

Министерство науки ^и высшего образования РФ
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Программирование»
на тему: «Разработка программы сложной структуры
методом нисходящего программирования»

С.В.А.
28.12.22

Выполнила:
студентка группы 21BO2 Хрусталёва И. М.

Принял:
Митрохин М.А.

Пенза 2022

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Структура модулей программы, схема алгоритма программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

1. В соответствии с графиком выполнения курсового проекта.

2.

3.

Дата выдачи задания «1» _____ октября 2022 г.

Дата защиты проекта «_» _____ 2022 г.

Руководитель Митрохин Максим Александрович



Задание получил _____ «1» _____ октября 2022 г.

Студент Хрусталёва Ирина Михайловна / бул

Содержание

Введение.....	4
1 Постановка задачи.....	5
2 Выбор решения.....	7
2.1 Определение необходимых модулей программы	7
2.2 Определение структуры файла базы данных	8
3 Описание разработки программы.....	9
4 Отладка и тестирование	12
5 Описание программы.....	14
5.1 Общие сведения.....	14
5.2 Функционал назначения.....	14
5.3 Описание логической структуры.....	14
6 Руководство пользователя.....	16
Заключение	18
Список используемых источников	19
Приложение А Листинги программы.....	20
Приложение А.1 Файл «f.cpp».....	20
Приложение А.2 Файл «main.cpp».....	31
Приложение А.3 Файл «f.h»	33
Приложение В Снимки экрана	34

Введение

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

Базы данных — это совокупность структур, предназначенных для хранения больших объемов информации, и программных модулей, осуществляющих управление данными, выборку, сортировку и другие действия над ними. В базе данных можно сохранять любую информацию, чтобы позже получать к ней доступ.

Информация базы данных хранится в одной или нескольких таблицах. Любая таблица с данными состоит из набора однотипных записей, расположенных друг за другом, которые можно добавлять, изменять и удалять. Каждая запись является набором именованных полей или ячеек, которые могут хранить самую разнообразную информацию (фамилия, дата рождения и пр.).

С появлением и расширением компьютерных технологий появились противоречия между ограниченными возможностями человека по восприятию и переработке информации с существующими массивами хранящейся и передаваемой информации. Для решения подобных проблем применяются автоматизированные базы данных. Они стали неотъемлемой частью практически всех компьютерных систем – от глобальной отрасли до отдельного предприятия. Для персональных компьютеров было разработано множество различных программ, предназначенных для решения познавательной потребности человека.

Данная программа является информационно-поисковой системой, позволяющей любому пользователю просматривать и добавлять информацию, а также выполнять над ней другие операции.

1 Постановка задачи

Необходимо разработать приложение, наименование которого: База данных «Студенты». Программа содержит следующие поля: номер, фамилия, имя, отчество, дата рождения, факультет, группа и номер телефона.

Программа должна включать в себя функции добавления записей в базу данных (далее – БД), удаления записей из БД, редактирования записей БД, поиска и сортировки записей БД, фильтрации записей БД по определенному критерию, создания новой БД, сохранения текущей БД, загрузки сохраненных БД. Для реализации данных требований необходимо разработать интуитивно понятный для пользователя интерфейс и определить названия пунктов в меню. Нужно разработать алгоритм функционирования меню и обработки событий клавиатуры в консоли, продумать структуру работы программы и механизм взаимодействия входящих в нее функций.

Многомодульность программы. Требуется поделить программу на логические модули, что повысит расширяемость, упростит редактирование и исправление возникающих ошибок при отладке или тестировании программы.

Использование сложных типов данных – структур, списков, файлов. Требуется изучить базовые и сложные типы данных, что позволит организовать структурирование БД.

Режим работы системы – текстовый / графический. Требуется определить тип интерфейса, изучить способы его реализации и механизмы работы.

Устройство ввода информации – клавиатура, мышь. Требуется однозначное распознавание действия пользователя, реализованного через клавиатуру или мышь, и выполнение соответствующего действия программы.

Пользовательский интерфейс должен быть построен на основе меню и панели инструментов. Требуется изучить способы разработки меню, выбрать наиболее подходящий вариант для данного задания и условий. Меню и панель инструментов являются основной задачей разработки интуитивно понятного интерфейса.

Наличие заставки. Заставка необходима для сообщения пользователю достаточной визуальной информации о программе и возможности перехода к работе с программой.

Операционная система MS Windows. Язык программирования – С. Среда разработки – MS Visual Studio 2019.

2 Выбор решения

2.1 Определение необходимых модулей программы

Разработанная программа состоит из трех модулей:

1. f.h
2. f.cpp
3. main.cpp

Структура модулей программы приведена на рисунке 1.

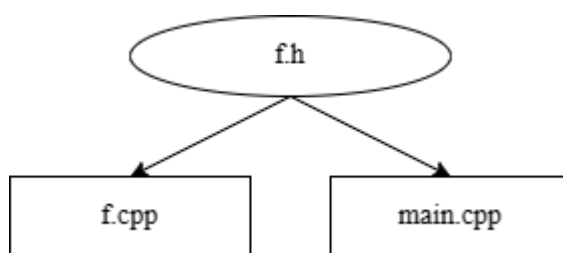


Рисунок 1 – Структура модулей программы

Заголовочный файл f.h необходим для объявления глобальных переменных, функций из модуля f.cpp и подключения стандартных библиотек языка C. Данная библиотека была создана специально для данной курсовой работы.

Модуль f.cpp содержит все требуемые функции программы: добавление записей в БД, удаление записей из БД, редактирование записей в БД, поиск записей в БД, фильтрация записей в БД по определенному критерию, создание новой БД, сохранение текущей БД, загрузка сохраненных БД, сортировки записей в БД.

Модуль main.c – основной модуль данной курсовой работы. Он содержит в себе заставку и вызов функций из модуля f.cpp.

Все модули с расширением .cpp используют заголовочные файлы из библиотеки f.h. Тем самым выполняется требование модульности данной курсовой работы. Управление приложением осуществляется с помощью клавиатуры. Одновременная работа двух программ с одной базой данных не допускается. Данные из БД отображаются в программе в виде таблицы из нескольких столбцов.

2.2 Определение структуры файла базы данных

Для структурирования данных внутри файла была выбрана следующая структура данных:

```
struct student
{
    int number; //порядковый номер
    char name[12]; //имя
    char surname[20]; //фамилия
    char patronymic[20]; //отчество
    char birthdate[11]; //дата рождения
    char faculty[10]; //факультет
    char group[7]; //группа
    char phone[13]; //номер телефона
};
```

Поля структуры «Порядковый номер», «Имя», «Фамилия», «Отчество», «Дата рождения», «Факультет», «Группа», «Номер телефона» имеют текстовый формат. Под них выделено различное количество символов (от 7 до 21). Поле «Порядковый номер» определено переменной типа int.

База данных реализована в виде массива структур.

Количество записей в файле ограничено пользователем.

3 Описание разработки программы

В написании данной программы использован достаточно распространенный язык программирования С. Он даёт программисту возможность выражать свои алгоритмы и данные наиболее подходящими для этого средствами.

В качестве среды программирования был выбран программный продукт Visual Studio 2019.

Интерфейс информационной системы в силу своего удобства и наименьшего времени доступа к данным представлен в виде меню, перемещение по пунктам которого осуществлено с помощью клавиш управления.

Для выхода из программы добавлен пункт меню «Выход».

Исходя из того, что данные о студентах (например, группа) со временем могут добавляться и изменяться, в БД предусмотрена функция редактирования записей и соответствующий пункт меню «Добавление или изменение студента».

Для быстрого поиска полной информации об имеющихся записях (студентах) меню содержит пункт «Поиск». Причем поиск осуществляется по всем категориям данных, что позволяет пользователю без труда и потери времени получить доступ к нужной записи.

Для упорядочения БД предусмотрен пункт меню «Сортировка», в котором можно выстроить записи БД всем критериям.

Перечисленные выше операции можно выполнять как в уже созданной БД, так и в создаваемой в настоящем времени БД. Для открытия существующей базы предусмотрен пункт меню «Загрузить», новая создается в начале программы. Для сохранения созданных записей предусмотрен одноименный пункт меню.

Основная программа выполняется функцией `main()`. Функция отвечает за инициализацию консоли, вывод заставки, передачу управления в функции обработки событий меню. После ввода некоторого значения с клавиатуры функция обработки событий меню получает идентификатор выбранного пункта меню, в соответствии с которым организуется передача управления в нужную функцию. Блок-схема функции `main()` и функции обработки событий меню представлена на рисунке 2.



Рисунок 2 – Схема программы

Когда главная функция обнаруживает событие от клавиатуры, вызывается функция обработки события. В зависимости от активного элемента выполняются те или иные действия. Алгоритм работы меню приведен на рисунке 3.

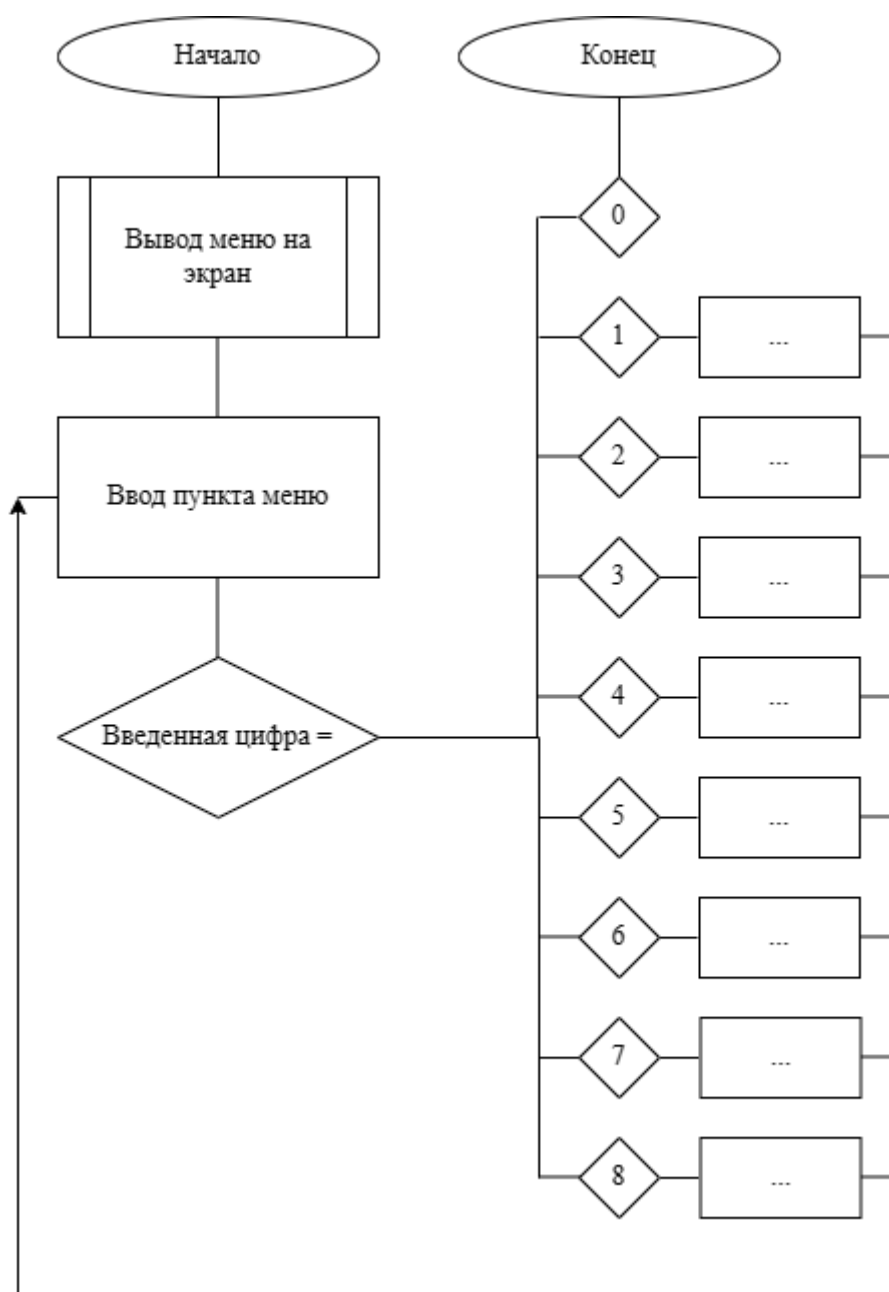


Рисунок 3 – Алгоритм работы меню

4 Отладка и тестирование

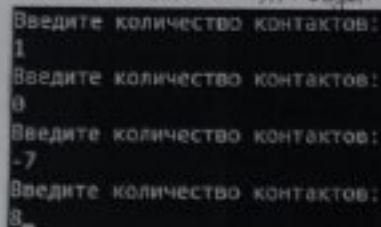
Средой разработки была выбрана программа Visual Studio 2019. Программа содержит в себе все необходимые средства для успешной отладки и тестирования модулей программы, а также всей программы в целом. Данная среда разработки выпущена относительно недавно, что позволяет проводить необходимые операции над разработанной программой наиболее эффективно и продуктивно.

Для выполнения отладки применялись средства Visual Studio, такие как: точки останова, анализ содержимого переменных, выполнение кода по шагам (по строкам), анализ памяти.

Тестирование проводилось в рабочем порядке, на всех этапах разработки программы, в т.ч. после завершения написания кода. Тестирование позволило выявить множество проблем и недочетов по части внешнего вида. Для эффективной работы программы были созданы проверки, не позволяющие проводить обработку некоторых частей программы в определенных условиях.

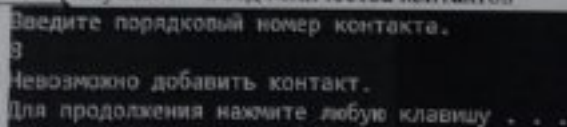
Результаты отладки приведены ниже.

Ввод количества контактов приведен на рисунке 4.



```
Введите количество контактов:  
1  
Введите количество контактов:  
0  
Введите количество контактов:  
-7  
Введите количество контактов:  
8
```

Рисунок 4 – Ввод количества контактов



```
Введите порядковый номер контакта.  
3  
Невозможно добавить контакт.  
Для продолжения нажмите любую клавишу . . .
```

Рисунок 5 – Ложное добавление контакта

*Ложное добавление контактов
приведено на рисунке 5.*

Имена укажите конкретные, приведенные на рисунке 6.

Введите порядковый номер контакта.
0
Невозможно удалить контакт.
Для продолжения нажмите любую клавишу . . .

Рисунок 6 – Ложное удаление контакта

Сортировка по какому параметру?
0: И
1: Имя.
2: Фамилия.
3: Отчество.
4: Дата рождения.
5: Факультет.
6: Группа.
7: Телефон.
Ваш выбор:
10

Вывод информации
пока не приво-
ден на рисунке 7

Выбор при этом
не должен приводить
к рисунку 8

Рисунок 7 – Выбор несуществующего пункта

Не выбрано.
Для продолжения нажмите любую клавишу . . .

Рисунок 8 – Ошибка при таком выборе

Введите имя файла:
l.dat
Ошибка открытия файла
Для продолжения нажмите любую клавишу . . .

Ошибка открытия файла
приведена на рисунке 8

Рисунок 9 – Ошибка открытия файла

Введите имя файла:
k.dat
Ошибка удаления файла!
Для продолжения нажмите любую клавишу . . .

Ошибка удаления
файла приведена
на рисунке 9

Рисунок 10 – Ошибка удаления файла

5 Описание программы

5.1 Общие сведения

Программа «База данных «Студенты»» предназначена для работы с базой студентов. Программа реализована на Visual Studio 2019.

Программа написана на языке C, с использованием компилятора Microsoft Visual C++. Её текст содержится в различных файлах и приведен в приложении А.

5.2 Функционал назначения

Программа «База данных «Студенты»» предназначена для действий с базой данных, таких как добавление записей в БД, удаление записей из БД, редактирование записей в БД, поиск записей в БД, фильтрация записей в БД по определенному критерию, создание новой БД, сохранение текущей БД, загрузка сохраненных БД, сортировки записей в БД.

5.3 Описание логической структуры

Модуль `main.cpp` является основным модулем программы. При запуске программы на экран он выводит заставку, собирает данные (количество записей и пункт меню) и вызывает функции из вспомогательного модуля `f.cpp`.

Вспомогательный модуль `f.cpp` содержит функции реализующие почти весь процесс работы с БД. Модуль `f.cpp` содержит все требуемые функции программы: добавление записей в БД, удаление записей из БД, редактирование записей в БД, поиск записей в БД, фильтрация записей в БД по определенному критерию, создание новой БД, сохранение текущей БД, загрузка сохраненных БД, сортировки записей в БД.

Меню наполовину реализовано в файле `f.cpp` функцией типа `void menu()`. Состоит из вывода пунктов, которые может выбрать пользователь: «Выход», «Вывод», «Добавить или изменить», «Удалить», «Сортировка», «Поиск», «Загрузить», «Сохранить».

Добавление студента реализуется функцией `void add_student(student* a, int n)` в которую передаётся указатель на базу и количество записей в ней. В ней действуют основные функции для записи/изменения студента `gets_s` и `scanf`.

Удаление студента/студентов реализуется функцией `void delete_student(student* a, int n)` и `void delete_s(student* a, int n)`, в которые передаётся указатель на базу и количество записей в ней. В ней действуют основная функция для удаления студента `strncpy` с помощью очистки строк.

Студенты выводятся с помощью функции `void print_student(student* a, int n)`, в которую передаётся указатель на базу и количество записей в ней. В ней действуют основная функция для вывода студентов `print` с помощью цикла.

Работа с файлами происходит с помощью функций `void save_student(struct student* st, int n)`, `void download_student(struct student* p, int n)`, а также `void delete_file()`. В две из них передаётся указатель на базу и количество записей в ней. В них действуют функции, такие как: `fopen`, `fprintf`, `exit`, `getc`, `fclose` и `remove`.

Сортировка и поиск происходят с помощью функций `void search(student* a, int n)`, `void sort(student* a, int n)`, в которые передаётся указатель на базу и количество записей в ней. В них активно работают операторы условия и цикла.

Выход из меню и программы реализуется в файле `main.cpp`.

Главной задачей функций файла `f.cpp` является работа с базой данных.

Главной задачей файла `main.cpp` является обеспечение возможности выбора пользователю одного из пунктов меню и возврата программе идентификатора выбранного пункта, в соответствии с которым происходит передача управления в нужную часть программы. Основное меню реагирует на нажатие клавиш: цифры от 0 до 8. Дополнительная задача файла `main.cpp` – вывод заставки.

8. Функциональные возможности

Целью данной программы является управление информацией о студентах.

Рисунок 8.1. Функциональные возможности

Рисунок 8.2. Меню функциональных возможностей

Для выхода из программы нажмите 9: Exit.

Для добавления или изменения студента нажмите 1: Add.

Для удаления студента нажмите 2: Del.

Для вывода списка студентов нажмите 3: List.

Для сохранения БД в файл нажмите 4: Save.

Для загрузки БД из файла нажмите 5: Load.

Для удаления файла с БД нажмите 6: DelFile.

Для поиска студента нажмите 7: Find.

Для сортировки студентов нажмите 8: Sort.

```

1. Выход
2. Добавление или изменение студента.
3. Удаление студента.
4. Вывод студентов.
5. Сохранение в файл.
6. Загрузка из файла.
7. Удаление файла.
8. Поиск студента.
9. Сортировка студентов.
10. Ваш выбор:

```

Рисунок 8.3. Основное меню

Для пунктов 1 и 2 вам необходимо знать порядковый номер студента.

Узнать его вы можете с помощью пункта 3 или 7.

В пунктах 7 и 8 вы можете выбрать критерий поиска/сортировки.

Дата добавления: 13.05.2014 13:00



Рисунок 2. Меню поиска.

Для поиска/сортировки по идентификационному номеру нажмите 0 + Enter.

Для поиска/сортировки по имени нажмите 1 + Enter.

Для поиска/сортировки по фамилии нажмите 2 + Enter.

Для поиска/сортировки по отчеству нажмите 3 + Enter.

Для поиска/сортировки по дате рождения нажмите 4 + Enter.

Для поиска/сортировки по факультету нажмите 5 + Enter.

Для поиска/сортировки по группе нажмите 6 + Enter.

Для поиска/сортировки по номеру телефона нажмите 7 + Enter.

Для работы с файлами вам нужно лишь имя файла, например «University.dat». Будьте внимательнее, используется лишь расширение .dat.

Заключение

В ходе выполнения курсовой работы были получены и расширены навыки разработки многомодульных программ на языке C, освоены функции работы с консолью, способы интерпретации событий от клавиатуры; изучены различные принципы реализации меню. Освоены на практике в большом объеме функции для работы с файлами. Закреплены приемы работы в компиляторе Visual Studio 2019, отточены навыки программирования на языке C.

В данной курсовой работе была написана программа: база данных «Студенты», используемая для хранения информации о студентах. Программа представляет ограниченный функционал, содержит базовые средства для выполнения операций в базе данных, которые в будущем можно усовершенствовать. Соответственно, программа не требует большого количества системных ресурсов, обеспечивает быструю обработку информации.

Список используемых источников

1. Курс программирования на языке С. С. Брайан; В. Подрезинский.
2. <http://all-ht.ru>
3. <https://ru.stackoverflow.com>
4. <https://metanit.com>
5. Программирование на C++ примеры и задачи. А. Вашиль.
1. Подольский В.В., Дуванов Е.Е. Курс программирования на языке С. Учебник / Подольский В.В., Дуванов Е.Е. - Пенза, 2012. - 384 с.
5. Вашиль А.Н. Программирование на C++ с примерами и задачами / Вашиль А.Н. - Пенза: Издательство, 2017. - 368 с. - ISBN 978-5-699-82445-3

Приложение А Листинги программы

Приложение А.1 Файл «f.c»

```
#include "f.h"

void add_student(student* a, int n)
{
    int f = 0;
    char* d;
    char p[20] = "";
    d = gets_s(p);
    printf("Введите порядковый номер контакта.\n");
    scanf("%d", &f);
    if ((1 <= f) && (f <= n))
    {
        d = gets_s(p);
        f = f - 1;
        a[f].number = f + 1;
        printf("Введите имя:\n");
        d = gets_s(a[f].name);
        printf("Введите фамилию:\n");
        d = gets_s(a[f].surname);
        printf("Введите отчество:\n");
        d = gets_s(a[f].patronymic);
        printf("Введите дату рождения:\n");
        d = gets_s(a[f].birthdate);
        printf("Введите факультет:\n");
        d = gets_s(a[f].faculty);
        printf("Введите группу:\n");
        d = gets_s(a[f].group);
        printf("Введите телефон:\n");
        d = gets_s(a[f].phone);
    }
    else
        printf("Невозможно добавить контакт.\n");
    d = gets_s(p);
    system("pause");
    system("cls");
}

void delete_student(student* a, int n)
{
    char* d;
    char p[20] = "";
    int f;
    printf("Введите порядковый номер контакта.\n");
    scanf_s("%d", &f);
    if ((1 <= f) && (f <= n))
    {
```

```

        f = f - 1;
        strncpy(a[f].name, "", 12);
        strncpy(a[f].surname, "", 20);
        strncpy(a[f].patronymic, "", 20);
        strncpy(a[f].birthdate, "", 10);
        strncpy(a[f].faculty, "", 10);
        strncpy(a[f].group, "", 7);
        strncpy(a[f].phone, "", 13);
    }
    else
        printf("Невозможно удалить контакт.\n");
    d = gets_s(p);
    system("pause");
    system("cls");
}
void delete_s(student* a, int n)
{
    for (int f = 0; f < n; f++)
    {
        a[f].number = f + 1;
        strncpy(a[f].name, "", 12);
        strncpy(a[f].surname, "", 20);
        strncpy(a[f].patronymic, "", 20);
        strncpy(a[f].birthdate, "", 10);
        strncpy(a[f].faculty, "", 10);
        strncpy(a[f].group, "", 7);
        strncpy(a[f].phone, "", 13);
    }
}

void print_student(student* a, int n)
{
    printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя", "Фамилия",
"Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
    for (int i = 0; i < n; i++)
    {
        printf("%d %12s %20s %20s %14s %10s %7s %13s\n", a[i].number, a[i].name,
a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group, a[i].phone);
    }
    system("pause");
    system("cls");
}

void save_student(struct student* st, int n)
{
    char filename[20] = "l.dat";
    printf("Введите имя файла:\n");
    scanf("%s", filename);
    FILE* fp;
    char* c;

```

```

int size = n * sizeof(struct student);

if ((fp = fopen(filename, "wb")) == NULL)
{
    fprintf(stderr, "Ошибка открытия файла\n");
    system("pause");
    exit(0);
}

c = (char*)st;
for (int i = 0; i < size; i++)
{
    putc(*c, fp);
    c++;
}
fclose(fp);
system("pause");
system("cls");
}

```

```

void download_student(struct student* p, int n)
{
    char filename[20] = "l.dat";
    printf("Введите имя файла:\n");
    scanf("%s", filename);
    FILE* fp;
    char* c;
    int i;

    if ((fp = fopen(filename, "r")) == NULL)
    {
        fprintf(stderr, "Ошибка открытия файла\n");
        system("pause");
        exit(0);
    }
    c = (char*)p;
    while ((i = getc(fp)) != EOF)
    {
        *c = i;
        c++;
    }
    fclose(fp);
    system("pause");
    system("cls");
}

void delete_file()
{
    char filename[20] = "l.dat";
    printf("Введите имя файла:\n");
    scanf("%s", filename);
    if (remove(filename) != 0)

```

```

        printf("Ошибка удаления файла!\n");
    else
        printf("Файл успешно удалён.\n");
    system("pause");
    system("cls");
}
void search(student* a, int n)
{
    printf("Поиск по какому параметру?\n");
    printf("0: №\n");
    printf("1: Имя.\n");
    printf("2: Фамилия.\n");
    printf("3: Отчество.\n");
    printf("4: Дата рождения.\n");
    printf("5: Факультет.\n");
    printf("6: Группа.\n");
    printf("7: Телефон.\n");
    printf("Ваш выбор:\n");
    int par, g, m = 0;
    char s[20];
    scanf("%d", &par);
    system("cls");
    switch (par)
    {
    case 0:
        printf("Введите порядковый номер контакта.\n");
        scanf("%d", &g);
        for (int i = 0; i < n; i++)
        {
            if (a[i].number == g)
                m = 1;
        }
        g = g - 1;
        if (m == 0)
            printf("Совпадений не найдено.\n");
        else
        {
            printf("Найденый студенты.\n");
            printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
            printf("%d %12s %20s %20s %14s %10s %7s %13s\n", a[g].number,
a[g].name, a[g].surname, a[g].patronymic, a[g].birthdate, a[g].faculty, a[g].group, a[g].phone);
        }
        break;
    case 1:
        printf("Введите имя.\n");
        scanf("%s", &s);

        for (int i = 0; i < n; i++)
        {
            char* istr;
            istr = strstr(a[i].name, s);

```

```

        if (istr != NULL)
            m = 1;
    }

    if (m == 0)
        printf("Совпадений не найдено.\n");
    else
    {
        printf("Найденные студенты.\n");
        printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
        for (int i = 0; i < n; i++)
        {
            char* istr;
            istr = strstr(a[i].name, s);
            if (istr != NULL)
                printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
        }
    }
    break;
case 2:
    printf("Введите фамилию.\n");
    scanf("%s", &s);

    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].surname, s);
        if (istr != NULL)
            m = 1;
    }

    if (m == 0)
        printf("Совпадений не найдено.\n");
    else
    {
        printf("Найденные студенты.\n");
        printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
        for (int i = 0; i < n; i++)
        {
            char* istr;
            istr = strstr(a[i].surname, s);
            if (istr != NULL)
                printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
        }
    }
    break;

```

case 3:

```
printf("Введите отчество.\n");
scanf("%s", &s);

for (int i = 0; i < n; i++)
{
    char* istr;
    istr = strstr(a[i].patronymic, s);
    if (istr != NULL)
        m = 1;
}

if (m == 0)
    printf("Совпадений не найдено.\n");
else
{
    printf("Найденные студенты.\n");
    printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].patronymic, s);
        if (istr != NULL)
            printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
    }
}
break;
```

case 4:

```
printf("Введите дату рождения.\n");
scanf("%s", &s);

for (int i = 0; i < n; i++)
{
    char* istr;
    istr = strstr(a[i].birthdate, s);
    if (istr != NULL)
        m = 1;
}

if (m == 0)
    printf("Совпадений не найдено.\n");
else
{
    printf("Найденные студенты.\n");
    printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
    for (int i = 0; i < n; i++)
    {
        char* istr;
```

```

        istr = strstr(a[i].birthdate, s);
        if (istr != NULL)
            printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
    }
}
break;
case 5:
    printf("Введите факультет.\n");
    scanf("%s", &s);

    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].faculty, s);
        if (istr != NULL)
            m = 1;
    }

    if (m == 0)
        printf("Совпадений не найдено.\n");
    else
    {
        printf("Найденные студенты.\n");
        printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
        for (int i = 0; i < n; i++)
        {
            char* istr;
            istr = strstr(a[i].faculty, s);
            if (istr != NULL)
                printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
        }
    }
break;
case 6:
    printf("Введите группу.\n");
    scanf("%s", &s);

    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].group, s);
        if (istr != NULL)
            m = 1;
    }

    if (m == 0)
        printf("Совпадений не найдено.\n");

```

```

else
{
    printf("Найденные студенты.\n");
    printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].group, s);
        if (istr != NULL)
            printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
    }
}
break;
case 7:
    printf("Введите телефон.\n");
    scanf("%s", &s);

    for (int i = 0; i < n; i++)
    {
        char* istr;
        istr = strstr(a[i].phone, s);
        if (istr != NULL)
            m = 1;
    }

    if (m == 0)
        printf("Совпадений не найдено.\n");
    else
    {
        printf("Найденные студенты.\n");
        printf("%1s %12s %20s %20s %14s %10s %7s %13s\n", "№", "Имя",
"Фамилия", "Отчество", "Дата рождения", "Факультет", "Группа", "Телефон");
        for (int i = 0; i < n; i++)
        {
            char* istr;
            istr = strstr(a[i].phone, s);
            if (istr != NULL)
                printf("%d %12s %20s %20s %14s %10s %7s %13s\n",
a[i].number, a[i].name, a[i].surname, a[i].patronymic, a[i].birthdate, a[i].faculty, a[i].group,
a[i].phone);
        }
    }
}
break;
default:
    printf("Не выбрано.\n");
    break;
}
system("pause");
system("cls");

```

```

}
void sort(student* a, int n)
{
    printf("Сортировка по какому параметру?\n");
    printf("0: №\n");
    printf("1: Имя.\n");
    printf("2: Фамилия.\n");
    printf("3: Отчество.\n");
    printf("4: Дата рождения.\n");
    printf("5: Факультет.\n");
    printf("6: Группа.\n");
    printf("7: Телефон.\n");
    printf("Ваш выбор:\n");
    int par;
    student d;
    scanf("%d", &par);
    system("cls");
    switch (par)
    {
    case 0:
        for (int i = 0; i < n - 1; i++)
            for (int j = i + 1; j < n; j++)
            {
                if (a[i].number > a[j].number)
                {
                    d = a[i];
                    a[i] = a[j];
                    a[j] = d;
                }
            }
        break;
    case 1:
        for (int i = 0; i < n - 1; i++)
            for (int j = i + 1; j < n; j++)
            {
                if (strcmp(a[i].name, a[j].name)>0)
                {
                    d = a[i];
                    a[i] = a[j];
                    a[j] = d;
                }
            }
        break;
    case 2:
        for (int i = 0; i < n - 1; i++)
            for (int j = i + 1; j < n; j++)
            {
                if (strcmp(a[i].surname, a[j].surname)>0)
                {
                    d = a[i];
                    a[i] = a[j];
                    a[j] = d;
                }
            }
        break;
    }
}

```

```

        }
    }
    break;
case 3:
    for (int i = 0; i < n - 1; i++)
        for (int j = i + 1; j < n; j++)
        {
            if (strcmp(a[i].patronymic, a[j].patronymic)>0)
            {
                d = a[i];
                a[i] = a[j];
                a[j] = d;
            }
        }
    break;
case 4:
    for (int i = 0; i < n - 1; i++)
        for (int j = i + 1; j < n; j++)
        {
            if (strcmp(a[i].birthdate, a[j].birthdate)>0)
            {
                d = a[i];
                a[i] = a[j];
                a[j] = d;
            }
        }
    break;
case 5:
    for (int i = 0; i < n - 1; i++)
        for (int j = i + 1; j < n; j++)
        {
            if (strcmp(a[i].faculty, a[j].faculty)>0)
            {
                d = a[i];
                a[i] = a[j];
                a[j] = d;
            }
        }
    break;
case 6:
    for (int i = 0; i < n - 1; i++)
        for (int j = i + 1; j < n; j++)
        {
            if (strcmp(a[i].group, a[j].group)>0)
            {
                d = a[i];
                a[i] = a[j];
                a[j] = d;
            }
        }
    break;
case 7:

```

```

        for (int i = 0; i < n - 1; i++)
            for (int j = i + 1; j < n; j++)
            {
                if (strcmp(a[i].phone, a[j].phone)>0)
                {
                    d = a[i];
                    a[i] = a[j];
                    a[j] = d;
                }
            }
        break;
default:
    printf("Не выбрано.\n");
    break;
}

system("pause");
system("cls");
}

```

```

int Menu()
{
    int OptMenu;
    printf("Выберите действие:\n");
    printf("0: Выход.\n");
    printf("1: Добавление или изменение студента.\n");
    printf("2: Удаление студента.\n");
    printf("3: Вывод студентов.\n");
    printf("4: Сохранение в файл.\n");
    printf("5: Загрузка из файла.\n");
    printf("6: Удаление файла.\n");
    printf("7: Поиск студента.\n");
    printf("8: Сортировка студентов.\n");
    printf("Ваш выбор:\n");
    scanf_s("%d", &OptMenu);
    system("cls");
    return OptMenu;
}

```

Приложение А.2 Файл «main.c»

```
#include "f.h"

int main(int argc, char* argv[])
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    printf("\n");
    printf("\n");
    printf("\n");
    printf("\n");
    printf("\n");
    printf("***** ** \n");
    printf("** ** ** ** ** \n");
    printf("** ** ** ** \n");
    printf("** ** ** ** \n");
    printf("***** ** \n");
    printf("\n");
    printf("\n");
    printf("\n");
    printf("\n");
    printf("\n");
    system("pause");
    system("cls");
    int n = 0;
    do
    {
        printf("Введите количество контактов:\n");
        scanf_s("%d", &n);
    } while (n < 2);
    system("cls");

    student* a = (student*)malloc(sizeof(struct student) * n);
    delete_s(a, n);

    int act;
    do {
        act = Menu();
        switch (act)
        {
            case 0:
                break;
            case 1:
                add_student(a,n);
                break;
            case 2:
                delete_student(a,n);
                break;
            case 3:
                print_student(a,n);
                break;
        }
    }
```

```
    case 4:
        save_student(a,n);
        break;
    case 5:
        download_student(a,n);
        break;
    case 6:
        delete_file();
        break;
    case 7:
        search(a,n);
        break;
    case 8:
        sort(a,n);
        break;
    default:
        break;
}
} while (act != 0);

free(a);
}
```

Приложение А.3 Файл «f.h»

```
#pragma once
#define _CRT_SECURE_NO_WARNINGS
#include <Windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <string.h>

struct student
{
    int number;
    char name[12];
    char surname[20];
    char patronymic[20];
    char birthdate[11];
    char faculty[10];
    char group[7];
    char phone[13];
};

void add_student(student* a, int n);
void delete_student(student* a, int n);
void delete_s(student* a, int n);
void print_student(student* a, int n);
void save_student(struct student* st, int n);
void download_student(struct student* p, int n);
void delete_file();
void search(student* a, int n);
void sort(student* a, int n);
int Menu();
```

Приложение В Снимки экрана

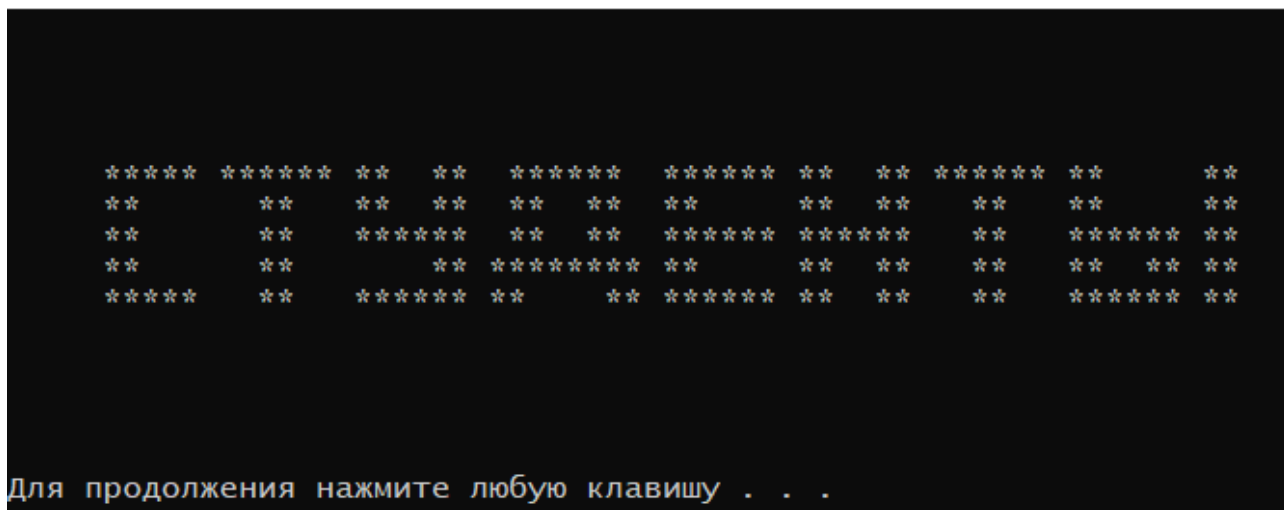


Рисунок В.1 – Снимок заставки

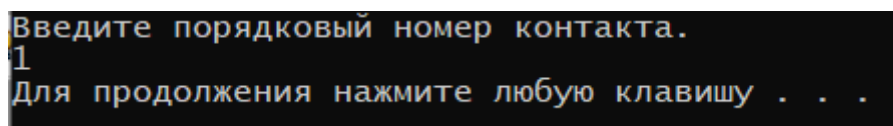


Рисунок В.2 – Удаление студента

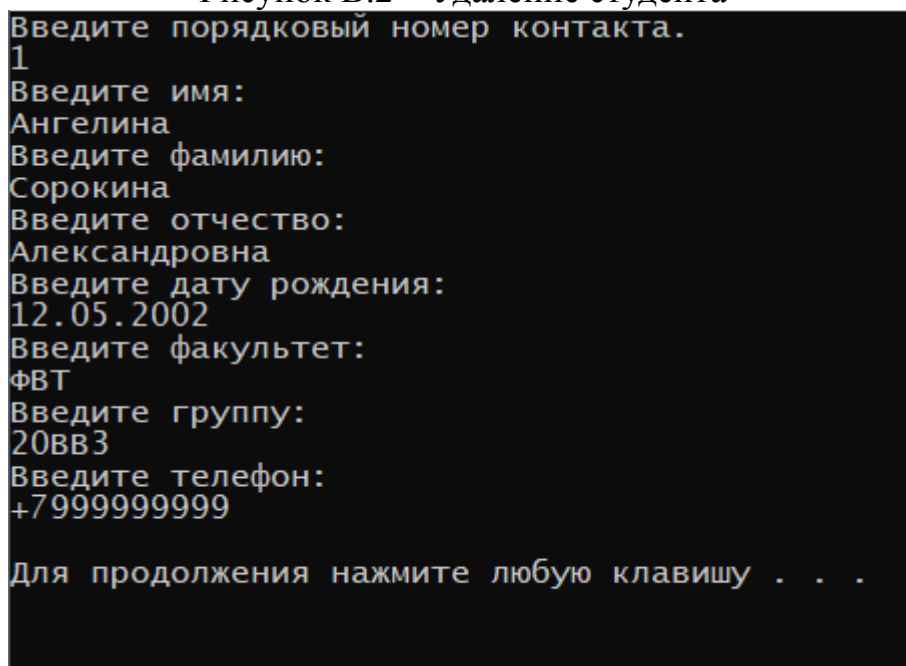


Рисунок В.3 – Добавление студента

№	Имя	Фамилия	Отчество	Дата рождения	Факультет	Группа	Телефон
1							
2							
3							

Для продолжения нажмите любую клавишу . . .

Рисунок В.4 – Вывод пустого списка студентов


```
Введите имя файла:  
f.dat  
Для продолжения нажмите любую клавишу . . .
```

Рисунок В.5 – Работа с файлами

```
Введите фамилию.  
Клоп  
Совпадений не найдено.  
Для продолжения нажмите любую клавишу . . .
```

Рисунок В.6 – Поиск студента

№	Имя	Фамилия	Отчество	Дата рождения	Факультет	Группа	Телефон
1	Игорь	Круг	Игоревич	10.09.2003	ИФКИС	21ик1	+79876543210
2	Роман	Девяткин	Николаевич\	31.02.2002	ФВТ	20во1	+79188709843
3	Алиса	Романова	Андреевна	23.03.2003	ФВТ	21вв3	+79176740466

Для продолжения нажмите любую клавишу . . .

Рисунок В.7 – Вывод студентов из файла

№	Имя	Фамилия	Отчество	Дата рождения	Факультет	Группа	Телефон
3	Алиса	Романова	Андреевна	23.03.2003	ФВТ	21вв3	+79176740466
2	Роман	Девяткин	Николаевич\	31.02.2002	ФВТ	20во1	+79188709843
1	Игорь	Круг	Игоревич	10.09.2003	ИФКИС	21ик1	+79876543210

Для продолжения нажмите любую клавишу . . .

Рисунок В.8 – Отсортированные студенты