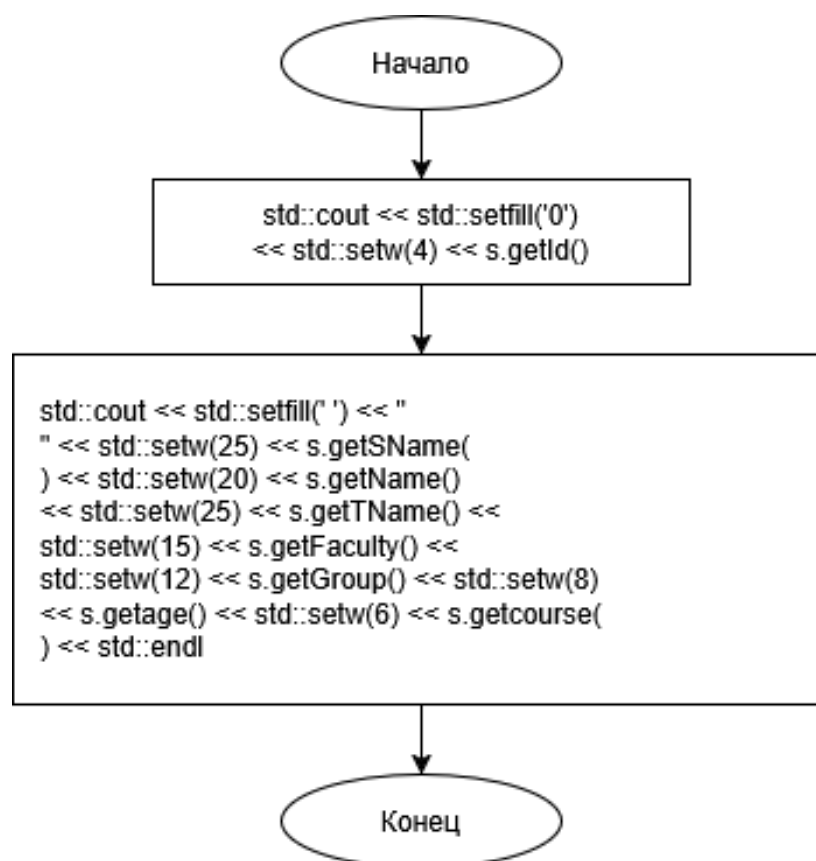


Рисунок 2.14 - Блок-схема функции printStudentRow

Блок-схема функции displayDatabase представлена на рисунке 2.15.

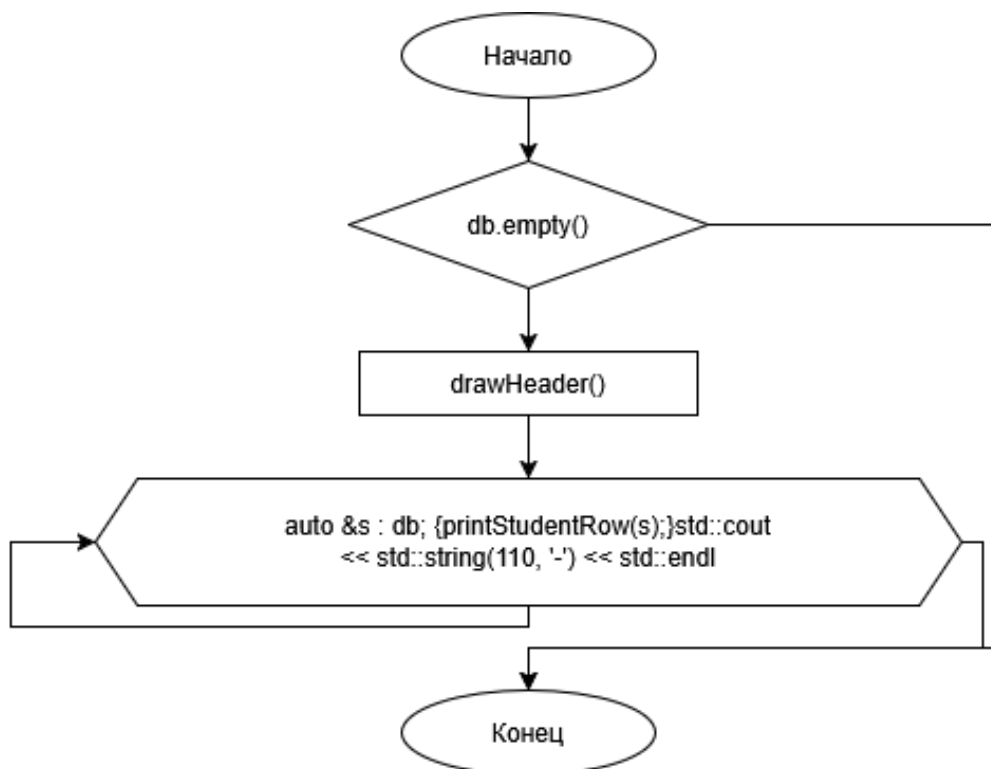


Рисунок 2.15 Блок-схема функции displayDatabase

Блок-схема функции searchMenu представлена на рисунке 2.16.

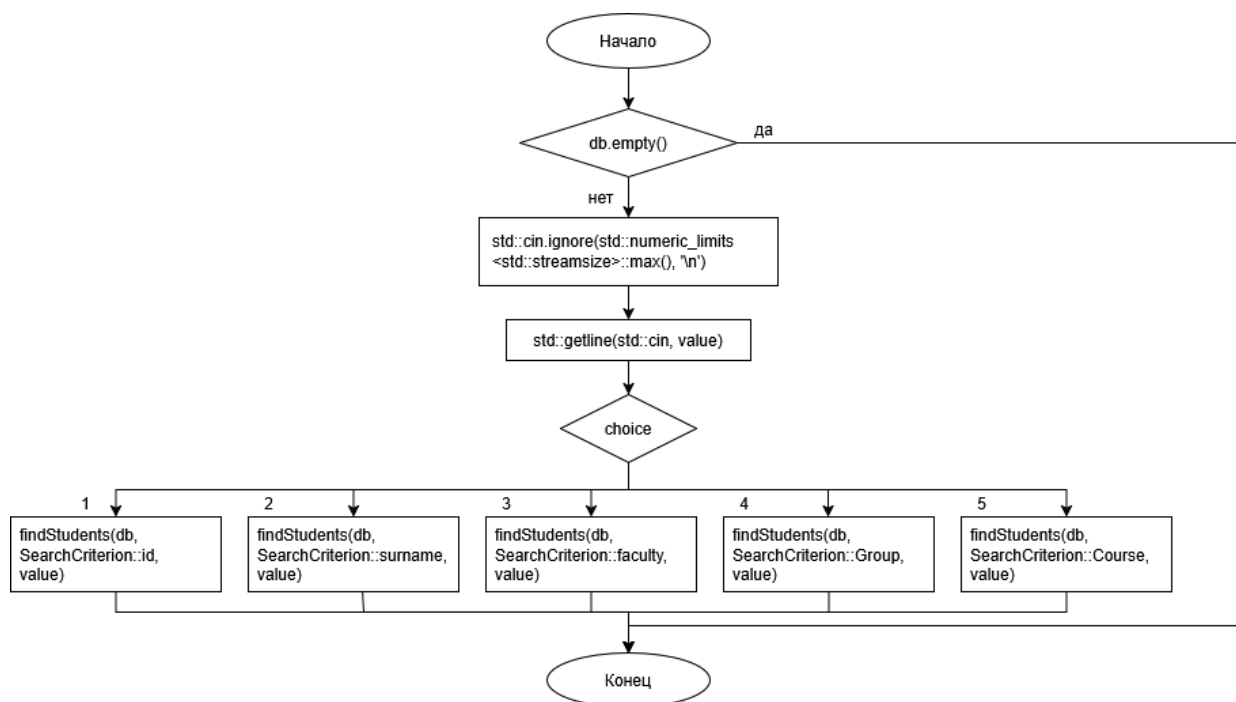


Рисунок 2.16 - Блок-схема функции searchMenu

Блок-схема функции calculateAverageAge представлена на рисунке 2.17.

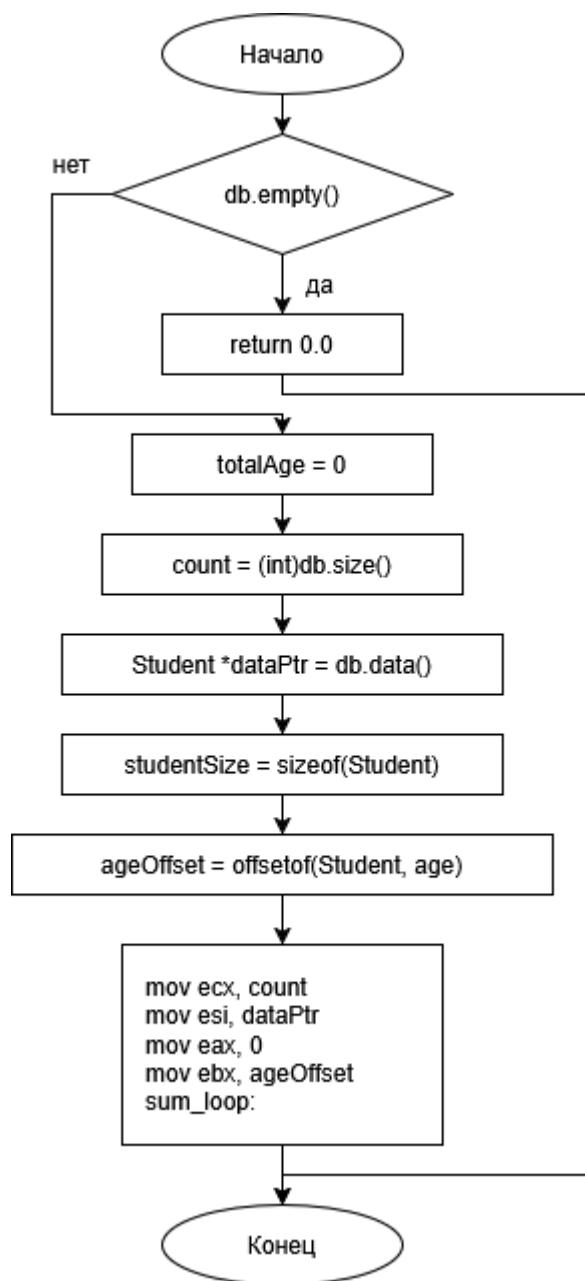


Рисунок 2.17 - Блок-схема функции calculateAverageAge

На рисунке 3 приведена иерархическая схема программы.

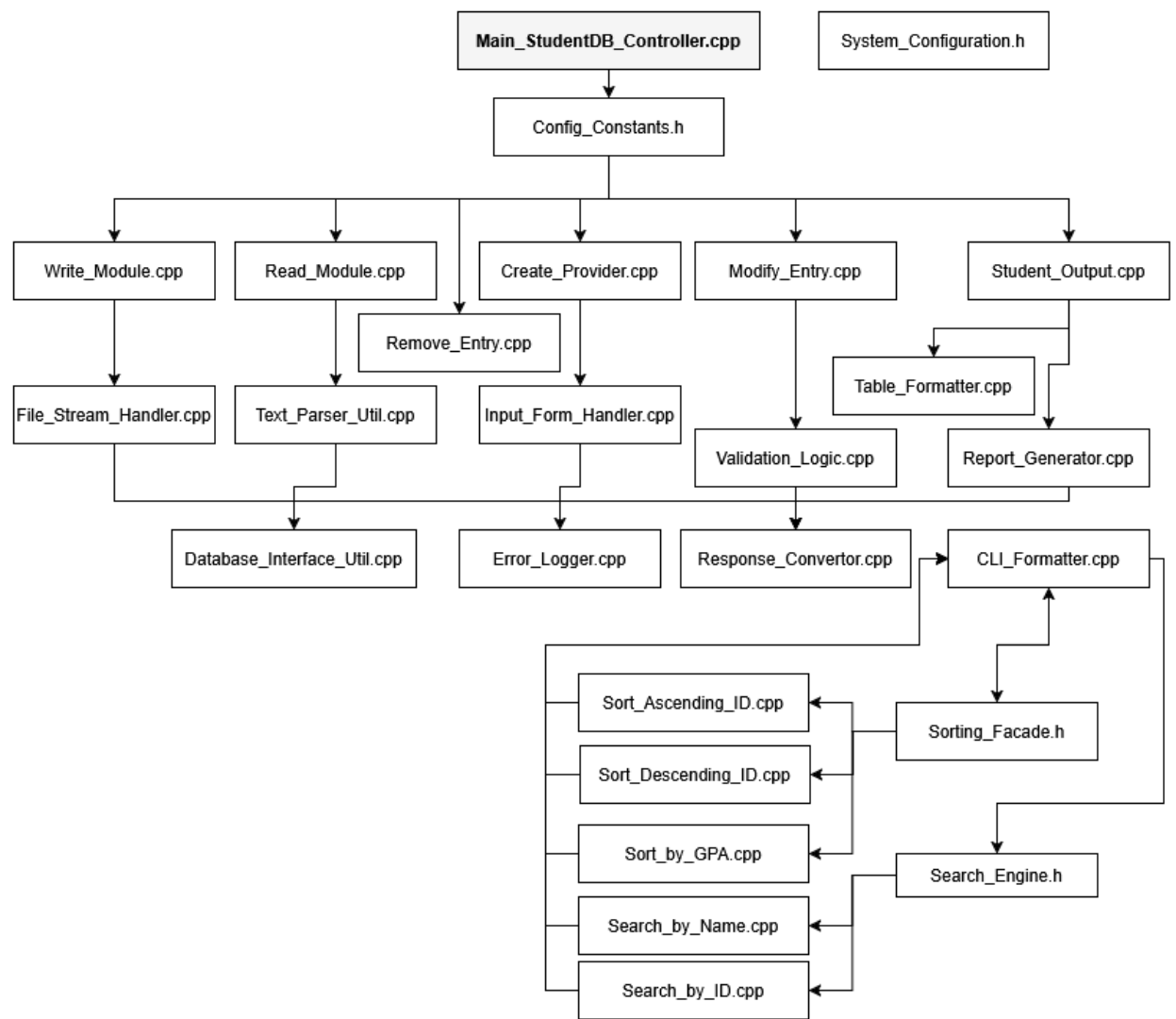


Рисунок 3 - иерархическая схема программы

Руководство пользователя

Программа curs.exe предназначена для хранения, обработки и анализа информации о студентах вуза. Программа имеет интуитивно понятный консольный интерфейс и поддерживает такие операции, как добавление новой записи о студенте в базу данных, удаление записи из базы данных, изменение существующих данных студента, поиск записей по различным фильтрам, сохранение базы данных в бинарный файл, импорт базы данных из файла, вывод отформатированной таблицы и низкоуровневый ассемблерный расчет среднего возраста.

Для открытия и вывода на экран текущего списка студентов необходимо в главном меню выбрать пункт (Рисунок 4.1) «1. Вывести список студентов». Программа считывает актуальные данные из оперативной памяти и построит структурированную таблицу на дисплее.

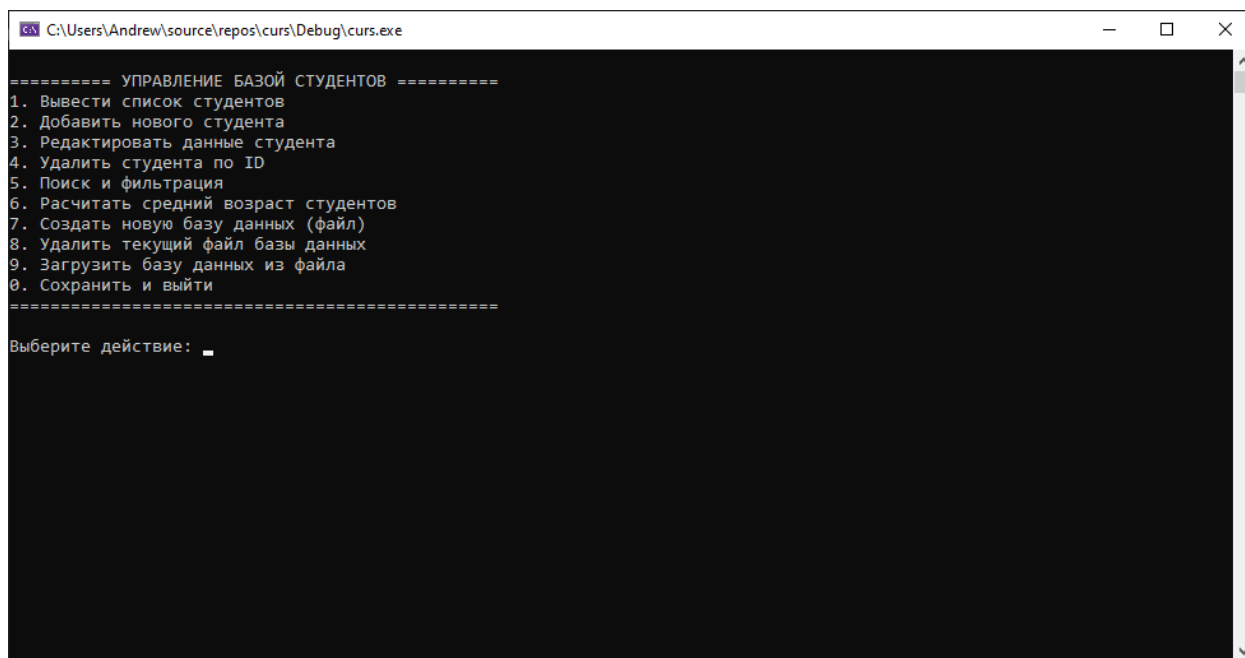


Рисунок 4.1 – Пункт вывести список студентов

Для добавления записи о студенте в базу данных необходимо выбрать пункт меню (Рисунок 4.2) «2. Добавить нового студента». Далее необходимо пошагово заполнить интерактивную форму, состоящую из следующих пунктов:

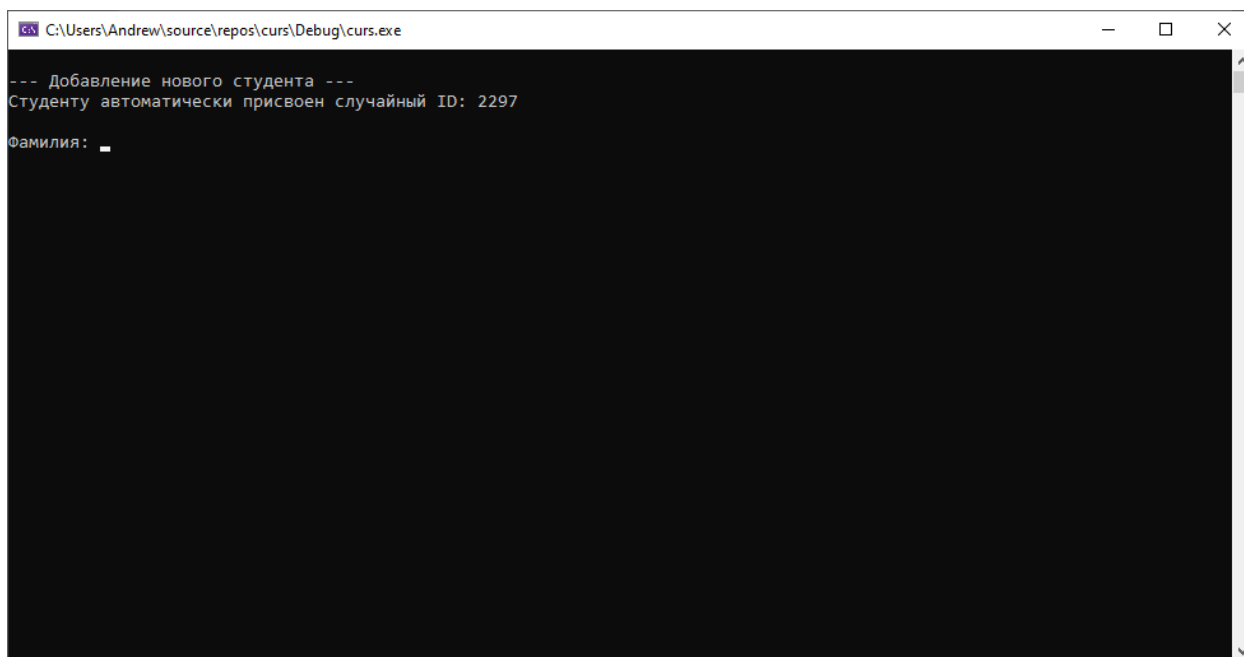


Рисунок 4.2 – Пункт Добавить студента

ввести фамилию студента, нажать клавишу Enter, ввести имя, нажать клавишу Enter, ввести отчество, нажать клавишу Enter, ввести название факультета, нажать клавишу Enter, ввести номер или шифр академической группы, нажать клавишу Enter, ввести возраст (целое число), нажать клавишу Enter, ввести номер курса (число от 1 до 6), нажать клавишу Enter.

Идентификатор (ID) генерируется программой автоматически с помощью датчика случайных чисел и проверяется на уникальность. Ввод данных строго контролируется встроенными модулями валидации (запрет на ввод цифр в текстовые поля и букв в числовые). В случае если данные введены корректно и запись успешно добавлена, на экране появится надпись «Студент успешно добавлен!» и данные автоматически продублируются в файл на жестком диске.

Для изменения существующей записи необходимо выбрать пункт главного меню (Рисунок 4.3) «3. Редактировать данные студента». Далее необходимо ввести ID изменяемой записи и ввести новые данные для анкеты студента, запо(лнив форму:

```
C:\Users\Andrew\source\repos\curs\Debug\curs.exe

--- Редактирование данных студента ---
Введите ID студента для редактирования: 0003

Студент найден! Текущие данные:
ФИО: Волковец Андрей Сергеевич
Факультет: ФВТ, Группа: 258884
Возраст: 18, Курс: 1

--- Введите новые данные ---
Новая Фамилия: Волковец
Новое Имя: Сергей
Новое Отчество: Андреевич
Новый Факультет: ФВТ
Новая Группа: 25888;
Новый Возраст: 18
Новый Курс: 1

Данные студента успешно обновлены и сохранены в файл!

Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.3 – Пункт Редактировать данные студента

Ввести новую фамилию, нажать клавишу Enter, ввести имя, нажать клавишу Enter, ввести отчество, нажать клавишу Enter, ввести факультет, нажать клавишу Enter, ввести новую академическую группу, нажать клавишу Enter, ввести возраст, нажать клавишу Enter, ввести номер курса и нажать клавишу Enter. Изменение уникального ID заблокировано для сохранения целостности связей. В случае если обновленные параметры прошли валидацию, на экране появится надпись «Данные студента успешно обновлены и сохранены в файл!».

Для удаления записи студента из базы данных необходимо в главном меню (Рисунок 4.4) выбрать пункт «3. Удалить студента по ID». Далее программа потребует ввести уникальный четырехзначный идентификатор удаляемой записи и нажать клавишу Enter. Система выполнит поиск в динамическом векторе и при обнаружении совпадения произведет физическое удаление элемента из оперативной памяти с последующим обновлением бинарного файла. В случае если операция прошла без ошибок, на экране появится надпись «Запись успешно удалена!», в противном случае программа выведет сообщение «Студент с данным ID не найден».

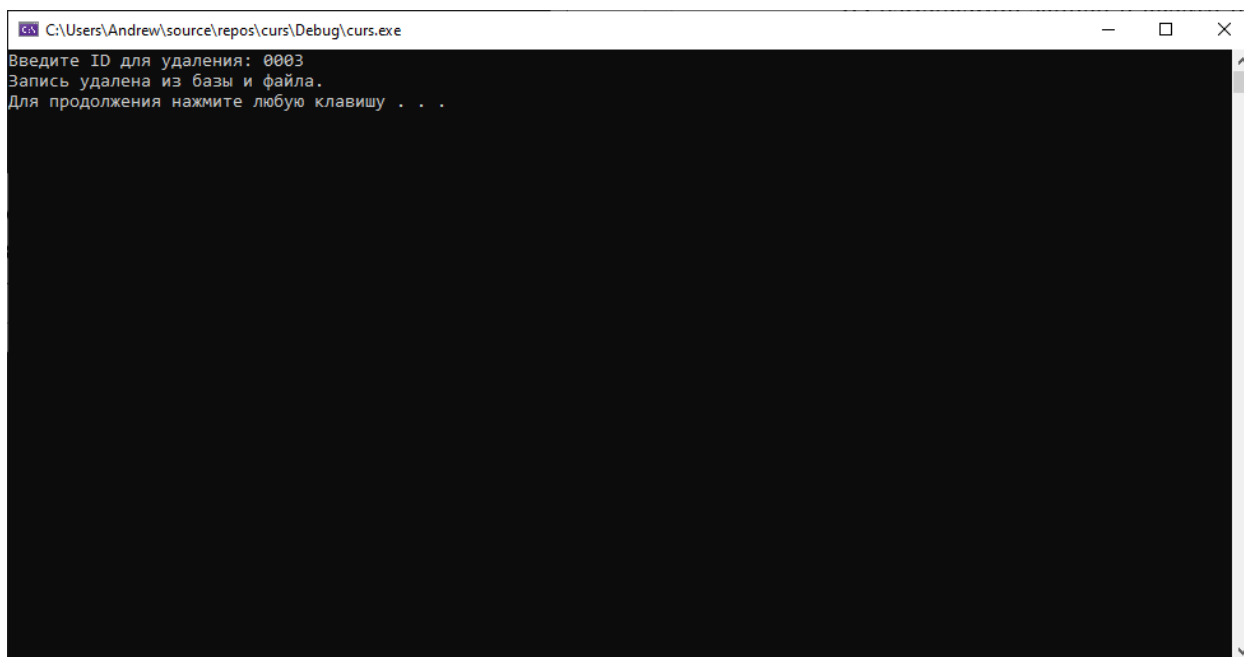


Рисунок 4.4 – Пункт Удалить запись студента

Для того чтобы сохранить текущую базу данных в новый сторонний файл, необходимо выбрать пункт меню (Рисунок 4.5) «7. Создать новую базу данных (файл)». Далее ввести имя файла с расширением .dat, в который будет записана база данных студентов. Если файл с таким именем уже существует, то программа перезапишет информацию в нем актуальным вектором структур. В случае если операция прошла без ошибок, на экране появится надпись «База данных успешно создана/сохранена!».

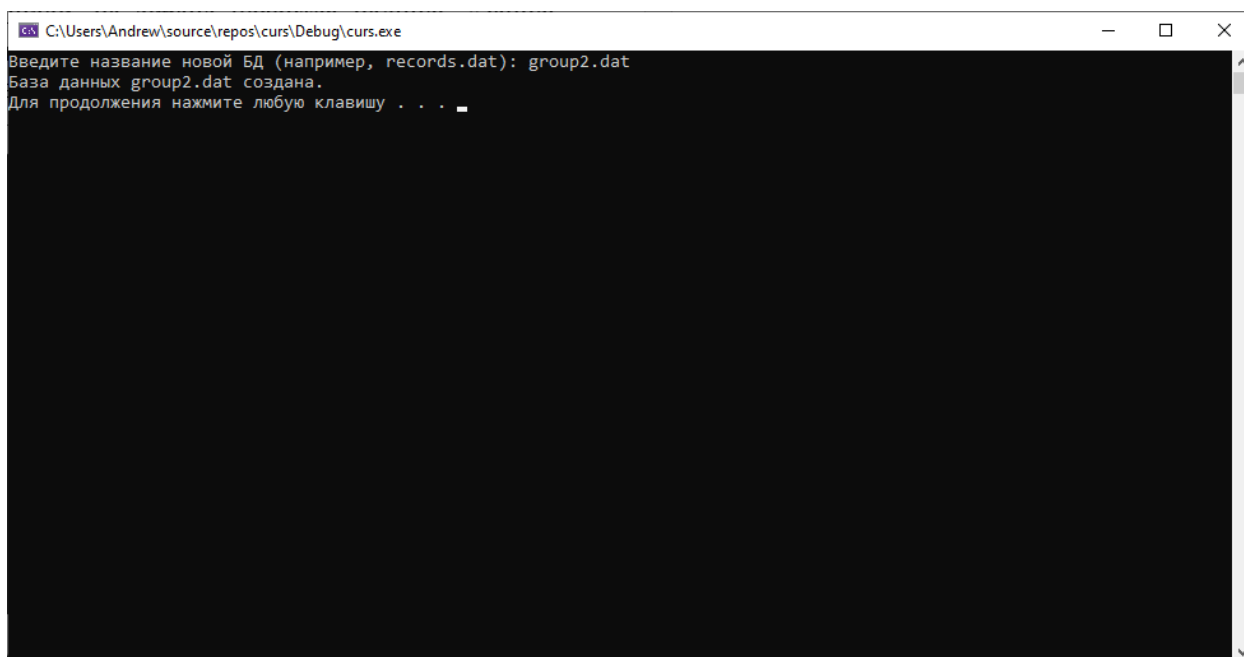


Рисунок 4.5 – Пункт Создать новую базу данных

Для того чтобы импортировать базу данных из файла, необходимо выбрать пункт главного меню (Рисунок 4.6) «9. Загрузить базу данных из файла». Далее следует ввести имя файла (например, group2.dat), из которого произойдет побайтовое чтение данных, и нажать клавишу Enter. Текущий список студентов в оперативной памяти будет полностью заменен на новый, считанный из указанного файла. В случае если файл отсутствует на диске, программа выведет сообщение «Ошибка: Файл не найден! Загрузка отменена.». В случае успешного считывания на экране появится надпись «База данных успешно загружена» и отобразится количество импортированных студентов.

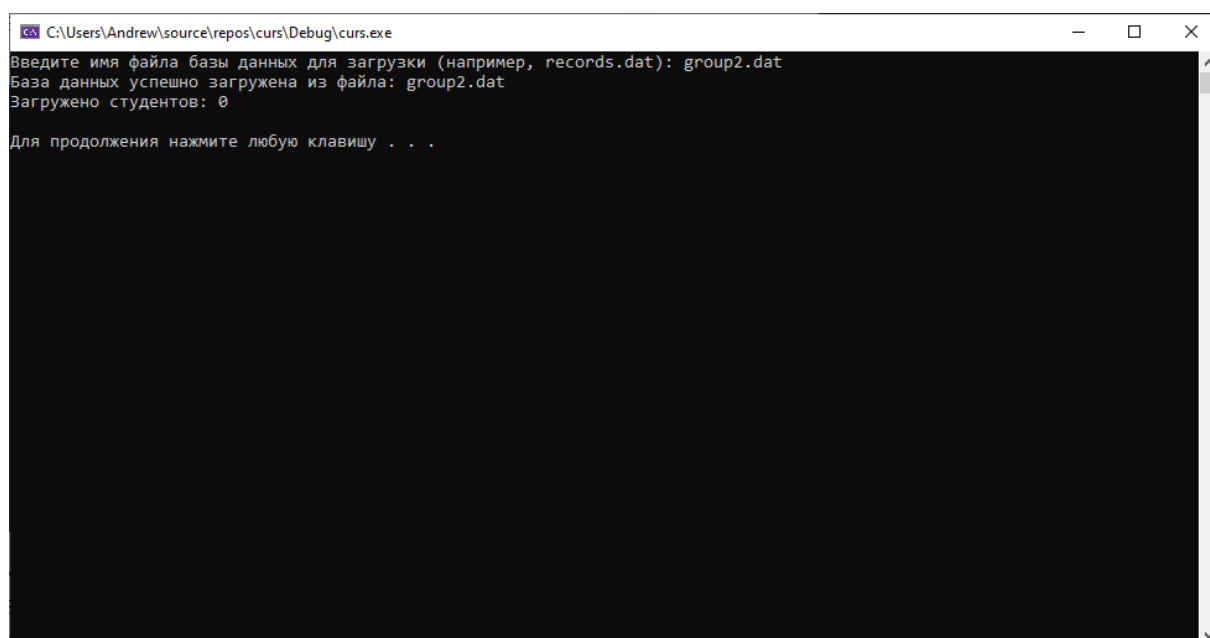


Рисунок 4.6 – Пункт Загрузить базу данных из файла

Для того чтобы записи были выведены в отфильтрованном или упорядоченном виде, необходимо в главном меню (Рисунок 4.7) выбрать пункт «Поиск и фильтрация». В открывшемся интерактивном подменю пользователь может выбрать интересующий его критерий выборки студентов. Поиск и вывод осуществляются по пяти основным параметрам: «5. Поиск по ID», «Поиск по Фамилии», «Фильтр по Факультету», «Фильтр по Группе», «Фильтр по Курсу».

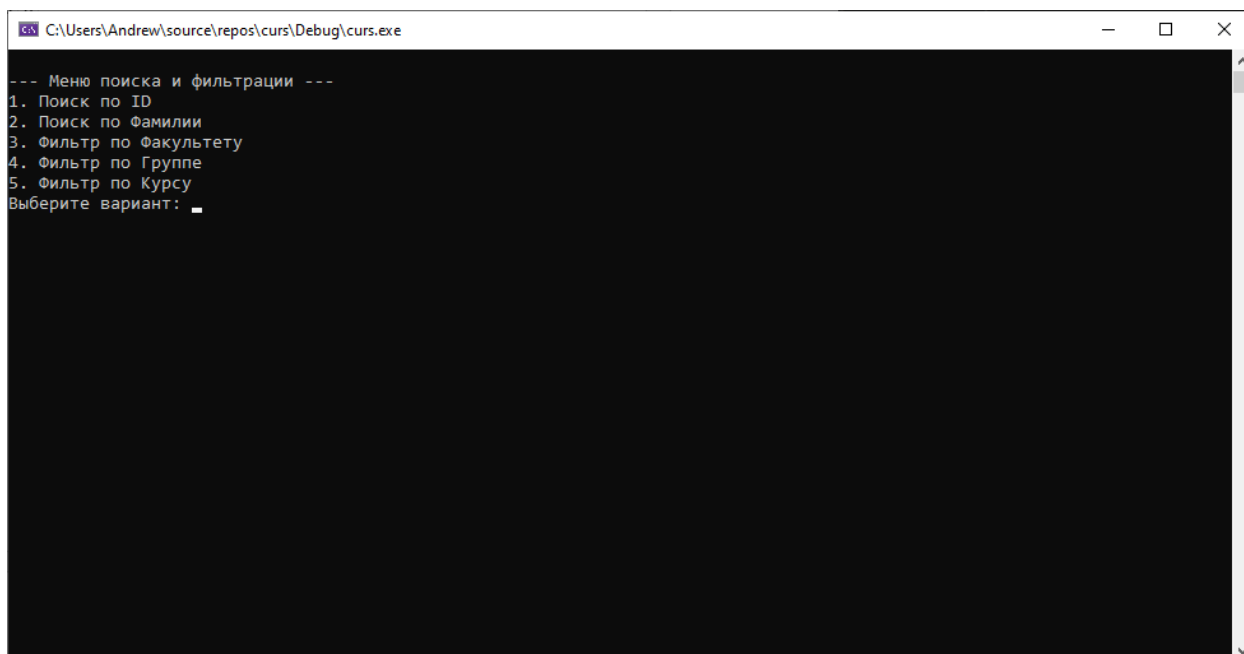


Рисунок 4.7 – Пункт Поиск и Фильтрация

Поиск по ID предназначен для точечного нахождения конкретной записи студента. Пользователю следует ввести четырехзначный идентификатор искомой записи и нажать клавишу Enter (Рисунок 4.8). В случае если студент с таким уникальным номером находится в векторе базы данных, программа выведет его персональную анкету на экран в виде строки таблицы. В противном случае на дисплее отобразится ошибка «Студент с данным ID не найден».

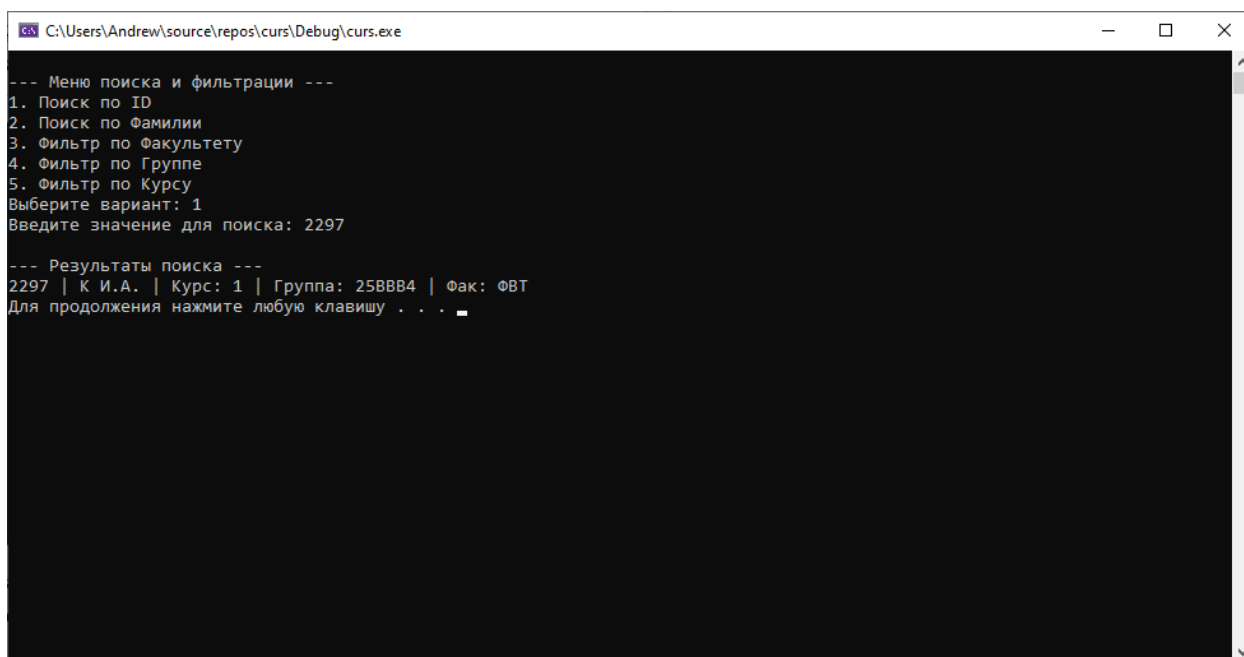


Рисунок 4.8 – Пункт Поиск и Фильтрация

Поиск по Фамилии предназначен для выборки студентов по их фамилиям (или их частям). Пользователь может ввести как полную фамилию студента, так и ее начальную часть, после чего необходимо нажать клавишу Enter (Рисунок 4.9). Если записи с подходящей фамилией присутствуют в базе данных, то на экран выведется отфильтрованная таблица студентов, в ином случае программа выдаст ошибку «База данных пуста» или сообщит, что совпадений не обнаружено.

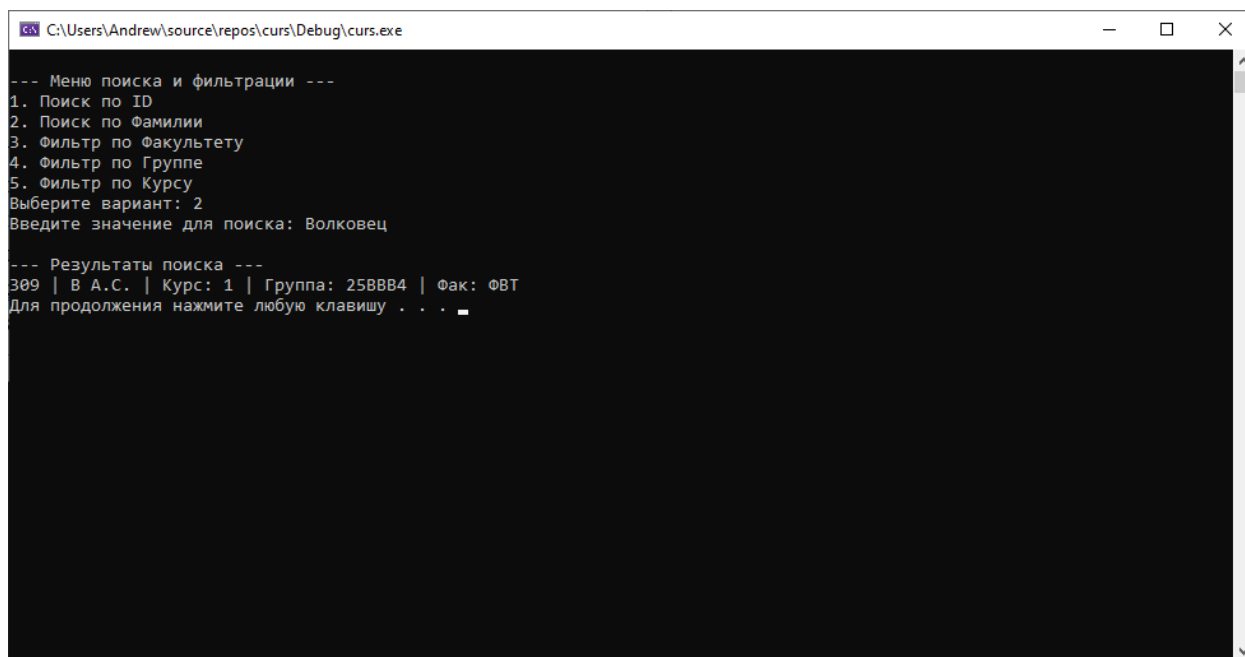
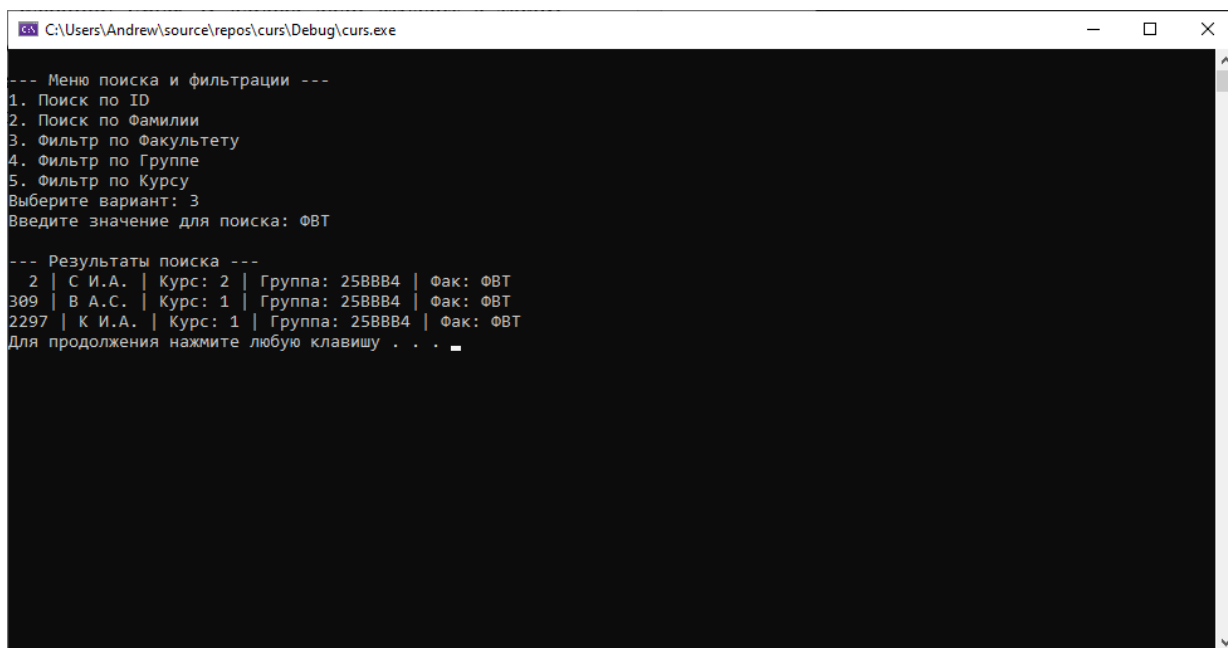


Рисунок 4.9 – Пункт Поиск и Фильтрация

Фильтр по Факультету необходим для группировки записей по конкретному подразделению вуза. Пользователь может ввести как полное название факультета, так и его сокращенный вариант (начальную часть строки), далее нужно нажать клавишу Enter (Рисунок 4.10). Если студенты данного факультета внесены в базу данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста» либо сообщит об отсутствии результатов по данному критерию.



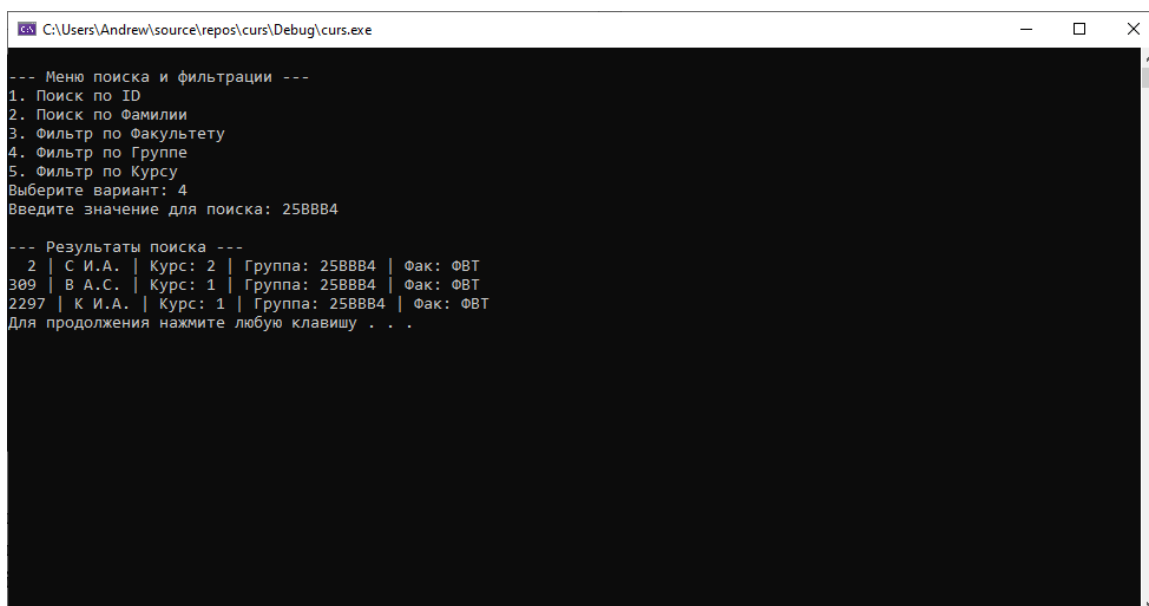
```
C:\Users\Andrew\source\repos\curs\Debug\curs.exe

--- Меню поиска и фильтрации ---
1. Поиск по ID
2. Поиск по Фамилии
3. Фильтр по Факультету
4. Фильтр по Группе
5. Фильтр по Курсу
Выберите вариант: 3
Введите значение для поиска: ФВТ

--- Результаты поиска ---
  2 | С И.А. | Курс: 2 | Группа: 258884 | Фак: ФВТ
309 | В А.С. | Курс: 1 | Группа: 258884 | Фак: ФВТ
2297 | К И.А. | Курс: 1 | Группа: 258884 | Фак: ФВТ
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.10 – Пункт Поиск и Фильтрация

Фильтр по Группе предназначен для поиска и отображения студентов, обучающихся в одном академическом коллективе. Пользователю необходимо ввести точное наименование или шифр группы с клавиатуры и нажать клавишу Enter (Рисунок 4.11). Если записи с данной группой присутствуют в динамической памяти, то на экран выведется таблица состава этой группы, в противном случае программа выдаст ошибку «База данных пуста» или сообщит, что указанная группа не найдена.



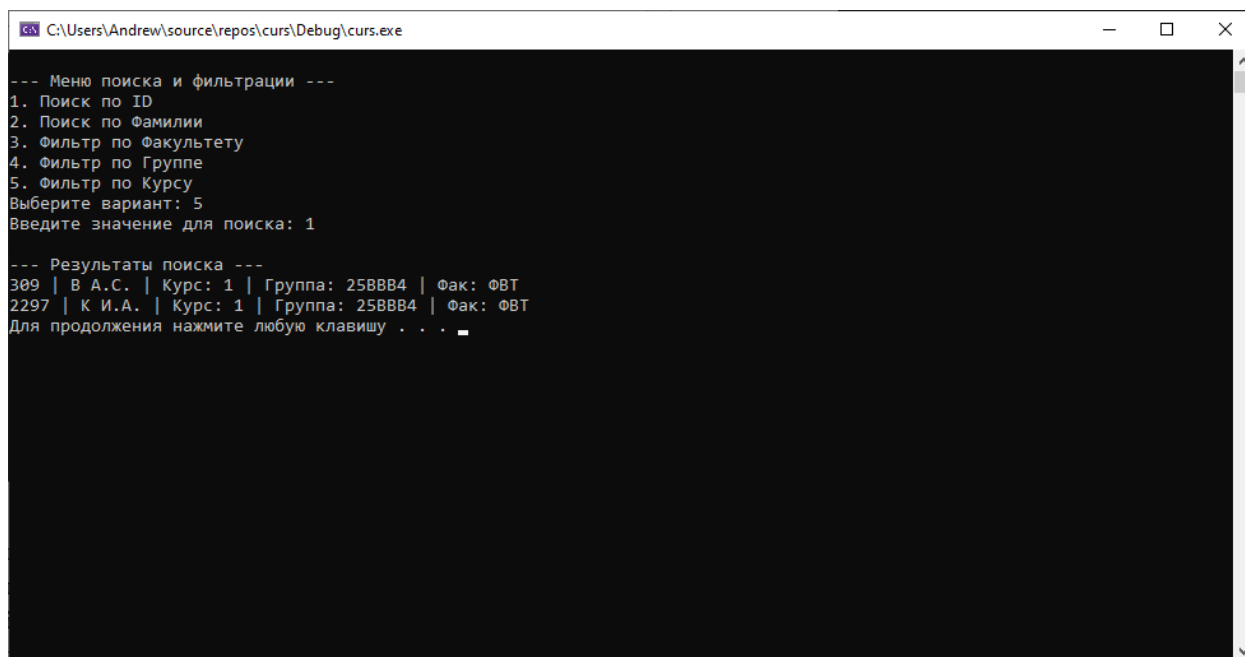
```
C:\Users\Andrew\source\repos\curs\Debug\curs.exe

--- Меню поиска и фильтрации ---
1. Поиск по ID
2. Поиск по Фамилии
3. Фильтр по Факультету
4. Фильтр по Группе
5. Фильтр по Курсу
Выберите вариант: 4
Введите значение для поиска: 258884

--- Результаты поиска ---
  2 | С И.А. | Курс: 2 | Группа: 258884 | Фак: ФВТ
309 | В А.С. | Курс: 1 | Группа: 258884 | Фак: ФВТ
2297 | К И.А. | Курс: 1 | Группа: 258884 | Фак: ФВТ
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.11 – Пункт Поиск и Фильтрация

Фильтр по Курсу предназначен для быстрого вывода студентов определенного года обучения. Пользователю необходимо ввести числовое значение курса (в диапазоне от 1 до 6) и нажать клавишу Enter (Рисунок 4.12). Программа выполнит циклическое сканирование базы данных и выведет на экран аккуратную таблицу студентов выбранного курса. Если на данном курсе никто не зарегистрирован или база данных пуста, на экране появится соответствующее предупреждение.



```
C:\Users\Andrew\source\repos\curs\Debug\curs.exe

--- Меню поиска и фильтрации ---
1. Поиск по ID
2. Поиск по Фамилии
3. Фильтр по Факультету
4. Фильтр по Группе
5. Фильтр по Курсу
Выберите вариант: 5
Введите значение для поиска: 1

--- Результаты поиска ---
309 | В.А.С. | Курс: 1 | Группа: 2588В4 | Фак: ФВТ
2297 | К.И.А. | Курс: 1 | Группа: 2588В4 | Фак: ФВТ
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.12– Пункт Поиск и Фильтрация

Для проведения расчетно-аналитической операции необходимо в главном меню (Рисунок 4.13) выбрать пункт «6. Рассчитать средний возраст студентов» и нажать клавишу Enter. Данный модуль предназначен для вычисления среднего арифметического возраста всех студентов, зарегистрированных в текущей базе данных, с использованием оптимизированных низкоуровневых вычислений.

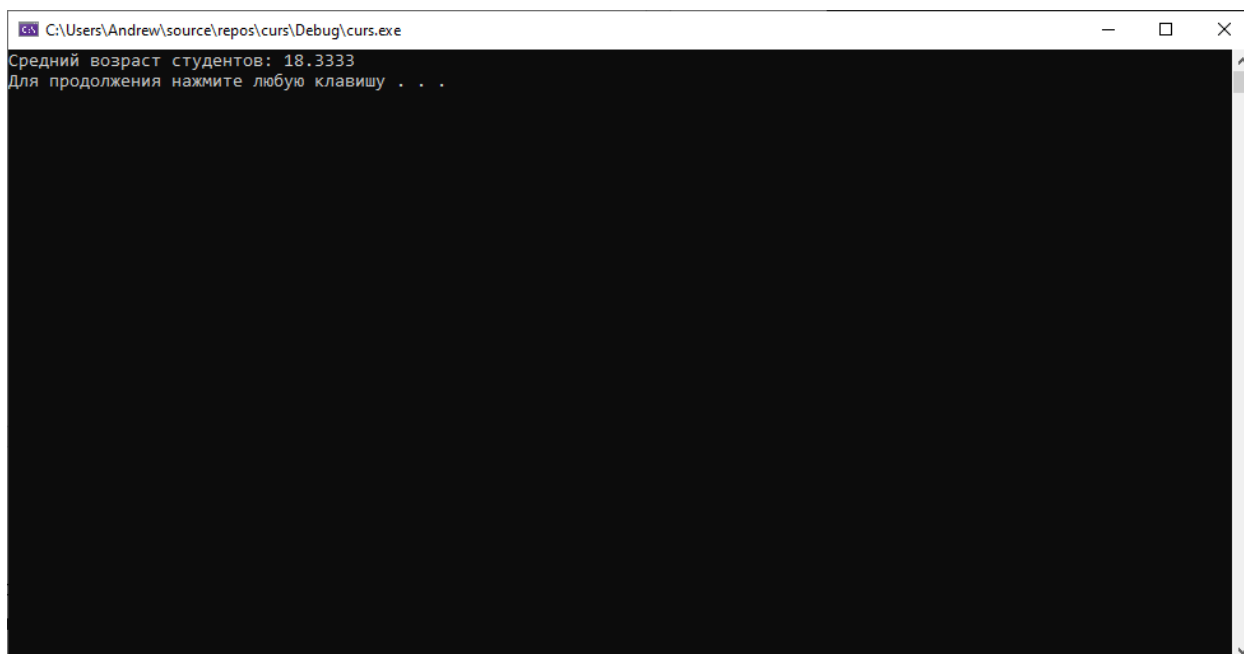


Рисунок 4.13 – Пункт Рассчитать средний возраст студентов

После активации пункта программа проверяет динамический вектор в оперативной памяти на наличие записей. Если база данных пуста, на экран выводится соответствующее предупреждение, и выполнение операции прерывается. Если в базе присутствует хотя бы одна анкета, управление передается специализированной функции, внутри которой реализован блок прямой ассемблерной вставки (`__asm` для архитектуры x86).

Для полного удаления рабочей базы данных с постоянного носителя информации необходимо в главном меню (Рисунок 4.14) выбрать пункт «8. Удалить текущий файл базы данных» (или аналогичный пункт удаления файла) и нажать клавишу Enter. Данная операция предназначена для быстрой очистки дискового пространства и сброса текущей рабочей сессии.

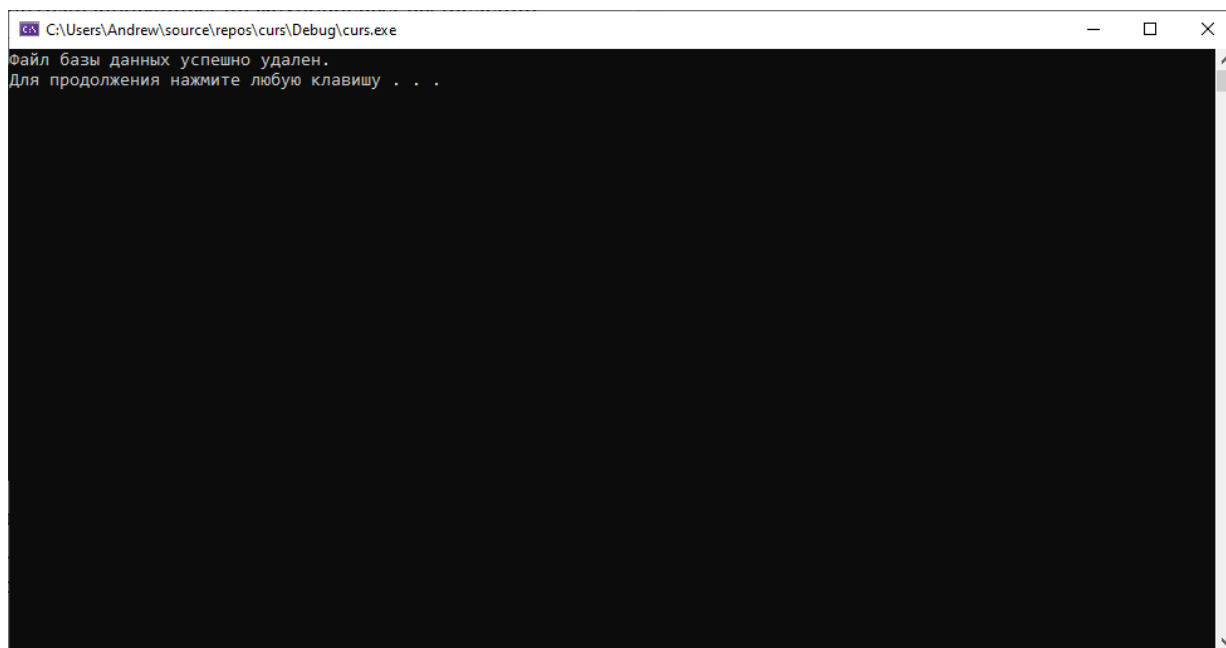


Рисунок 4.14 – Пункт Удаление файла текущей базы данных

Для завершения работы с приложением необходимо выбрать пункт главного меню «0. Сохранить и выйти» и нажать клавишу Enter. Программа закроет все файловые потоки, гарантированно запишет финальное состояние вектора студентов в рабочий файл на жестком диске и завершит выполнение.

Заключение

При выполнении данной курсовой работы были успешно закреплены и углублены практические навыки проектирования и разработки сложных многомодульных программных комплексов на базе языка высокого уровня C++. В ходе проектирования информационной системы «Управление базой данных студентов» была детально освоена организация побайтового ввода-вывода при работе с бинарными файлами произвольного доступа (.dat), изучены механизмы динамического управления оперативной памятью посредством стандартных контейнеров (std::vector), а также функции форматированного вывода и очистки консольного интерфейса.

Особое внимание в рамках проекта было уделено интеграции разнородных языковых сред: разработка и внедрение расчетно-аналитического модуля на базе низкоуровневой ассемблерной вставки (__asm) позволили освоить принципы прямой адресации памяти, работы с регистрами общего назначения процессора x86 и оптимизации математических вычислений. Кроме того, в процессе работы были получены фундаментальные навыки сквозной отладки, валидации пользовательского ввода и комплексного тестирования программного обеспечения в среде разработки Microsoft Visual Studio.

Созданное приложение полностью отвечает поставленным техническим требованиям, обеспечивая надежное хранение, интерактивное редактирование, многокритериальную фильтрацию и удаление анкетных данных студентов с гарантированным сохранением целостности информации на физическом носителе.

В качестве направлений для дальнейшей модернизации и развития программного продукта можно выделить переход от текстового консольного интерфейса (CLI) к полноценному графическому пользовательскому интерфейсу (GUI). Это позволит существенно повысить эргономику приложения, внедрить интерактивные элементы управления (окна, кнопки, списки) и обеспечить полноценную навигацию с помощью мыши, сделав

взаимодействие пользователя с базой данных студентов более комфортным и интуитивным.

Список используемых источников

1. Керниган, Б. Язык программирования С / Б. Керниган, Д. Ритчи. – 3-е изд. – 2015. – 304 с.
2. Прата С. Язык программирования С++/С. Прата. – 6-е изд. 2019. – 1248 с.
3. Тюгашев А.А. Языки программирования : учебное пособие.\ А.А. Тюгашев. – Санкт-Петербург : Питер, 2018 - 336 с.
4. Аблязов, Р. Программирование на ассемблере на платформе x86-64 / Р. Аблязов – Москва : ДМК Пресс, 2016 – 306 с.