

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

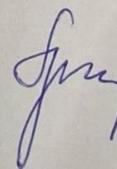
ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

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

Выполнил:
студент группы 23BBB1 Видяев А.А.

Принял:
к.т.н., доцент Токарев А.Н.

хор
6.06.24



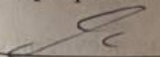
Пенза, 2024

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 Видяеву Алексею Анимовичу Группа 2180801

Тема проекта: *Разработка программы сложной структуры методом нисходящего программирования.*

Исходные данные (технические требования) на проектирование

*Библиотечное - база данных "студенты".
Добавление, удаление, редактирование, поиск, сортировка
информации. Прочтение данных в любое время.*

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.

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 "29" 02 2024

Дата защиты проекта " " "

Руководитель _____

Задание получил "29" февраля 2024 г.

Студент Висляев _____

Содержание

Введение.....	5
Постановка задачи.....	6
Выбор решения.....	7
Описание разработки программы.....	8
Отладка и тестирование.....	8
Описание программы.....	8
Руководство пользователя.....	15
Заключение.....	21
Список источников.....	используемых 28
Приложение АЛистинг программы.....	23
Приложение БРезультат работы программы.....	36

Введение

Базы данных играют ключевую роль в современном мире информационных технологий, являясь основой для хранения, организации и управления огромными объемами данных. Они широко применяются в различных сферах, включая бизнес, образование, здравоохранение, государственное управление и многие другие. Понимание принципов и методов работы с базами данных является необходимым навыком для специалистов в области информационных технологий.

Цель данной курсовой работы заключается в ознакомлении студентов с основами проектирования, создания и использования баз данных. В процессе выполнения работы студенты приобретут практические навыки работы с различными типами баз данных, а также узнают о современных подходах к их управлению и обеспечению безопасности.

Основными задачами курсовой работы являются:

1. Изучение основных концепций и технологий баз данных.
2. Освоение методов проектирования и создания баз данных.
3. Практическое применение полученных знаний для разработки собственной базы данных.
4. Исследование современных трендов и технологий в области баз данных.

Результаты выполненной курсовой работы будут иметь практическую значимость для студентов, так как они смогут применить полученные знания и навыки в своей будущей профессиональной деятельности. В частности, они смогут участвовать в проектировании, разработке и поддержке баз данных в организациях различного масштаба, что является востребованным навыком на рынке труда в сфере информационных технологий.

Таким образом, данная курсовая работа не только позволит студентам овладеть ключевыми компетенциями в области баз данных, но и подготовит их к успешной карьере в сфере информационных технологий.

Постановка задачи

Необходимо разработать программу для создания и работы с базой данных 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 так же позволит легко расширять функционал программы.

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

Режим работы видеосистемы – текстовый. Данный режим работы очень прост в использовании как для пользователя, так и для программиста. Разработка приложений в текстовом режиме очень быстра и легка, так как нет необходимости создавать графическое окно и отслеживать действия пользователя, при этом код можно сделать на любых технологиях с любым уровнем сложности.

Устройство ввод-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 необходимо для того, чтобы пользователь, запустивший программу, смог получить достаточную информацию о ней.

Выбор решения

При запуске программы открывается меню состоящие из 8 пунктов:

- 1) Вывести всех студентов. Программа выводит БД.
- 2) Сортировать студентов по фамилии. Программа сортирует БД по фамилии.
- 3) Найти студента по фамилии. Пользователь вводит фамилию программа ищет эту фамилию в БД и выводит строку с этой фамилией.
- 4) Добавить студента. Программа запрашивает данные о новом студенте и добавляет его данные в БД.
- 5) Удалить студента. Пользователь вводит фамилию, программа находит строку с этой фамилией и удаляет строку содержащую её.
- 6) Редактировать данные о студенте. Программа запрашивает фамилию студента а потом новые данные для этого студента и заносит в БД.
- 7) Сохранить данные в файл. Программа сохраняет БД в файл.
- 8) Выход. Выход из программы.

Навигация в меню программы осуществляется с помощью клавиатуры, вводом цифр от 0 до 7.

Описание разработки программы

Разработанная программа состоит из одного модуля:

- 1) main.cpp – основной файл программы. Меню программно

Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2022. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовалось несколько возможностей Visual Studio: трассировка, анализ содержимого переменных, точка останова.

Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено незначительное количество проблем, связанных с добавлением студента в БД.

Описание программы

При запуске программы происходит вывод меню, после чего пользователю необходимо выбрать пункт меню для дальнейшего использования базы данных. Выход из меню осуществляется после выбора пользователем нужного пункта меню. Описание состояний программы выполнено в таблице ниже.

Таблица 1 - Описание состояний программы

Клавиши, вызывающее событие	Действия пользователя	Действие программы
1, Enter	Выбран пункт «Вывести всех студентов»	Выводится БД
2, Enter	Выбран пункт «Сортировать студентов по фамилии»	Выводит отсортированную БД
3, Enter	Выбран пункт «Найти студента по фамилии»	Открывается диалог для поиска в БД
4, Enter	Выбран пункт «Добавить студента»	Открывается диалог добавления записи в БД
5, Enter	Выбран пункт «Удалить студента»	Открывается диалог удаления записи из БД

6, Enter	Выбран пункт «Редактировать данные о студенте»	Открывается диалог для редактирование данных в БД
7, Enter	Выбран пункт «Сохранить данные в файл»	БД сохраняется в файл
0, Enter	Выбран пункт «Выход»	Программа закрывается

Ниже на рисунке представлена схема данных.



Рисунок 1 - Схема данных

Далее приведены блок-схемы основных функций программы БД студенты

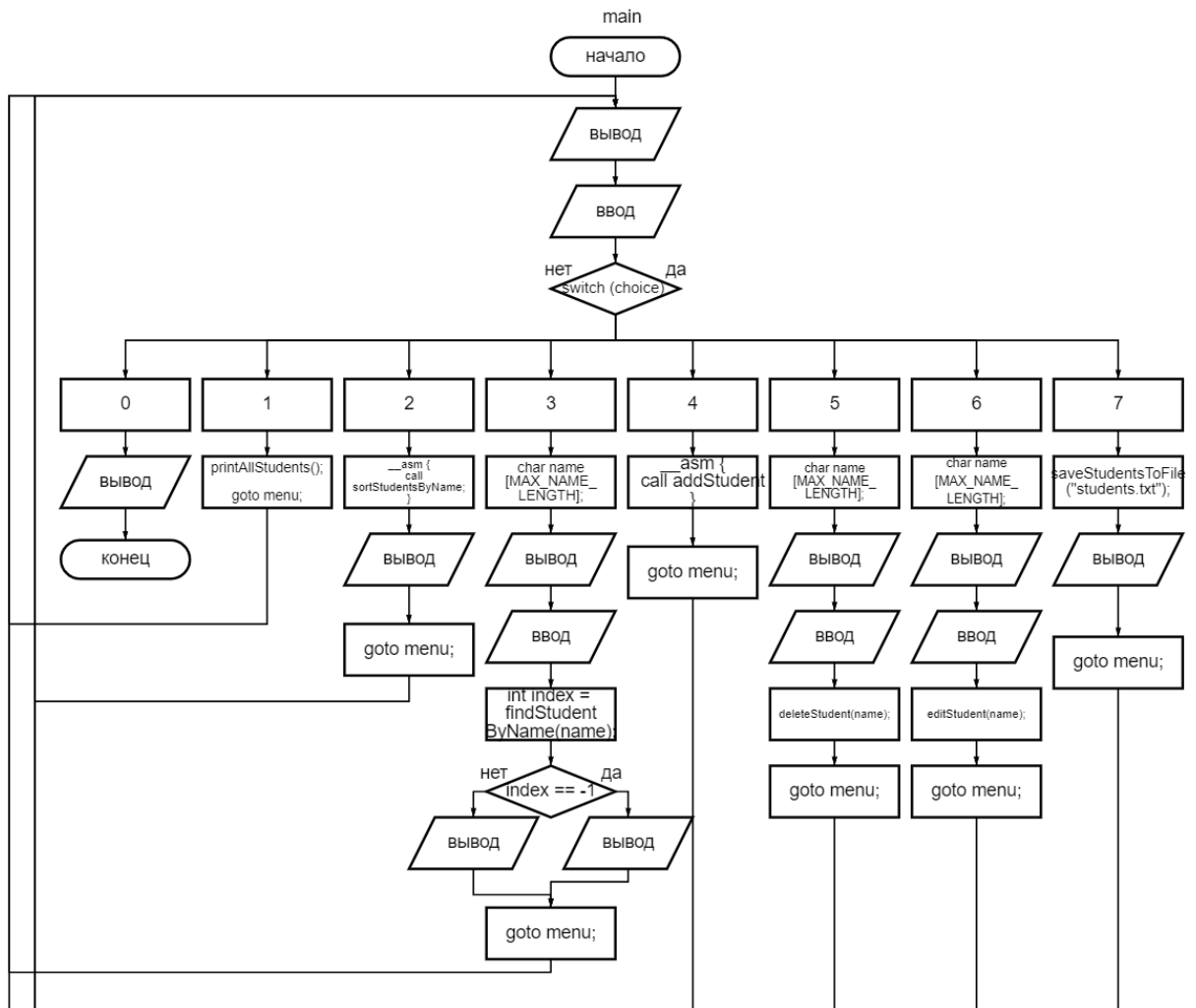


Рисунок 2 - Блок-схема функции main

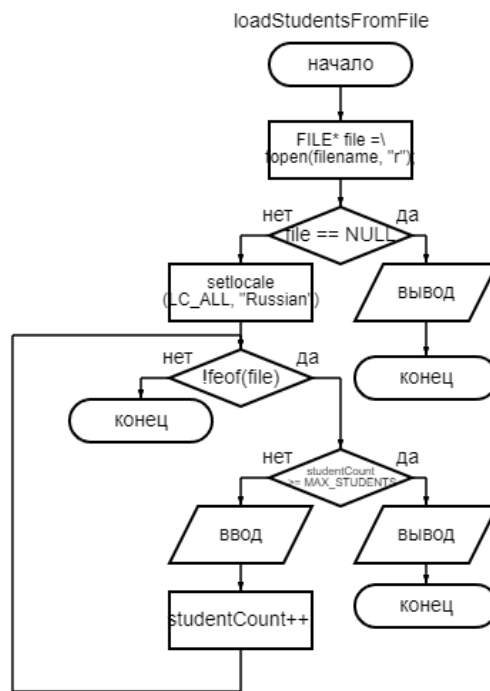


Рисунок 3 - Блок-схема функции loadStudentsFromFile

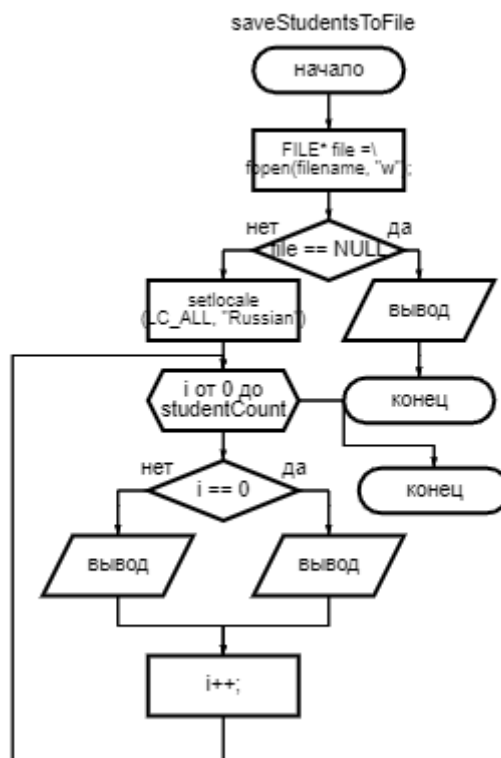


Рисунок 4 - Блок-схема функции saveStudentsToFile

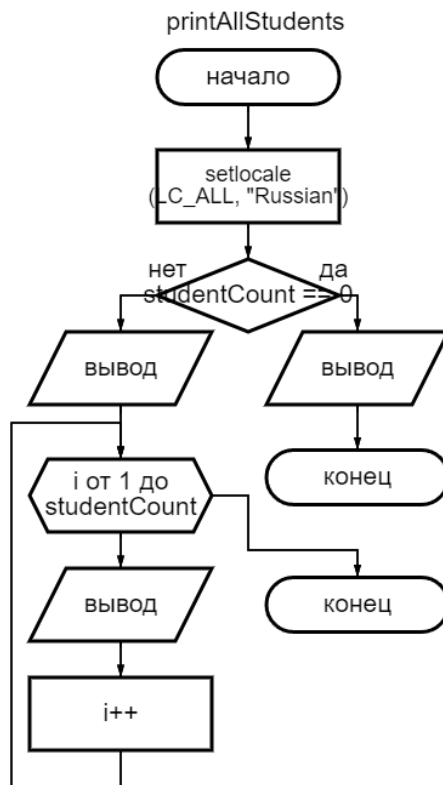


Рисунок5 - Блок-схема функции printAllStudents

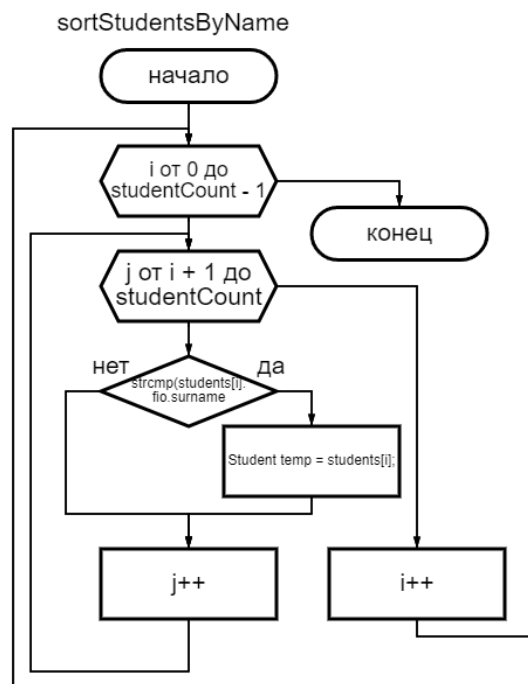


Рисунок 6 - Блок-схема функции sortStudentsByName

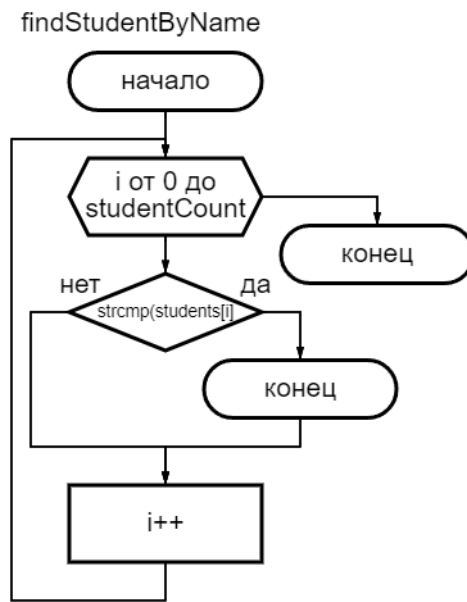


Рисунок 7 - Блок-схема функции findStudentByName

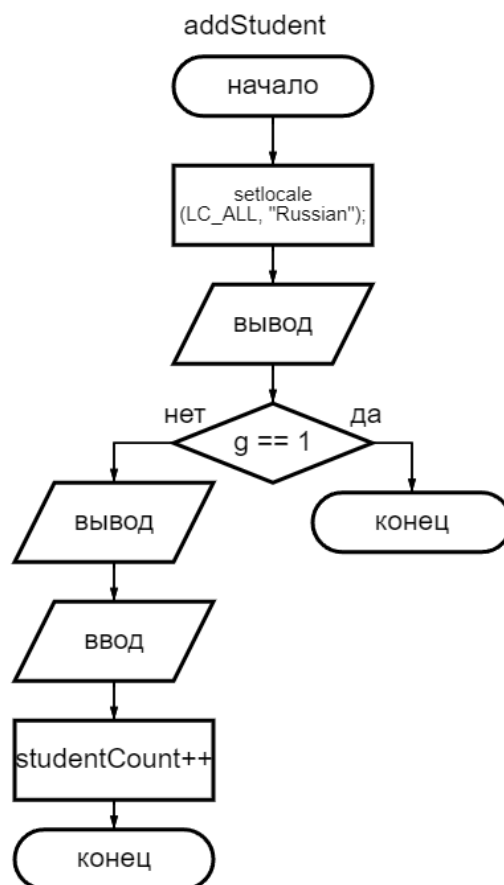


Рисунок 8 - Блок-схема функции addStudent

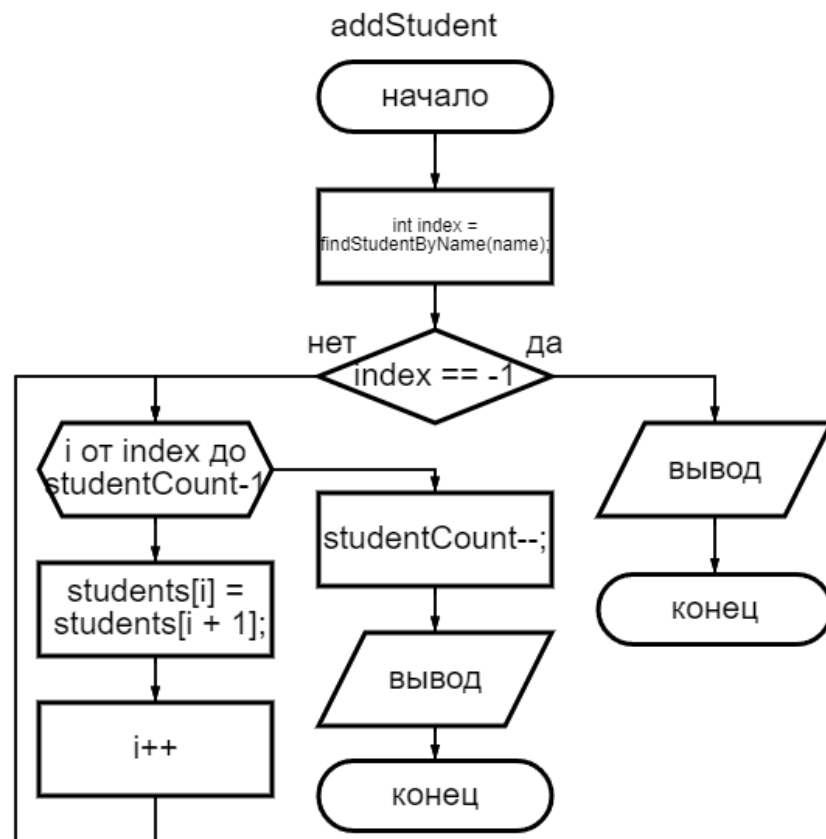


Рисунок 9 - Блок-схема функции addStudent

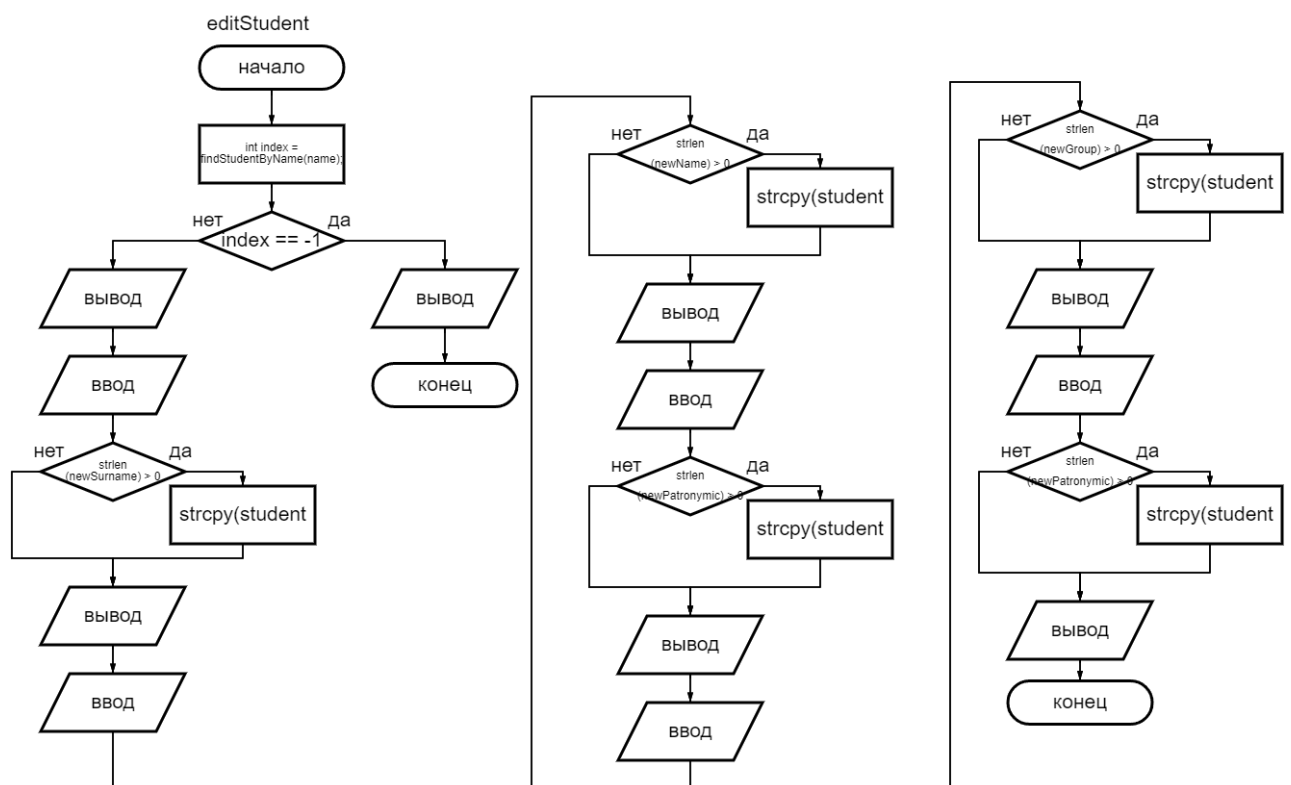


Рисунок 10 - Блок-схема функции editStudent

Руководство пользователя

Программа БД студенты предназначена для хранения информации о студентах. Программа имеет интуитивно понятный интерфейс и поддерживает такие операции, как добавление новой записи в базу данных, удаление записи из базы данных, изменение записи в базе данных, поиск записи в базе данных, сохранение текущей базы данных в файл, импорт базы данных из файла, сортировка базы данных.

Для вывода на экран текущей базы данных необходимо выбрать пункт меню «Вывести БД». В открывшемся меню будут выведены все записи текущей базы данных.

```
Ошибка открытия файла students.txt

--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: |
```

Рисунок 11 - Выбор пункта меню "Вывести всех студентов"

```
Ошибка открытия файла students.txt

--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 1
База данных пуста.
```

Рисунок 12 - Результат вывода пустой БД

```

--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 1
База данных по студентам:

```

Ф.И.О.			возраст	Группа
Фамилия	Имя	Отчество		
иван	иванов	иванович	1	19
ааааа	ааа	аа	аа	аа

Рисунок 13 - Результат вывода БД с записями

Для добавления записи в БД необходимо выбрать пункт меню «Добавить студента». Далее необходимо заполнить форму структуры из 5 пунктов: ввести название фамилию, нажать клавишу Enter, ввести имя, нажать клавишу Enter, ввести отчество, нажать клавишу Enter, ввести возраст, нажать клавишу Enter, ввести группу, нажать клавишу Enter. Если во время добавления записи не возникло ошибок, появится надпись «Студент добавлен.».

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 4
Введите фамилию студента: ааааа

Введите имя студента: ааа

Введите отчество студента: аа

Введите группу студента: аа

Введите возраст студента: аа

Студент добавлен.
```

Рисунок 14 - Добавление записей в БД

Для удаления записи из каталога необходимо выбрать пункт «Удалить студента». Далее необходимо ввести фамилию и нажать клавишу Enter. В случае, если не возникли ошибки, на экране появится надпись «Студент удалён.».

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 5
Введите фамилию студента для удаления: игнатовичевский
Студент удален.
```

Рисунок 15 - Удаление записи БД

Для изменения записи необходимо выбрать пункт меню «Редактировать данные о студенте». Далее необходимо ввести фамилию и ввести новые данные для этого студента. После ввода необходимо нажать Enter. Обновив данные, программа выведет надпись «Данные студента обновлены».

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 6
Введите фамилию студента для редактирования: иван
Введите новую фамилию студента (пустое для пропуска): ссс

Введите новое имя студента (пустое для пропуска): сс

Введите новое отчество студента (пустое для пропуска): сс

Введите новую группу студента (пустое для пропуска): сс

Введите новый возраст студента (пустое для пропуска): сс

Данные студента обновлены.
```

Рисунок 16 - Редактирование записей БД

Для того, чтобы сохранить текущую базу данных в файл, необходимо выбрать пункт меню «Сохранить данные в файл». В случае если не возникли ошибки, на экране появится надпись «Данные сохранены в файл students.txt».

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 7
Данные сохранены в файл students.txt
```

Рисунок 17 - Сохранение БД в файл

Для того, чтобы найти определенные записи в базе данных, необходимо выбрать пункт меню «Найти студента по фамилии». Далее ввести значение искомого элемента, после чего программа отпечатает на экран данные о студенте.

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 3
Введите фамилию студента: ссс
Студент найден:
|      ссс |          сс |          сс |          сс |          сс |
```

Рисунок 18 - Поиск записей в БД

Для выхода из программы необходимо выбрать пункт меню «Выход» и нажать клавишу Enter.

```
--- Меню ---  
1. Вывести всех студентов  
2. Сортировать студентов по фамилии  
3. Найти студента по фамилии  
4. Добавить студента  
5. Удалить студента  
6. Редактировать данные о студенте  
7. Сохранить данные в файл  
0. Выход  
Введите ваш выбор: 0  
Выход из программы.
```

Рисунок 19 - Выход из программы

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Была освоена работа с файлами, изучены функции работы с консолью и оперативной памятью. Также были получены основные навыки отладки и тестирования программ и программирования в среде Microsoft VisualStudio 2022 на языках C/C++ и Assembler.

В рамках курсовой работы была написана программа для работы с базой данных студенты. Она имеет все необходимые функции для работы с базами данных.

В дальнейшем программу можно улучшить добавлением графического интерфейса, что позволит улучшить навигацию в меню программы, добавлением возможности использования мыши, а так же обработки действий пользователей в оконном режиме.

Список используемых источников

1. К.Ирвин - Язык ассемблера для процессоров Intel
2. Н.Г.Голубь
Искусство_программирования_на_Ассемблере.Лекции_и_упр
ажнения
3. С.В.Зубков-Ассемблер_для_DOS,Windows_и_Unix

Приложение А

Листинг программы

LL.cpp

```
#define _CRT_SECURE_NO_WARNINGS

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <locale.h>
#include <windows.h>

#define MAX_STUDENTS 100
#define MAX_NAME_LENGTH 50
#define MAX_GROUP_LENGTH 10
#define MAX_AGE_LENGTH 10

char ender[] = { " +-----+-----+-----+-----+-----+\n" };
char table[] =
" +-----+-----+-----+-----+\n"
" |          Ф.И.О.          | |          |\n"
" |          | возраст | Группа |\n"
" |-----+-----+-----+-----+\n"
" | Фамилия | Имя | Отчество | |          |\n"
" |-----+-----+-----+-----+\n";
// Структура для хранения данных о студенте
typedef struct
{
    struct fio
    {
        char surname[MAX_NAME_LENGTH];
        char name[MAX_NAME_LENGTH];
        char patronymic[MAX_NAME_LENGTH];
    } fio;
    char group[MAX_GROUP_LENGTH];
    char age[MAX_AGE_LENGTH];
} Student;

// Массив для хранения данных о студентах
Student students[MAX_STUDENTS];
int studentCount = 0;

// Функция для загрузки данных из файла
void loadStudentsFromFile(const char* filename)
{
    FILE* file = fopen(filename, "r");
    if (file == NULL)
    {
        printf("Ошибка открытия файла %s\n", filename);
        return;
    }

    setlocale(LC_ALL, "Russian"); // Устанавливаем русскую локаль

    while (!feof(file))
    {
        if (studentCount >= MAX_STUDENTS)
        {
            printf("Превышен максимальный размер базы данных.\n");
            break;
        }
        // ... (the rest of the function code is missing in the image)
    }
}
```

```

break;
}
// Используем fscanf, чтобы прочитать строку из файла, и fgets для ввода с клавиатуры.
fscanf(file, "%s %s %s %s %s", students[studentCount].fio.surname, students[studentCount].fio.name,
        students[studentCount].fio.patronymic, students[studentCount].group, students[studentCount].age);
studentCount++;
}

fclose(file);
}

// Функция для сохранения данных в файл
void saveStudentsToFile(const char* filename)
{
    FILE* file = fopen(filename, "w");
    if (file == NULL)
    {
        printf("Ошибка открытия файла %s\n", filename);
        return;
    }

    setlocale(LC_ALL, "Russian"); // Устанавливаем русскую локаль

    for (int i = 0; i < studentCount; i++)
    {
        if (i == 0)
        {
            fprintf(file, "%s %s %s %s %s", students[i].fio.surname, students[i].fio.name,
                    students[i].fio.patronymic, students[i].group, students[i].age);
        }
        else {
            fprintf(file, "\n%s %s %s %s %s", students[i].fio.surname, students[i].fio.name,
                    students[i].fio.patronymic, students[i].group, students[i].age);
        }
    }

    fclose(file);
}

// Функция для вывода данных о всех студентах
void printAllStudents()
{
    setlocale(LC_ALL, "Russian"); // Устанавливаем русскую локаль
    if (studentCount == 0) {
        printf("База данных пуста.\n");
        return;
    }

    printf("База данных по студентам:\n");
    printf("\n%s", table);
    for (int i = 0; i < studentCount; i++)
    {
        printf(" | %12s | %13s | %13s | %12s | %10s |\n", students[i].fio.surname, students[i].fio.name,
                students[i].fio.patronymic, students[i].group, students[i].age);
        printf("%s", ender);
    }
    return;
}

// Функция для сортировки студентов по фамилии

```

```

void sortStudentsByName() {
    for (int i = 0; i < studentCount - 1; i++)
    {
        for (int j = i + 1; j < studentCount; j++)
        {
            if (strcmp(students[i].fio.surname, students[j].fio.surname) > 0)
            {
                Student temp = students[i];
                students[i] = students[j];
                students[j] = temp;
            }
        }
    }
}

// Функция для поиска студента по фамилии
int findStudentByName(const char* name)
{
    for (int i = 0; i < studentCount; i++)
    {
        if (strcmp(students[i].fio.surname, name) == 0)
        {
            return i;
        }
    }
    return -1;
}

// Функция для добавления нового студента
void addStudent()
{
    setlocale(LC_ALL, "Russian"); // Устанавливаем русскую локаль
    char over[] = "Превышен максимальный размер базы данных.\n";
    int g = 0;
    __asm
    {
        mov ecx, studentCount;
        cmp ecx, MAX_STUDENTS;
        jb ende;
        lea eax, over;
        push eax;
        call dword ptr printf;
        mov edx, 1;
        mov g, edx;
    ende:

    }
    if (g == 1)
    {
        return;
    }
    printf("Введите фамилию студента: ");
    fgets(students[studentCount].fio.surname, MAX_NAME_LENGTH, stdin);
    students[studentCount].fio.surname[strlen(students[studentCount].fio.surname, "\n")] = 0;
    getchar(); // Очистка буфера после ввода фамилии

    printf("Введите имя студента: ");
    fgets(students[studentCount].fio.name, MAX_NAME_LENGTH, stdin);
    students[studentCount].fio.name[strlen(students[studentCount].fio.name, "\n")] = 0;
    getchar(); // Очистка буфера после ввода имени

    printf("Введите отчество студента: ");

```

```

fgets(students[studentCount].fio.patronymic, MAX_NAME_LENGTH, stdin);
students[studentCount].fio.patronymic[strcspn(students[studentCount].fio.patronymic, "\n")] = 0;
getchar(); // Очистка буфера после ввода отчества

printf("Введите группу студента: ");
fgets(students[studentCount].group, MAX_GROUP_LENGTH, stdin);
students[studentCount].group[strcspn(students[studentCount].group, "\n")] = 0;
getchar(); // Очистка буфера после ввода группы

printf("Введите возраст студента: ");
fgets(students[studentCount].age, MAX_AGE_LENGTH, stdin);
students[studentCount].age[strcspn(students[studentCount].age, "\n")] = 0;
getchar(); // Очистка буфера после ввода возраста

studentCount++;
printf("Студент добавлен.\n");
return;
}

// Функция для удаления студента по имени
void deleteStudent(const char* name)
{
    int index = findStudentByName(name);
    if (index == -1)
    {
        printf("Студент с фамилией %s не найден.\n", name);
        return;
    }

    for (int i = index; i < studentCount - 1; i++)
    {
        students[i] = students[i + 1];
    }
    studentCount--;
    printf("Студент удален.\n");
}

// Функция для редактирования данных о студенте
void editStudent(const char* name)
{
    setlocale(LC_ALL, "Russian"); // Устанавливаем русскую локаль

    int index = findStudentByName(name);
    if (index == -1)
    {
        printf("Студент с фамилией %s не найден.\n", name);
        return;
    }

    printf("Введите новую фамилию студента (пустое для пропуска): ");
    char newSurname[MAX_NAME_LENGTH];
    fgets(newSurname, MAX_NAME_LENGTH, stdin);
    newSurname[strcspn(newSurname, "\n")] = 0;
    if (strlen(newSurname) > 0) {
        strcpy(students[index].fio.surname, newSurname);
    }
    getchar(); // Очистка буфера после ввода фамилии

    printf("Введите новое имя студента (пустое для пропуска): ");
    char newName[MAX_NAME_LENGTH];
    fgets(newName, MAX_NAME_LENGTH, stdin);
    newName[strcspn(newName, "\n")] = 0;

```

```

        if (strlen(newName) > 0)
        {
            strcpy(students[index].fio.name, newName);
        }
    }
    getchar(); // Очистка буфера после ввода имени

    printf("Введите новое отчество студента (пустое для пропуска): ");
    char newPatronymic[MAX_NAME_LENGTH];
    fgets(newPatronymic, MAX_NAME_LENGTH, stdin);
    newPatronymic[strcspn(newPatronymic, "\n")] = 0;
    if (strlen(newPatronymic) > 0)
    {
        strcpy(students[index].fio.patronymic, newPatronymic);
    }
    getchar(); // Очистка буфера после ввода отчества

    printf("Введите новую группу студента (пустое для пропуска): ");
    char newGroup[MAX_GROUP_LENGTH];
    fgets(newGroup, MAX_GROUP_LENGTH, stdin);
    newGroup[strcspn(newGroup, "\n")] = 0;
    if (strlen(newGroup) > 0)
    {
        strcpy(students[index].group, newGroup);
    }
    getchar(); // Очистка буфера после ввода группы

    printf("Введите новый возраст студента (пустое для пропуска): ");
    char newAge[MAX_AGE_LENGTH];
    fgets(newAge, MAX_AGE_LENGTH, stdin);
    newAge[strcspn(newAge, "\n")] = 0;
    if (strlen(newAge) > 0)
    {
        strcpy(students[index].age, newAge);
    }
    getchar(); // Очистка буфера ввода от оставшегося '\n'

    printf("Данные студента обновлены.\n");
}

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251); // Устанавливаем русскую локаль

    // Загрузка данных из файла (если файл существует)
    loadStudentsFromFile("students.txt");

    // Главное меню
    int choice;
menu:
    do
    {
        main:
            printf("\n--- Меню ---\n");
            printf("1. Вывести всех студентов\n");
            printf("2. Сортировать студентов по фамилии\n");
            printf("3. Найти студента по фамилии\n");
            printf("4. Добавить студента\n");
            printf("5. Удалить студента\n");
            printf("6. Редактировать данные о студенте\n");
            printf("7. Сохранить данные в файл\n");
            printf("0. Выход\n");

```

```

printf("Введите ваш выбор: ");
scanf("%d", &choice);
    getchar(); // Очистка буфера ввода

    switch (choice)
    {
    case 1:

        printAllStudents();

        goto menu;
        break;
    case 2:
        __asm {
            call sortStudentsByName;
        }
    printf("Студенты отсортированы по фамилии.\n");
    goto menu;
        break;
    case 3:
        {
            char name[MAX_NAME_LENGTH];
            printf("Введите фамилию студента: ");
            fgets(name, MAX_NAME_LENGTH, stdin);
            name[strcspn(name, "\n")] = 0;
            int index = findStudentByName(name);
            if (index == -1)
            {
                printf("Студент с фамилией %s не найден.\n", name);
            }
        }
        else
        {
            printf("Студент найден:\n");
            printf(" |%12s| %13s | %13s| %12s| %10s|\n", students[index].fio.surname, students[index].fio.name,
                students[index].fio.patronymic, students[index].group,
                students[index].age);
        }
        goto menu;
        break;
    }
    case 4:
        __asm {
            call addStudent
        }
        goto menu;
        break;
    case 5:
        {
            char name[MAX_NAME_LENGTH];
            printf("Введите фамилию студента для удаления: ");
            fgets(name, MAX_NAME_LENGTH, stdin);
            name[strcspn(name, "\n")] = 0;
            deleteStudent(name);
            goto menu;
            break;
        }
    case 6:
        {
            char name[MAX_NAME_LENGTH];
            printf("Введите фамилию студента для редактирования: ");
            fgets(name, MAX_NAME_LENGTH, stdin);
            name[strcspn(name, "\n")] = 0;

```

```

        editStudent(name);
        goto menu;
        break;
    }
    case 7:
        saveStudentsToFile("students.txt");
        printf("Данные сохранены в файл students.txt\n");
        goto menu;
        break;
    case 0:
        printf("Выход из программы.\n");
        break;
    default:
        printf("Неверный выбор.\n");
        break;
    }
} while (choice != 0);

return 0;
}

```

Приложение Б

Результат работы программы

```
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 3
Введите фамилию студента: ссс
Студент найден:
|      ссс|          сс |          сс|          сс|          сс|
--- Меню ---
1. Вывести всех студентов
2. Сортировать студентов по фамилии
3. Найти студента по фамилии
4. Добавить студента
5. Удалить студента
6. Редактировать данные о студенте
7. Сохранить данные в файл
0. Выход
Введите ваш выбор: 0
Выход из программы.

D:\flesh\Progr\лабы\GPT SILA\Project1\Debug\Project1.exe (процесс 20324) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно.
```

Рисунок 20 – Результат работы программы

```
| ссс сс сс сс сс
ааааа ааа аа аа аа
```

Рисунок 21 - Сохраненная в файл база данных