

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

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Программирование»

на тему «Разработка программы сложной структуры методом многомодульного
программирования»

наименование программы «База данных по студентам»

Выполнил студент группы 21ВВ2:

Кривенкова В.С.

Принял:

Голотенков Н.О.

Ст.л. 

Пенза, 2022

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 Кривенковой В.С. Группа 21ВВ2

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

Исходные данные (технические требования) на проектирование

*Приложение: База данных по студентам.
Функции приложения: создание новой базы данных,
добавление записей, удаление записей, редактиро-
вание записей, поиск и сортировка записей, фильтрация
записей по опред. критерию, Сохранение текущей
базы данных в файл, Загрузка сохраненных баз
данных из файла.*

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 "22" февраля

Дата защиты проекта "07" 06 2022

Руководитель Гоготенков Н.О.

Задание получил "22" февраля 2022г.

Студент Кривенкова Валерия Сергеевна

Содержание

Введение	4
1. Постановка задачи.	5
2) Выбор решения. 2.1. Определение необходимых модулей программы.....	7
2.2 Определение структуры базы данных.....	8
3. Описание разработки программы.	8
4. Отладка и тестирование.	10
5. Описание программы.	10
6. Руководство пользователя.....	12
Заключение	17
Список используемых источников.....	18
Приложение А. Листинг программы.....	18

Введение

Базы данных — это совокупность структур, предназначенных для хранения больших объемов информации и программных модулей, осуществляющих управление данными, их выборку, сортировку и другие подобные действия. Информация базы данных хранится в одной или нескольких таблицах. Любая таблица с данными состоит из набора однотипных записей, расположенных

друг за другом. Они представляют собой строки таблицы, которые можно добавлять, удалять или изменять.

Каждая запись является набором именованных полей, или ячеек, которые могут хранить самую разнообразную информацию.

Создав одну таблицу, вы уже получаете полноценную базу данных. Однако в реальной жизни структуры баз данных, а соответственно и способы их создания, намного сложнее. В последние годы на первый план выдвигается новая отрасль - информационная индустрия, связанная с производством технических средств, методов, технологий для производства новых знаний. Эта индустрия тесно связана с развитием компьютерных технологий.

Структура – это составной тип данных, который получается путем объединения компонентов, принадлежащих к другим типам данных.

Структура предназначена для объединения нескольких переменных в один тип данных.

1. Постановка задачи.

Необходимо разработать программу «База данных. Список студентов». В данной программе информация структурирована по следующим полям: «ФИО», «Группа», «Факультет», «Курс», «Возраст». Для реализации данной программы необходимо определить структуру базы данных, размер, размер каждой записи базы данных.

Программа должна позволять пользователю осуществлять обработку данных, для этого в программе должны присутствовать следующие функции:

- Создание новой базы данных
- Добавление записей
- Удаление записей
- Поиск и сортировка записей
- Редактирование записей
- Фильтрация записей по определенному критерию
- Сохранение текущей базы данных в файл.

- Чтение базы данных из файла

Для реализации вышеперечисленных требований необходимо разработать интуитивно понятный интерфейс пользователя, определить названия пунктов меню, определить тип интерфейса, продумать навигацию по базе данных, изучить функции для работы с файлами. Это необходимо для четкого понимания функциональности программы.

В программе должны использоваться сложные типы данных: структуры, массивы, списки, файлы. Это необходимо для структурирования базы данных, для обработки данных в коде программы, для просмотра базы данных, с использованием файла, не используя запуск программы.

Устройствами ввода являются клавиатура и мышь. Навигация по пунктам меню и данные вводятся посредством клавиатуры. Программа должна быть разработана для работы в операционной системе MS Windows. Средой разработки является Microsoft Visual Studio 2010, язык программирования Си и Ассемблер.

2) Выбор решения.

2.1. Определение необходимых модулей программы.

Разработанная программа состоит из следующих модулей

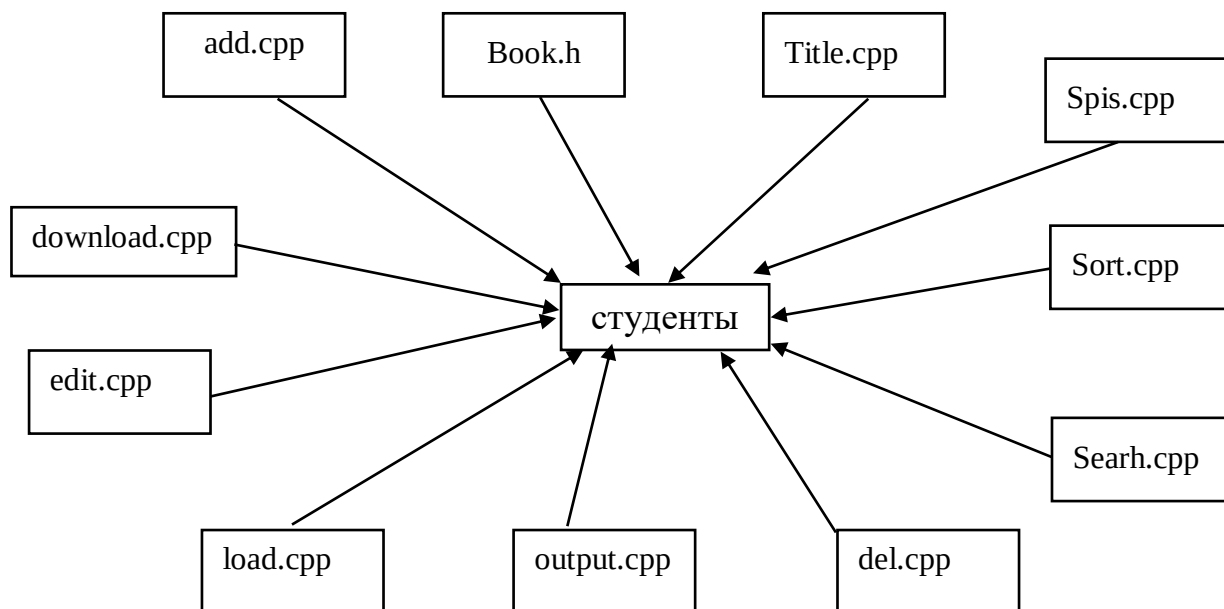


Рисунок 1 – Структура модулей программы

Файлы: «delete.cpp», «edit.cpp», «download.cpp», «load.cpp», «output.cpp», «search.cpp», «sort.cpp», «spis.cpp», «add.cpp» необходимы для включения функций обработки данных. Данные файлы содержат необходимые функции чтения, поиска, редактирование, добавления новых записей, создания, сортировки, вывода на экран базы данных и функцию сохранения результата с загрузкой из базы обработки базы данных.

Все вышеперечисленное удовлетворяет требованию модульности программы данной курсовой работы. Управление приложением осуществляется с помощью клавиатуры. При запуске программы выводится заставка-приветствие, после чего программа будет готова к дальнейшему исполнению. Программа предназначена для работы с базой данных, база данных находится в файле, файл так же можно создать

самому. Открытие и сохранение выходного файла выполняется средствами данной программы. Программа дает возможность просматривать содержимое базы данных, искать данные, сортировать их. Данные из базы отображаются в программе в виде таблицы из пяти столбцов «FIO», «Group», «faculty», «course», «age».

2.2 Определение структуры базы данных

Type def struct group_list

```
{  
    char fio[16],  
    group[16],  
    faculty[16],  
    course[16],  
    age[16];  
    group_list* next;  
};
```

Поля FIO («Фио»), group («Группа»), faculty («Факультет»), course («Курс»), age («Возраст») представляют собой символьную последовательность, имеющую ограничение в 16 символов.

Выбранная структура проста и соответствует всем требованиям программы.

3. Описание разработки программы.

В качестве среды программирования был выбран программный продукт Microsoft Visual studio 2010

Интерфейс базы данных представляет собой меню. Перемещение по пунктам меню осуществляется при помощи цифр от 1 до 10, вводимых с клавиатуры.

При создании базы данных, после ввода каждого узла динамического списка будет выводиться сообщение о прекращении создании базы данных.

Прекращение создания осуществляется вводом с клавиатуры ``.

“Create data base” - создается база данных. Данный пункт позволяет добавить новый узел в конец существующей базы данных.

“Load”- предоставляет возможность представить базу данных в текстовом формате, создавая для этого файл. Также пользователь вводит нужное ему имя файла и разрешение, после чего произойдет его создание и запись базы данных.

“Get from file” - пункт позволяет загрузить базу данных из файла в память программы.

“Add”- пункт для обновления базы данных.

“Delete” - корректировки предусмотрена функция удаления узла.

“Database output” – пункт позволяет в любой момент использования программы выбрать его и тем самым просмотреть содержимое существующей базы данных.

“Search” - пункт дает пользователю возможность найти в базе данных необходимую информацию по определенному слову.

“Filter” - пункт осуществляет возможность отбора данных по определенному критерию.

После завершения работы с базой данных для выхода из программы существует пункт меню «Exit». После выполнения данного пункта существующая база данных будет удалена из памяти программы.

Разработанная программа состоит из нескольких модулей, которые реализовывались в следующем порядке:

- 1) Курсач.cpp – основной файл программы. Меню программы.
- 2) book.h- описание заголовков функции. Динамическая структура.
- 3) title.cpp – титульный лист.
- 4) spis.cpp – добавление новой базы данных.
- 5) sort.cpp – сортировка элементов.
- 6) search.cpp –поиск элемента списка.
- 7) del.cpp – удаление элемента из списка.
- 8) output.cpp – вывод записей в определенном формате.
- 9) load.cpp – загрузка бд в файл.
- 10) edit.cpp –редактирование базы данных.

11) download.cpp –загрузка бд из файла.

12) add.cpp – добавление новой строки.

4. Отладка и тестирование.

При создании проекта использовалась программа Microsoft Visual Studio 2019. Она имеет все необходимые для разработки и отладки программы. Для отладки программы использовались: точки остановок, анализ содержимого локальных и глобальных переменных, пошаговое выполнение кода.

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.

5. Описание программы.

После запуска программы появляется навигационное меню, содержащее пункты, которые имеют определенную нумерацию. Для выбора необходимого пункта меню доступен ввод цифры, которой соответствует определенный пункт меню. После ввода цифры осуществляется выполнение функции, то есть ожидается выполнение определенного действия, либо выводится подменю, в котором нужно выбрать необходимое действие.

Структура главного меню:

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1,Enter	Создание базы данных	Запускается диалог добавления записи и добавление записи в БД
2,Enter	Загрузка	Запускается диалог и запись существующей БД в файл.

3,Enter	Получить из файла	База загружается из файла
4,Enter	Добавить	Добавление новых строк
5,Enter	Удаление	Запускается диалог удаления записи и удаление записи из БД.
6,Enter	Вывод базы данных	Выводится окно с бд.
7,Enter	Поиск	Находит в бд необходимую информацию по определенному слову.
8,Enter	Сортировка	пункт осуществляет возможность отбора данных по определенному критерию.
9,Enter	Редактировать	пункт осуществляет возможность редактировать бд.
10,Enter	Выход	Выполняется закрытие программы.

6. Руководство пользователя

При запуске программы у нас появляется титульный лист.

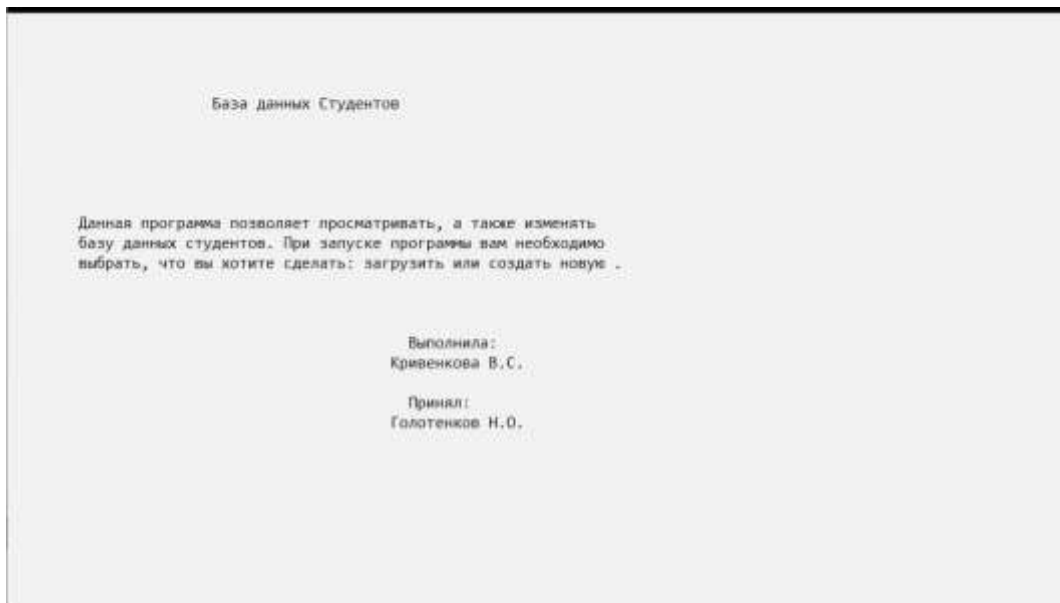


Рисунок 6. - Титульный лист

Для чтения и вывода на экран существующего списка необходимо выбрать пункт меню «Get from file », либо можно создать новую базу данных, выбрав «Create data base », и нажать клавишу Enter. Для добавления базы данных необходимо выбрать пункт меню «Добавить запись». Далее необходимо заполнить форму структуры из пяти пунктов: ввести ФИО, нажать клавишу Enter, ввести номер группы, нажать клавишу Enter, ввести факультет, нажать клавишу Enter, ввести курс, нажать клавишу Enter, ввести возраст, нажать клавишу Enter.

Для редактирования базы данных следует выбрать пункт 9.

N	Фамилия	Группа	Факультет	Курс	Возраст
1	Голотенков	10002	ФФТ	-	44
2	Кривенкова	21002	ФФТ	1	18
3	Рогашевский	21003	ФМТЕ	1	19
4	Беззаботин	19003	ФПТЭТ	3	21
5	Макарова	18ГГ1	ИЭИУ	23	23
6	Акбулатова	20Т31	ФФЕН	2	19
7	Вереметьев	173Ц	ФПИСН	5	24

Выберите строку для редактирования: 7
 Выберите поле:
 1) Фамилия
 2) Группа
 3) Факультет
 4) Курс
 5) Возраст:
 5
 Введите новое значение: 20

Рисунок 9. – Редактирование

После окончания ввода следует нажать клавишу Enter. В случае успешного выполнения операции программа выйдет в главное меню.

Для создания файла и записи в него созданной базы данных предназначен пункт меню 2

N	Фамилия	Группа	Факультет	Курс	Возраст
1	Нефедова	20001	ФФТ	2	20
2	Сагатов	21003	ФФТ	1	19
3	Сорокина	21002	ФФТ	1	19
4	Кривенкова	21002	ФФТ	1	18
5	Голотенков	10002	ФФТ	-	44
6	Кривенкова	21002	ФФТ	1	18
7	Рогашевский	21003	ФМТЕ	1	19
8	Беззаботин	19003	ФПТЭТ	3	21
9	Макарова	18ГГ1	ИЭИУ	23	23
10	Акбулатова	20Т31	ФФЕН	2	19
11	Вереметьев	173Ц	ФПИСН	5	20

Для продолжения нажмите любую клавишу . . .

Рисунок 10. - Запись базы данных в файл

Для удаления узла базы данных предназначен пункт меню под номером 5.



Рисунок 11. Удаление узла базы данных.

Пункт 7 отвечает за поиск в базе данных.

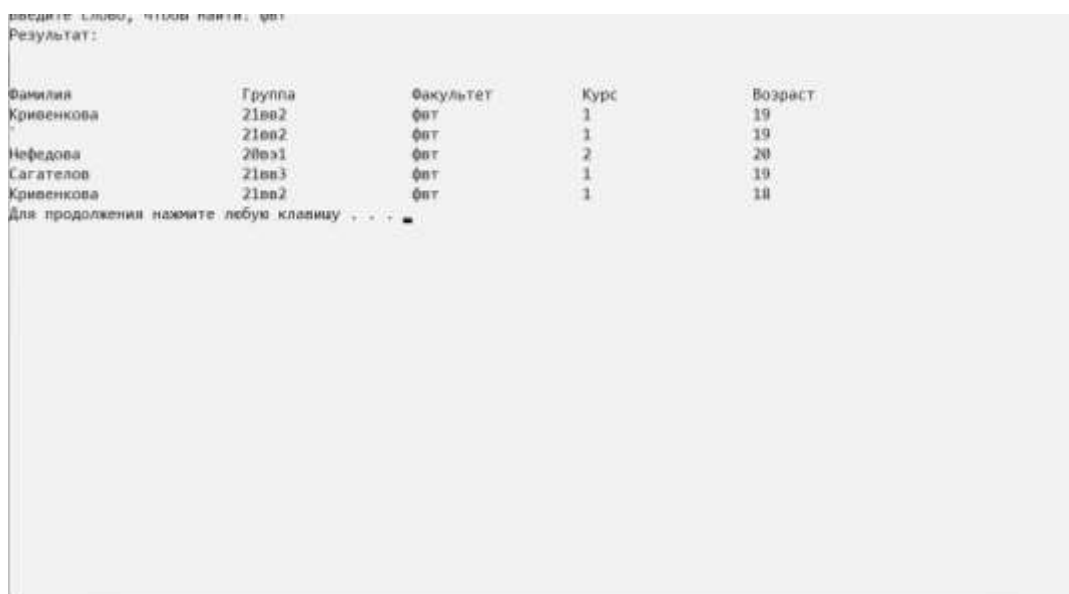


Рисунок 12. Удаление узла базы данных.

Пункт 9 в главном меню позволяет редактировать записи в базе данных.

N	Фамилия	Группа	Факультет	Курс	Возраст
1	Кривенкова	21ав2	фвт	1	19
2	Т	21ав2	фвт	1	19
3	Нефедова	20ав1	фвт	2	20
4	Сагатов	21ав3	фвт	1	19
5	Кривенкова	21ав2	фвт	1	18
6	Голотенков	100В2	ФВТ	-	44
7	Кривенкова	21ВВ2	ФВТ	1	18
8	Рогачевский	21ВВ3	ФВТЕ	1	19
9	Беззаботин	19ВВ3	ФПТЭТ	3	21
10	Макарова	18ГГ1	ИЗМУ	23	23
11	Амбулатова	20ТЗ1	ФМЕН	2	19
12	Череметьев	17ЗЩ	ФПИСН	5	20
Выберите строку для редактирования: ▾					

N	Фамилия	Группа	Факультет	Курс	Возраст
1	Кривенкова	21ав2	фвт	1	19
2	Мтировин	21ав2	фвт	1	19
3	Нефедова	20ав1	фвт	2	20
4	Сагатов	21ав3	фвт	1	19
5	Кривенкова	21ав2	фвт	1	18
6	Голотенков	100В2	ФВТ	-	44
7	Кривенкова	21ВВ2	ФВТ	1	18
8	Рогачевский	21ВВ3	ФВТЕ	1	19
9	Беззаботин	19ВВ3	ФПТЭТ	3	21
10	Макарова	18ГГ1	ИЗМУ	23	23
11	Амбулатова	20ТЗ1	ФМЕН	2	19
12	Череметьев	17ЗЩ	ФПИСН	5	22
Выберите строку для редактирования: ▾					

Рисунок 13. - Редактирование базы данных (до и после)

Пункт меню под номером 8 выполняет сортировку.

4)Сортировка по Группе
5)Сортировка по Факультету
Выберите сортировку:3

N	Фамилия	Группа	Факультет	Курс	Возраст
1	Голотенков	10002	ФФТ	-	44
2	Кривенкова	21002	ФФТ	1	19
3	Сагатов	21003	ФФТ	1	19
4	Кривенкова	21002	ФФТ	1	18
5	Кривенкова	21002	ФФТ	1	18
6	Рогашевский	21003	ФФТЕ	1	19
7	Мтирошин	21002	ФФТ	2	19
8	Нефедова	20001	ФФТ	2	20
9	Ахбулатова	20Т31	ФФЕН	2	19
10	Беззаботин	19003	ФФТЭТ	3	21
11	Шереметьев	173Ц	ФФПИСН	5	22
12	Макарова	18ГГ1	ИЭИУ	23	23

Для продолжения нажмите любую клавишу . . .

Рисунок 14. – Результат сортировки.

Последний пункт 10 осуществляет выход из программы.

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Были освоены приемы создания псевдографического интерфейса, изучены функции работы с консолью, способы обработки событий от клавиатуры. Усвоены механизмы реализации меню. Изучены функции работы с файлами. Получены базовые навыки программирования на языке Си. Изучены основные возможности среды программирования Visual Studio 2019. Получены навыки отладки и тестирования программ.

В рамках выполнения курсовой работы была написана программа, необходимая для ведения базы данных компьютерных игр. Программа предоставляет небольшой, но достаточный для использования список возможностей. Не требует большого количества системных ресурсов и обеспечивает быструю обработку базы данных.

Список используемых источников.

1. [Создание базы данных программным способом с помощью ADO.NET - Visual C++ | Microsoft Docs](#)
2. [Простая база данных на c++](#)
3. [Words are DUST: C/C++ работа с базой данных MySQL \(words-are-dust.blogspot.com\)](#)

Приложение А. Листинг программы.

Файл Курсач.срр

```
#define _CRT_SECURE_NO_WARNINGS
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
#include <locale.h>
#include <io.h>
#include <windows.h>
#include "book.h"
```

```
void main()
{
```

```
SetConsoleCP(1251);
SetConsoleOutputCP(1251);
system("color F0");
title();
system("pause");
int count;
game_list* head = NULL;
for (;;)
{
    system("cls");
    char menu[] = ("
1.Создание базы данных      | \n ----- \n
| 2.Загрузка                | \n ----- \n
| 3.Получить из файла       | \n ----- \n
| 4.Добавить                 | \n ----- \n
| 5.Удаление                 | \n ----- \n
| 6.Вывод базы данных       | \n ----- \n
| 7.Поиск                    | \n ----- \n
\n | 8.Сортировка              | \n ----- \n
\n | 9.Редактировать           | \n ----- \n
\n | 10.Выход                  | \n ----- \n
----- \n \n");
    __asm
    {
        xor eax, eax
        lea eax, menu
        push eax
        call dword ptr[printf]
        add esp, 4
    }
    printf("\n Введите номер операции:");
    scanf("%d", &count);
    switch (count)
    {
    case 1:
        system("cls");
        spisok(&head);
```

```

        break;
    case 2:
        system("cls");
        load(&head);
        break;
    case 3:
        system("cls");
        download(&head);
        system("pause");
        break;
    case 4: {
        system("cls");
        spisok(&head);
        system("pause");
        break;
    }
    case 5:
        system("cls");
        del(&head);
        system("pause");
        break;
    case 6:
        system("cls");
        fflush(stdin);
        pri(&head);
        system("pause");
        break;
    case 7:
        system("cls");
        search(&head);
        system("pause");
        break;
    case 8:
        system("cls");
        sort(&head);
        system("pause");
        break;
    case 9:
        system("cls");
        edit(&head);
        pri(&head);
        break;
    case 10:
        exit(0);
        break;
    }
}
}

```

Файл add.cpp

```

#include "stdafx.h"
#include "book.h"
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <locale.h>
#include <io.h>
#include <Windows.h>

void push(game_list** head, char* fio, char* group, char* faculty, char* course, char*
age)
{

```

```

    game_list* tmp;
    tmp = new game_list;
    strcpy_s(tmp->fio, fio);
    strcpy_s(tmp->group, group);
    strcpy_s(tmp->faculty, faculty);
    strcpy_s(tmp->course, course);
    strcpy_s(tmp->age, age);
    tmp->next = (*head);
    (*head) = tmp;
}

```

Файл book.h

```
#pragma once
```

```

typedef struct game_list
{
    char fio[16],
        group[16],
        faculty[16],
        course[16],
        age[16];
    game_list* next;
};

```

```

void sort(game_list** head);
void spisok(game_list** head);
void download(game_list** head);
void title();
void del(game_list** head);
void search(game_list** head);
void edit(game_list** head);
void load(game_list** head);
void download(game_list** head);
void pri(game_list** head);
void push(game_list** head, char* nazvanie, char* partW, char* capital, char* population,
char* square);

```

Файл del.cpp

```

#include "stdafx.h"
#include "book.h"
#include <Windows.h>

```

```

void del(game_list** head)
{
    system("cls");
    printf("\nВведите номер строки которую хотите удалить\n");
    int n;
    scanf("%d", &n);

    if ((*head) != NULL)
    {
        if (n - 1 == 0) {
            game_list* prev = (*head);
            (*head) = (*head)->next;
            //если самый первый элемент то мы сдвигаем голову на один шаг, а
первый элемент удаляем
            delete prev;
        }
        else {
            game_list* previous = (*head);
            for (int i = 0; i < n - 1; i++) {
                previous = previous->next;
            }
            game_list* todelete = previous->next;

```

```

        previous->next = todelete->next;
        delete todelete;
    }
}
else
{
    printf("\nСписок пуст!\n");
}
}

```

Файл del.cpp

```

#include "stdafx.h"
#include "book.h"

void download(game_list** head)
{
    int i;
    FILE* S;
    char name[32], fio[32], group[32], faculty[32], course[32], age[32];
    int fd;
    *head = NULL;
    fprintf(stdout, "%s", "Введите имя файла:\n");
    scanf("%s", &name);
    while ((S = fopen(name, "r+")) == (NULL))
    {
        fprintf(stdout, "%s", "Файл не найден!\nПопробуйте еще раз:");
        scanf("%s", &name);
    }
    fprintf(stdout, "Файл %s найден правильно!\n", name);
    if (!feof(S))
    {
        fseek(S, 100, SEEK_SET);
        while (!feof(S))
        {
            fscanf(S, "%s", &fio);
            fscanf(S, "%s", &group);
            fscanf(S, "%s", &faculty);
            fscanf(S, "%s", &course);
            fscanf(S, "%s", &age);
            if (!feof(S))
                push(head, fio, group, faculty, course, age);
        }
    }
    else
    {
        printf("\nПУСТО!\n\n");
    }
    fclose(S);
}

```

Файл edit.cpp

```

#include "stdafx.h"
#include "book.h"
#include <Windows.h>

void edit(game_list** head)
{
    game_list* ptr = (*head);
    int i = 0;
    char name[16];
    pri(head);
    if (ptr != NULL)

```

```

{
    ptr = (*head);
    printf("Выберите строку для редактирования:");
    scanf("%d", &i);
    i--;
    for (; i > 0; i--)
    {
        ptr = ptr->next;
        if (ptr == NULL)
        {
            exit(-1);
            printf("Ошибка при вводе!");
        }
    }
    printf("Выберите поле \n 1)Фамилия \n 2) Группа \n 3) Факультет \n 4) Курс
\n 5) Возраст \n");
    scanf("%d", &i);
    fflush(stdin);
    printf("Введите новое значение: ");
    gets(name);
    switch (i)
    {
        case 1:
            strcpy(ptr->fio, name);
            break;
        case 2:
            strcpy(ptr->group, name);
            break;
        case 3:
            strcpy(ptr->faculty, name);
            break;
        case 4:
            strcpy(ptr->course, name);
            break;
        case 5:
            strcpy(ptr->age, name);
            break;
        default: break;
    }

}
else {
    printf("ПУСТО!");
}
}

```

Файл load.cpp

```

#include "stdafx.h"
#include "book.h"

void load(game_list** head)
{
    FILE* S;
    game_list* ptr = (*head);
    char name[32];
    fflush(stdin);
    fprintf(stdout, "%s", "Введите имя файла:");
    scanf("%s", &name);
    while ((S = fopen(name, "w")) == (NULL))
    {
        fprintf(stdout, "%s", "Файл не найден!\n");
        scanf("%s", &name);
    }
    fprintf(stdout, "Файл %s был найден успешно!\n", name);
}

```

```

    if ((*head) != NULL)
    {
        game_list* ptr = *head;
        fprintf(S, "\n\n%-25s %-18s %-18s %-18s %-20s\n", "Фамилия", "Группа",
"Факультет", "Курс", "Возраст");
        while (ptr != NULL)
        {
            fprintf(S, "%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group,
ptr->faculty, ptr->course, ptr->age);
            ptr = ptr->next;
        }
    }
    else
    {
        printf("ПУСТО\n\n");
    }
    fclose(S);
}

```

Файл output.cpp

```

#include "stdafx.h"
#include "book.h"

void pri(game_list** head)
{
    if (*head != NULL)
    {
        int i = 1;
        game_list* ptr = (*head);
        ptr = (*head);
        printf("N      %-30s %-20s %-20s %-20s %-22s\n", "Фамилия", "Группа",
"Факультет", "Курс", "Возраст");
        while (ptr != NULL)
        {
            printf("%-5d %-30s %-20s %-20s %-20s %-22s\n", i, ptr->fio, ptr-
>group, ptr->faculty, ptr->course, ptr->age);
            ptr = ptr->next;
            i++;
        }
    }
    else
    {
        printf("Пусто");
    }
}

```

Файл search.cpp

```

#include "stdafx.h"
#include "book.h"
#include <string.h>

void search(game_list** head)
{
    char name[16];
    game_list* ptr = (*head);
    fflush(stdin);
    printf("Введите слово, чтобы найти: ");
    gets(name);
    if (ptr == NULL)
        printf("ПУСТО");
    else {
        printf("Результат:");
        printf("\n\n%-25s %-18s %-18s %-18s %-20s\n", "Фамилия", "Группа",
"Факультет", "Курс", "Возраст");
    }
}

```

```

    }
    while (ptr != NULL)
    {
        if (strstr(ptr->fio, name) != NULL)
            printf("%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group, ptr->
>faculty, ptr->course, ptr->age);
        if (strstr(ptr->group, name) != NULL)
            printf("%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group, ptr->
>faculty, ptr->course, ptr->age);
        if (strstr(ptr->faculty, name) != NULL)
            printf("%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group, ptr->
>faculty, ptr->course, ptr->age);
        if (strstr(ptr->course, name) != NULL)
            printf("%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group, ptr->
>faculty, ptr->course, ptr->age);
        if (strstr(ptr->age, name) != NULL)
            printf("%-25s %-18s %-18s %-18s %-20s\n", ptr->fio, ptr->group, ptr->
>faculty, ptr->course, ptr->age);
        ptr = ptr->next;
    }
}

```

Файл sort.cpp

```

#define _CRT_SECURE_NO_WARNINGS
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
#include <locale.h>
#include <io.h>
#include <Windows.h>
#include "book.h"

void sort_age(game_list** head)
{
    char fio[16], group[16], faculty[16], course[16], age[16];
    int size = 0;
    game_list* ptr = (*head);
    while (ptr != NULL)
    {
        ptr = ptr->next;
        size++;
    }
    ptr = (*head);
    for (int j = 1; j < size; j++)
    {
        ptr = (*head);
        for (int i = 0; i < size - j; i++)
        {
            if (atoi(ptr->age) > atoi(ptr->next->age))
            {
                strcpy(fio, ptr->fio);
                strcpy(group, ptr->group);
                strcpy(faculty, ptr->faculty);
                strcpy(course, ptr->course);
                strcpy(age, ptr->age);

                strcpy(ptr->fio, ptr->next->fio);
                strcpy(ptr->group, ptr->next->group);
                strcpy(ptr->faculty, ptr->next->faculty);
                strcpy(ptr->course, ptr->next->course);
                strcpy(ptr->age, ptr->next->age);
            }
        }
    }
}

```

```

        strcpy(ptr->next->fio, fio);
        strcpy(ptr->next->group, group);
        strcpy(ptr->next->faculty, faculty);
        strcpy(ptr->next->course, course);
        strcpy(ptr->next->age, age);
    }
    ptr = ptr->next;
}
}

void sort_course(game_list** head)
{
    char fio[16], group[16], faculty[16], course[16], age[16];
    int size = 0;
    game_list* ptr = (*head);
    while (ptr != NULL)
    {
        ptr = ptr->next;
        size++;
    }
    ptr = (*head);
    for (int j = 1; j < size; j++)
    {
        ptr = (*head);
        for (int i = 0; i < size - j; i++)
        {
            if (atoi(ptr->course) > atoi(ptr->next->course))
            {
                strcpy(fio, ptr->fio);
                strcpy(group, ptr->group);
                strcpy(faculty, ptr->faculty);
                strcpy(course, ptr->course);
                strcpy(age, ptr->age);

                strcpy(ptr->fio, ptr->next->fio);
                strcpy(ptr->group, ptr->next->group);
                strcpy(ptr->faculty, ptr->next->faculty);
                strcpy(ptr->course, ptr->next->course);
                strcpy(ptr->age, ptr->next->age);

                strcpy(ptr->next->fio, fio);
                strcpy(ptr->next->group, group);
                strcpy(ptr->next->faculty, faculty);
                strcpy(ptr->next->course, course);
                strcpy(ptr->next->age, age);
            }
            ptr = ptr->next;
        }
    }
}

void sort_fio(game_list** head)
{
    char fio[16], group[16], faculty[16], course[16], age[16];
    int size = 0;
    game_list* ptr = (*head);
    while (ptr != NULL)
    {
        ptr = ptr->next;
        size++;
    }
    ptr = (*head);
    for (int j = 1; j < size; j++)
    {

```

```

ptr = (*head);
for (int i = 0; i < size - j; i++)
{
    if (ptr->fio[0] > ptr->next->fio[0])
    {
        strcpy(fio, ptr->fio);
        strcpy(group, ptr->group);
        strcpy(faculty, ptr->faculty);
        strcpy(course, ptr->course);
        strcpy(age, ptr->age);

        strcpy(ptr->fio, ptr->next->fio);
        strcpy(ptr->group, ptr->next->group);
        strcpy(ptr->faculty, ptr->next->faculty);
        strcpy(ptr->course, ptr->next->course);
        strcpy(ptr->age, ptr->next->age);

        strcpy(ptr->next->fio, fio);
        strcpy(ptr->next->group, group);
        strcpy(ptr->next->faculty, faculty);
        strcpy(ptr->next->course, course);
        strcpy(ptr->next->age, age);
    }
    ptr = ptr->next;
}
}

void sort_faculty(game_list** head)
{
    char fio[16], group[16], faculty[16], course[16], age[16];
    int size = 0;
    game_list* ptr = (*head);
    while (ptr != NULL)
    {
        ptr = ptr->next;
        size++;
    }
    ptr = (*head);
    for (int j = 1; j < size; j++)
    {
        ptr = (*head);
        for (int i = 0; i < size - j; i++)
        {
            if (ptr->faculty[0] > ptr->next->faculty[0])
            {
                strcpy(fio, ptr->fio);
                strcpy(group, ptr->group);
                strcpy(faculty, ptr->faculty);
                strcpy(course, ptr->course);
                strcpy(age, ptr->age);

                strcpy(ptr->fio, ptr->next->fio);
                strcpy(ptr->group, ptr->next->group);
                strcpy(ptr->faculty, ptr->next->faculty);
                strcpy(ptr->course, ptr->next->course);
                strcpy(ptr->age, ptr->next->age);

                strcpy(ptr->next->fio, fio);
                strcpy(ptr->next->group, group);
                strcpy(ptr->next->faculty, faculty);
                strcpy(ptr->next->course, course);
                strcpy(ptr->next->age, age);
            }
            ptr = ptr->next;
        }
    }
}

```

```

    }
}

void sort_group(game_list** head)
{
    char fio[16], group[16], faculty[16], course[16], age[16];
    int size = 0;
    game_list* ptr = (*head);
    while (ptr != NULL)
    {
        ptr = ptr->next;
        size++;
    }
    ptr = (*head);
    for (int j = 1; j < size; j++)
    {
        ptr = (*head);
        for (int i = 0; i < size - j; i++)
        {
            if (ptr->group[0] > ptr->next->group[0])
            {
                strcpy(fio, ptr->fio);
                strcpy(group, ptr->group);
                strcpy(faculty, ptr->faculty);
                strcpy(course, ptr->course);
                strcpy(age, ptr->age);

                strcpy(ptr->fio, ptr->next->fio);
                strcpy(ptr->group, ptr->next->group);
                strcpy(ptr->faculty, ptr->next->faculty);
                strcpy(ptr->course, ptr->next->course);
                strcpy(ptr->age, ptr->next->age);

                strcpy(ptr->next->fio, fio);
                strcpy(ptr->next->group, group);
                strcpy(ptr->next->faculty, faculty);
                strcpy(ptr->next->course, course);
                strcpy(ptr->next->age, age);
            }
            ptr = ptr->next;
        }
    }
}

```

```

void sort(game_list** head)
{
    int a;
    printf("1) Сортировка по Фамилии\n2) Сортировка по Возрасту \n3)Сортировка по Курсу\n4)Сортировка по Группе\n5)Сортировка по Факультету\n Выберите сортировку:");
    scanf("%d", &a);
    switch (a)
    {
        case 1:
            sort_fio(head);
            pri(head);
            system("pause");
            break;
        case 2:
            sort_age(head);
            pri(head);
            system("pause");
            break;
        case 3:
            sort_course(head);
    }
}

```

```

        pri(head);
        system("pause");
        break;
    case 4:
        sort_group(head);
        pri(head);
        system("pause");
        break;
    case 5:
        sort_faculty(head);
        pri(head);
        system("pause");
        break;
    default:
        break;
}
}

```

Файл spis.cpp

```

#include "stdafx.h"
#include "book.h"

void spisok(game_list** head)
{
    fflush(stdin);
    char fio[32], group[32], faculty[32], course[32], age[32];
    printf("Введите Фамилию:");
    gets(fio);
    for (;;)
    {
        fflush(stdin);
        printf("Введите Группу:");
        gets(group);
        printf("Введите Факультет:");
        gets(faculty);
        printf("Введите Курс:");
        gets(course);
        printf("Введите Возраст:");
        gets(age);
        push(head, fio, group, faculty, course, age);
        printf("Для завершения введите:  \nВведите Фамилию:");
        gets(fio);
        if (fio[0] == '\0')
            break;
    }
}

```

Файл title.cpp

```

#include "stdafx.h"
#include "book.h"
#include <conio.h>

void title()
{
    printf("\n\n\n\n");
    printf("                                База данных Студентов  \n");

    printf("\n\n\n\n\n");

    printf("                                Данная программа позволяет просматривать, а также изменять\n");
    printf("                                базу данных студентов. При запуске программы вам необходимо\n");
    printf("                                выбрать, что вы хотите сделать: загрузить или создать новую\n");
    printf("\n\n\n\n");
}

```

```

printf("
printf("

printf("

printf("
    _getch();
}

```

```

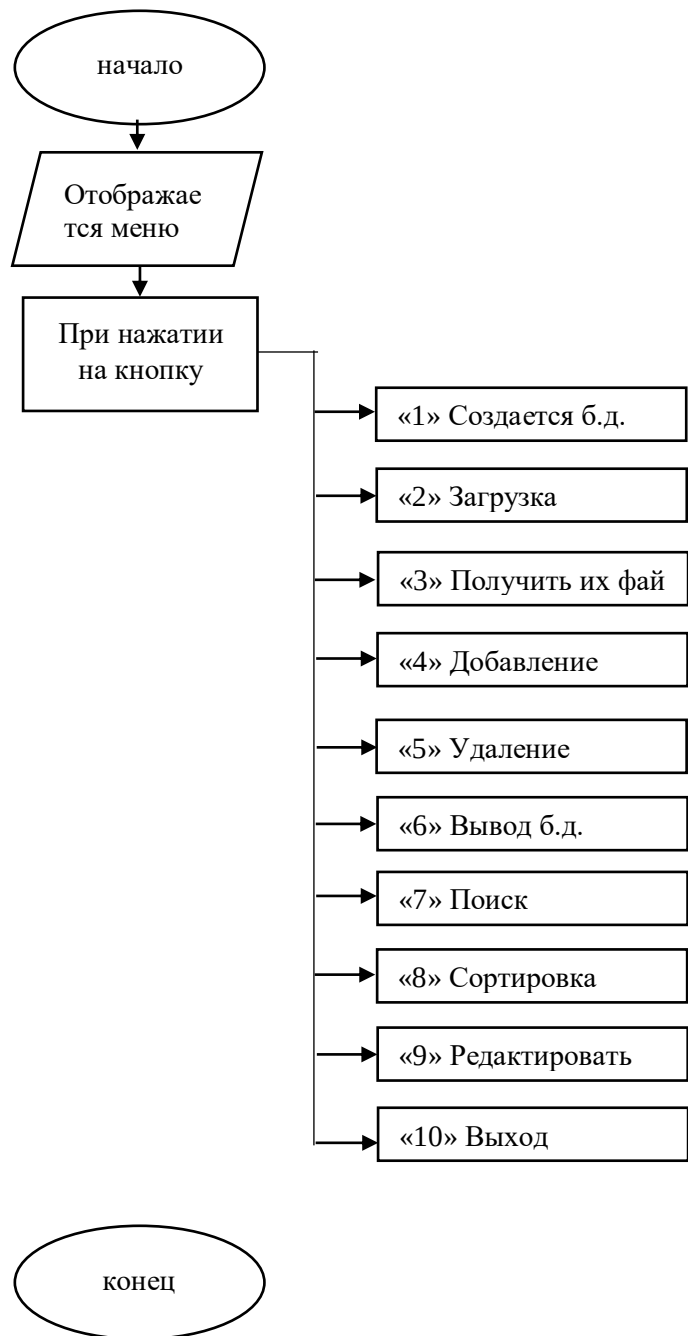
Выполнила:\n");
Кривенкова В.С.\n\n");

Принял:\n");
Голотенков Н.О.\n");

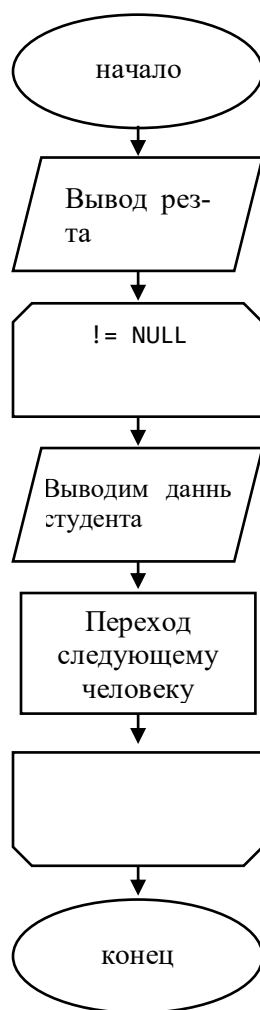
```

Приложение В. Схемы функций программы.

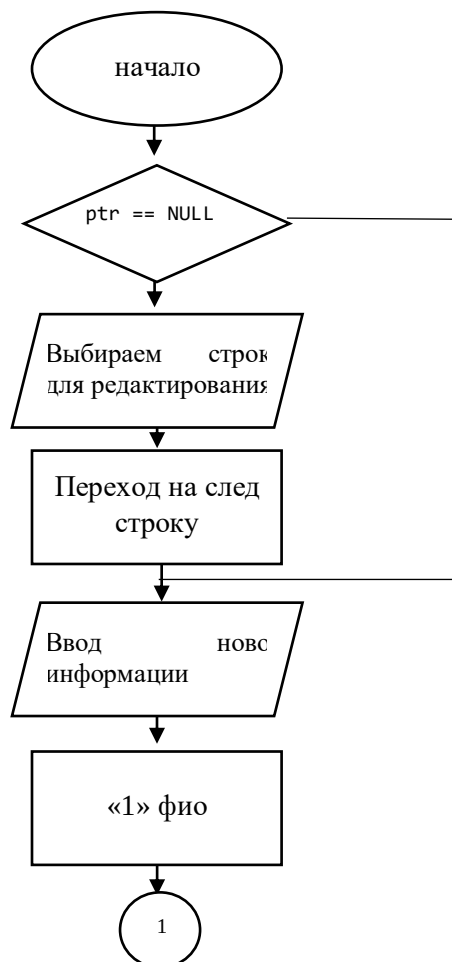
Функция main

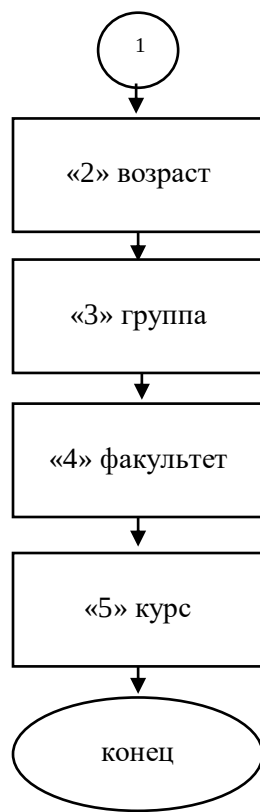


Функция output



Функция edit





Функция spis

