

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

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по дисциплине «Программирование»
на тему «База данных. Студенты»

Выполнил:

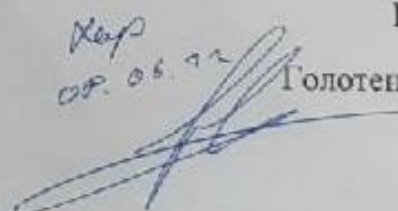
студент группы 21ВВ2

Щеглов Д.А.

Принял:

Голотенков Н.О.

Кур
09.06.22



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 Щеглову Дмитрию Александровичу Группа 74ВВ2

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

Исходные данные (технические требования) на проектирование

База данных по содержанию

Функции: "Создание новой базы данных", "Добавление записей",

"Удаление записей", "Редактирование записей"

Поиск и сортировка записей, "Вывод записей" по определенному критерию

Сохранение информации базы данных в файл. Загрузка информации 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 " 06 " 06. 2022

Руководитель Голосевский Н.О.

Задание получил " 22 " февраля 2022 г.

Студент Щербаков Дмитрий Александрович

Содержание

Введение	5
Постановка задачи.	6
Определение функционала программы.....	7
Определение необходимых процедур программы	7
Описание разработки программы	8
Отладка и тестирование	9
Блок-схема программы	9
Руководство пользователя	22
Заключение.....	33

Введение

Способности и возможности людей обрабатывать информацию ограничены, особенно в условиях возрастающих объёмов информации, поэтому появилась потребность использовать способы хранения, обработки и передачи информации (информационные технологии), отчуждённые (удалённые) от одушевлённого носителя – человека.

Любая технология связана с выполнением определённых операций и процессов, с изменением качества, формы, состояния и содержания материала, объекта и т.п.

Например, простейшим видом технологии является доставка почтальоном почтовых отправлений.

Информационные технологии – это методы и способы, использующие компьютерные программно-технические средства, отдельные или совокупные информационные процессы и операции для достижения поставленных целей.

Также существует другое определение:

Информационные технологии – это совокупность программно-технических средств вычислительной техники (СВТ), приёмов, способов и методов их применения, предназначенных для сбора, хранения, обработки, передачи и использования информации в конкретных предметных областях.

ИТ предназначены для снижения трудоёмкости процессов использования информационных ресурсов.

Постановка задачи.

Необходимо разработать программу «Студенты». Для этого нужно обозначить поля, которые будут использованы в структуре данных.

Для реализации данного пункта необходимо определить структуру базы данных, максимальный размер каждой записи базы данных. Она будет состоять из 6 ячеек, в которых будут содержаться следующие пункты: ID, ФИО, группа, курс, год поступления, год выпуска. Это необходимо для выявления и минимизации ограничений программы.

Программа должна предоставить возможность добавления, редактирования и удаления новой записи в базу данных. Для реализации этих требований нужно разработать интуитивно понятный интерфейс пользователя, определить названия пунктов меню. Также необходимо определить тип интерфейса, продумать навигацию по базе данных изучить функции для работы с файлами. Это необходимо для четкого понимания функциональности программы.

Многомодульность программы. Необходимо поделить программу на логические модули. Это повысит быстродействие программы, а также облегчит поиск ошибок.

Использование сложных типов данных – массивов, структур, файлов. Это необходимо для структуризации базы данных и при обработке данных в коде программы.

Программа должна работать в графическом или текстовом режиме видеоподсистемы. Необходимо определиться с типом интерфейса и с элементами управления. В данном случае будет использована консоль Windows.

Устройства ввода – вывода – клавиатура и мышь. Необходимо научиться анализировать и различать события, возникающие от работы с клавиатурой и мышью. Необходимо однозначно идентифицировать и выполнять те или иные действия в зависимости от действия пользователя. Это необходимо для легкого использования программы.

Пользовательский интерфейс должен быть построен на основе меню и панели инструментов. Это необходимо для создания интуитивно понятного интерфейса.

Определение функционала программы.

Определение необходимых процедур программы

Разработанная программа состоит из нескольких процедур:

Создание новой базы данных.

1. Для создания используется процедура `creation()`. Она создаёт или открывает файл и записывает данные, введенные с клавиатуры.

2. Ввод, редактирование и удаление данных.

Осуществляется с помощью использования созданных процедур `add_end()`, `deleter()`, `redact()`. Поскольку используются разные типы данных, проведена обработка ошибок, связанных с несоответствием типов данных.

2. Сортировка различных типов данных.

Чтобы пользователь мог сортировать студентов не только по числовым, но и по текстовым данным, были созданы процедура `sortirovka()`, которая отвечает за упорядочивание данных по возрастанию.

3. Поиск данных.

Для поиска записей была создана процедура `search()` для поиска по двум критериям: ФИО, Группа. Во время создания процедур была использована команда «`strstr`».

4. Фильтрация записей.

Осуществляется с помощью использования процедуры `filter()`. Фильтрация проводится по числовым данным. Для удобства можно задать критерии фильтрации как диапазоном чисел, так и условиям `>`, `<` или `=`.

5. Загрузка из файла и запись в файл.

Если пользователь захочет сохранить данные о студенте в файл, то он может вызвать процедуру `upload()`. Она сохраняет все данные в файл в формате `txt`. Для загрузки данных из уже созданного `txt` файла служит процедура `download()`. Она отвечает за загрузку файла с готовой базой автомобилей.

Описание разработки программы

Для написания данной программы будет использован язык программирования Си. Этот язык является распространённым языком программирования. Обладая простой языковой базой, он может работать с файлами и производить сложные математические операции. Это необходимо для осуществления поставленной задачи. Язык имеет низкоуровневую природу, а это значит, что возможна переносимость проекта на другой язык программирования.

Целью создателей Си была разработка мобильного языка, который можно было бы использовать для разработки системного программного обеспечения. Дополнительное свойство языка Си позволяет программисту изолировать машинно-зависимые операторы и контролировать их компиляцию в другой среде. Это дополнительно повышает мобильность программного обеспечения, написанного на данном языке.

В качестве среды программирования был выбран программный продукт Visual Studio 2010.

Интерфейс информационной системы будет представлен в виде консольного меню. Оно является наиболее удобным и обеспечивает наименьшее время доступа к данным. Перемещение по пунктам меню будет осуществляться с помощью ввода ответов на запросы с клавиатуры.

Исходя из того, что при первом знакомстве с программой пользователю нужно дать краткую информацию о её особенностях, в программе будут описан интерфейс управления и функции.

Для завершения работы предусмотрен пункт «Закончить работу». При завершении работы приходит запрос на сохранение данных.

Отладка и тестирование

В качестве среды разработки была выбрана программа Visual Studio 2010. Программа предоставляет все средства необходимые при разработке и отладке разрабатываемых модулей и программ. Для отладки использовались такие инструменты как точка останова, трассировка, анализ содержимого переменных, анализ содержимого памяти.

Тестирование проводилось в рабочем порядке, в процессе разработки и после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных в основном с сортировкой, ошибками переполнения или выхода за границу допустимого значения, ошибками несоответствия типа.

Блок-схема программы

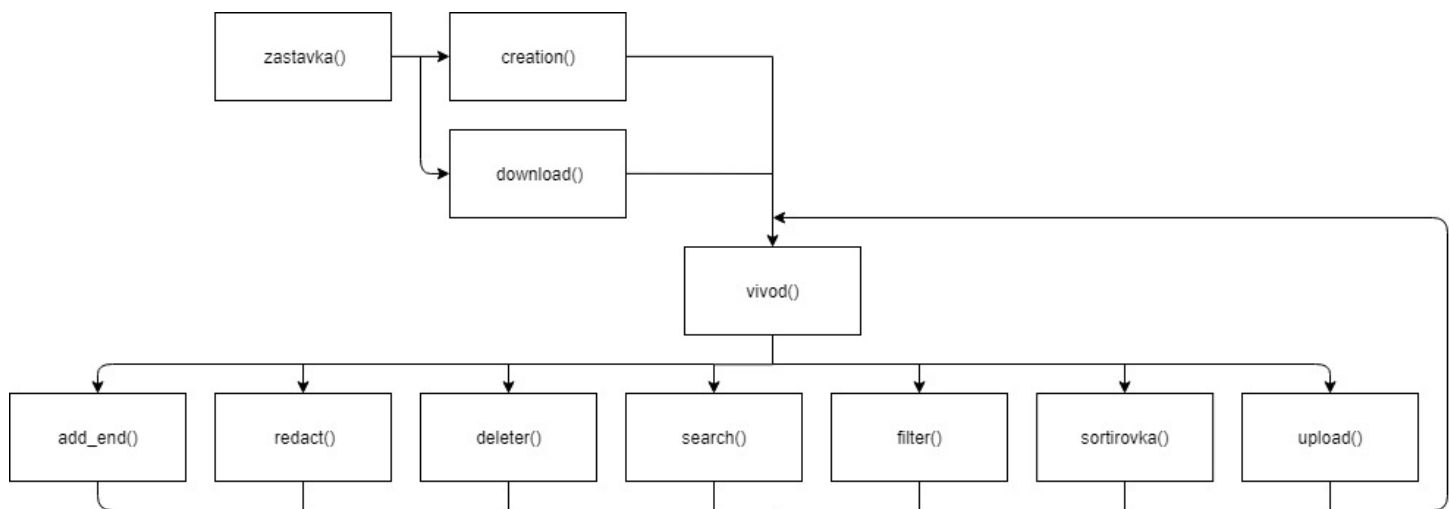


Рисунок 1 - иерархическая структура программы

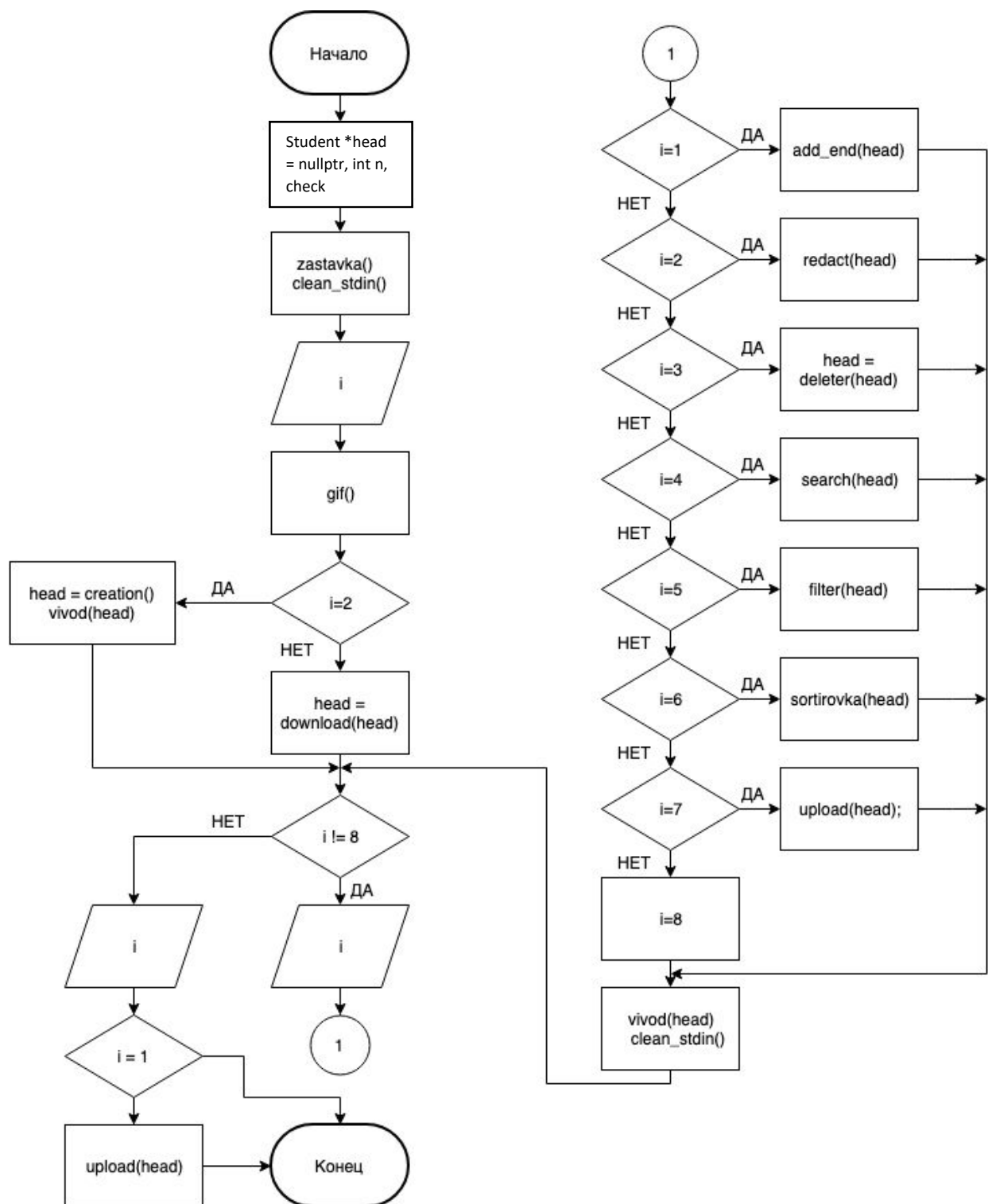


Рисунок 2 – функция main()

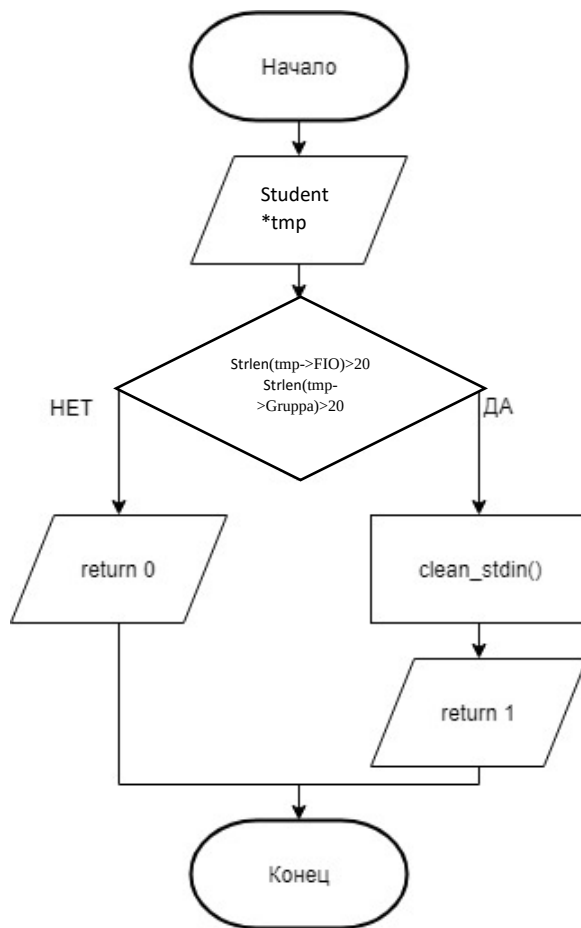


Рисунок 3 - функция для проверки переполнения

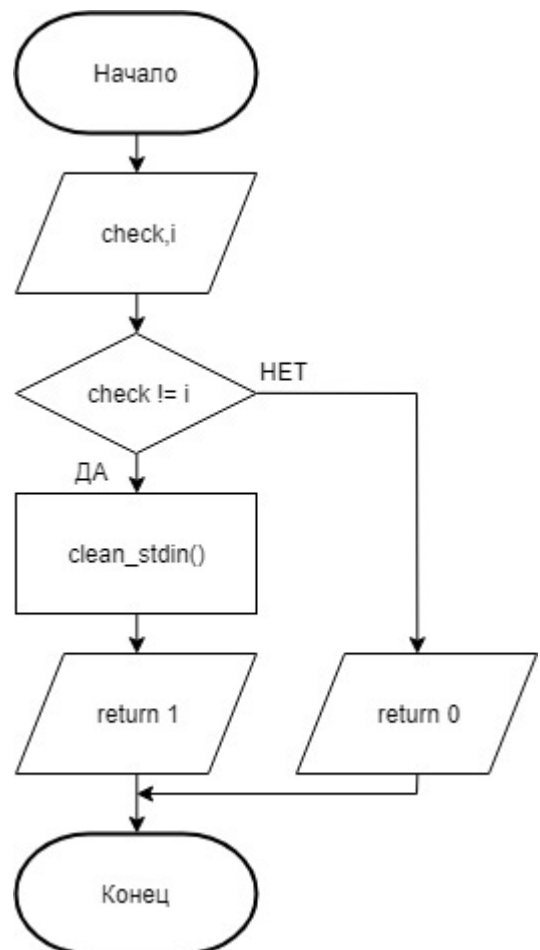


Рисунок 4 - функция для проверки на соответствие типа

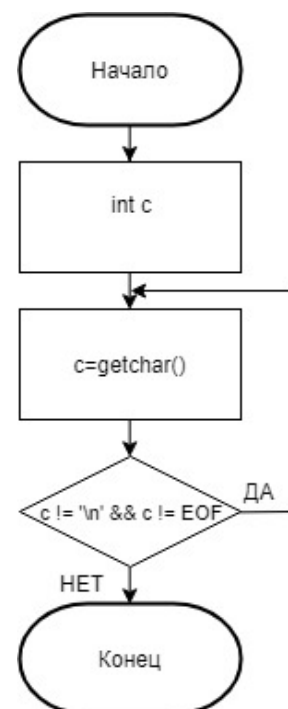


Рисунок 5 - функция для очистки буфера ввода

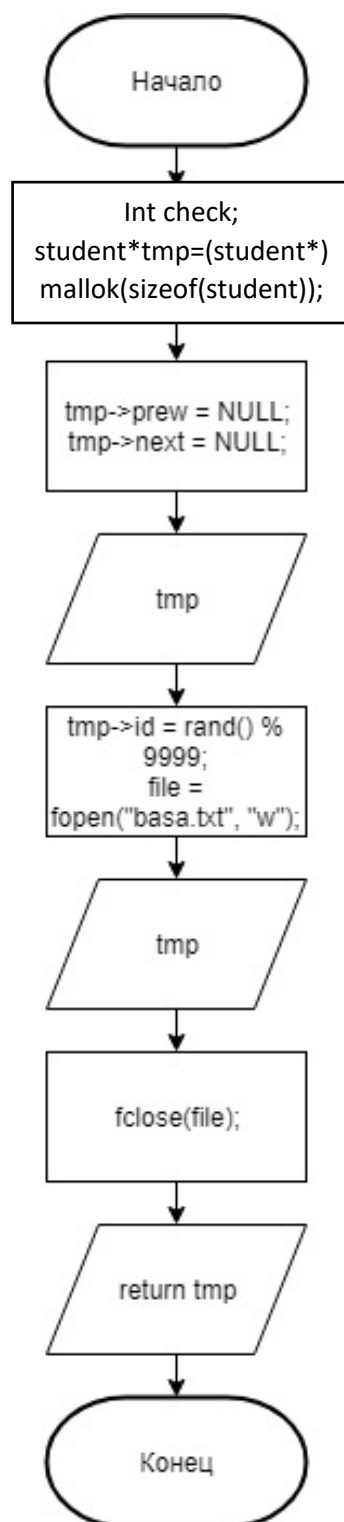


Рисунок 6 - функция для создания новой базы данных

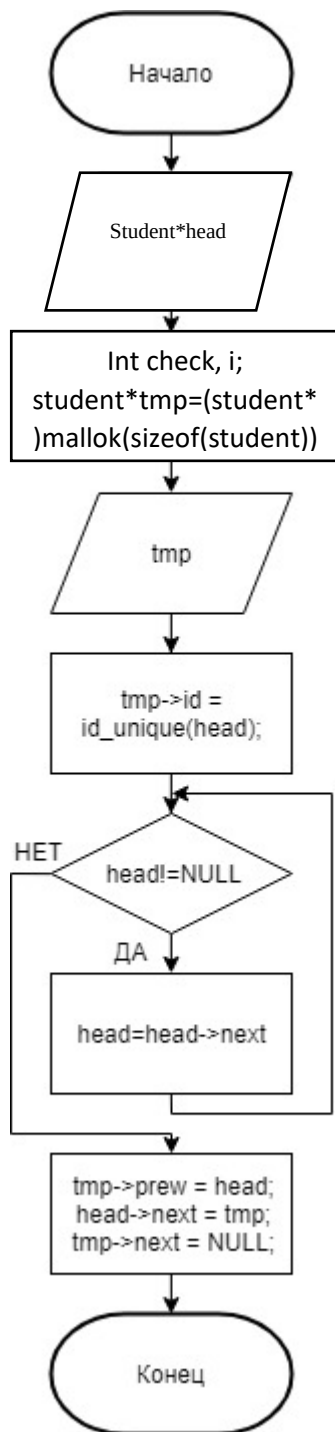


Рисунок 7 - функция для добавления элемента

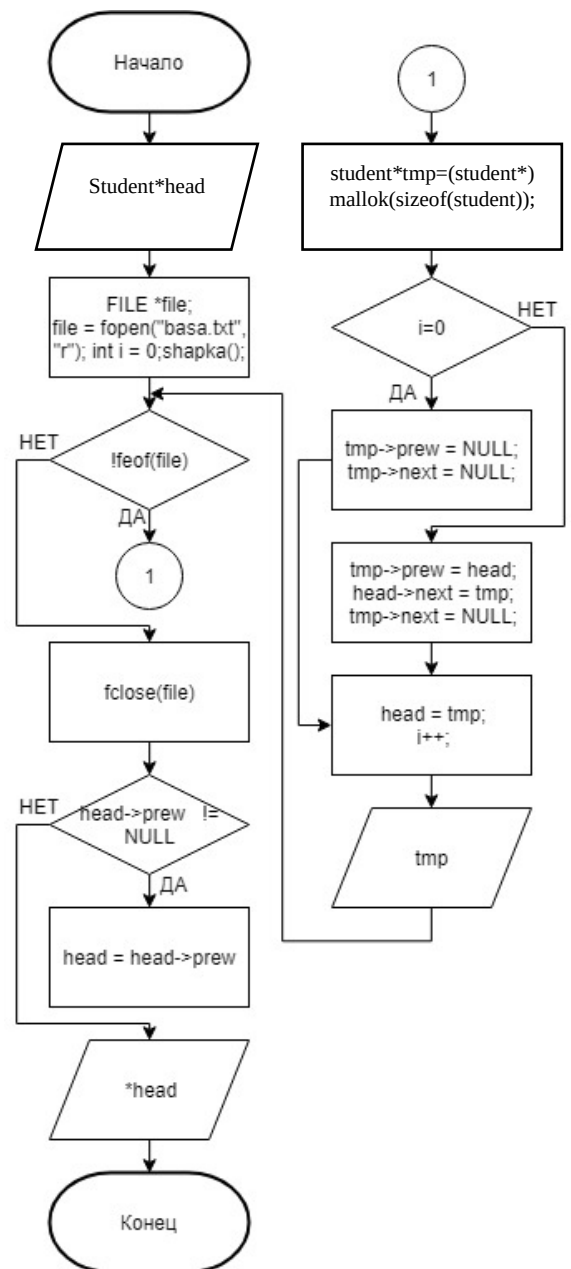


Рисунок 8 - функция для загрузки базы данных из файла

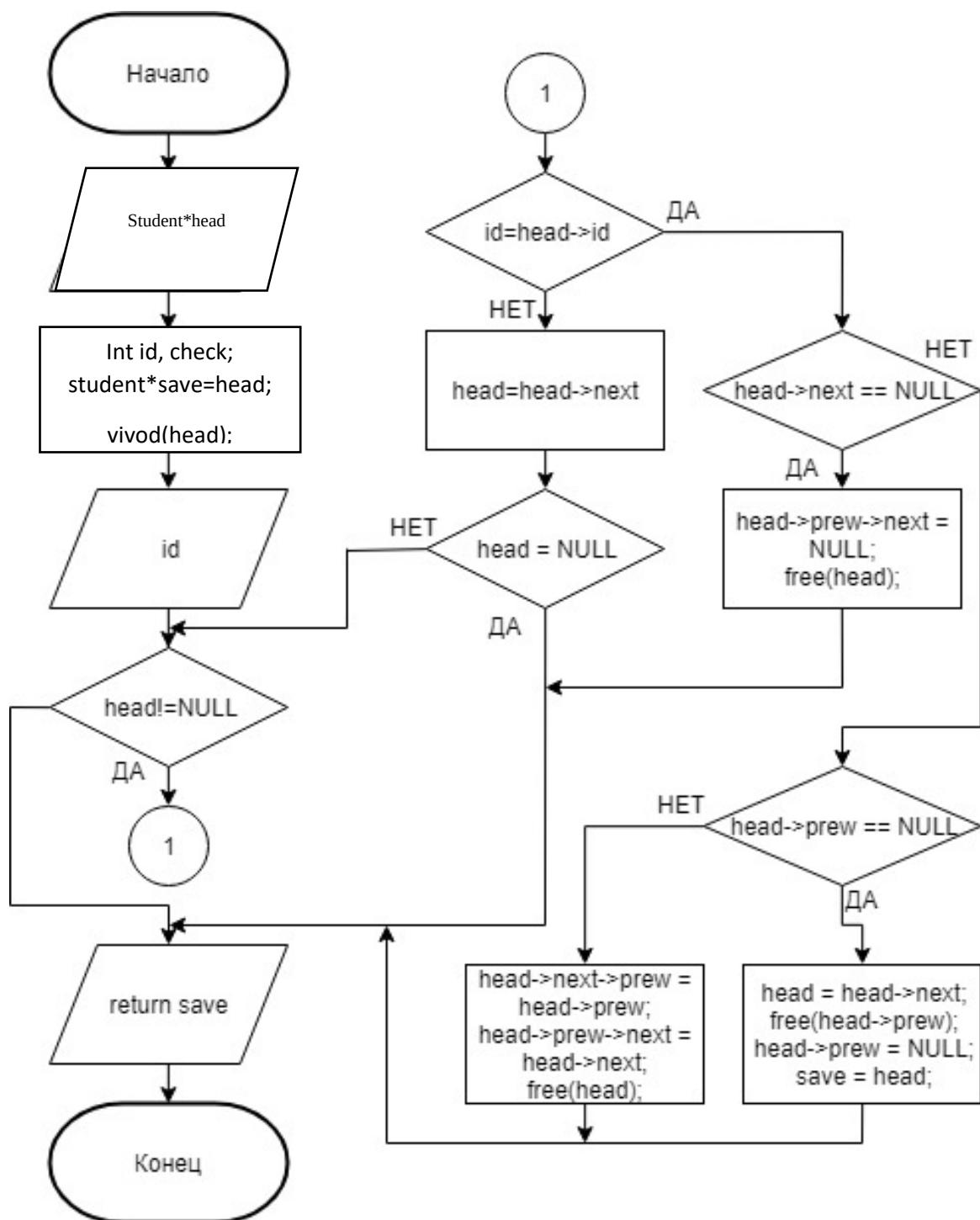


Рисунок 9 - функция для удаления элемента

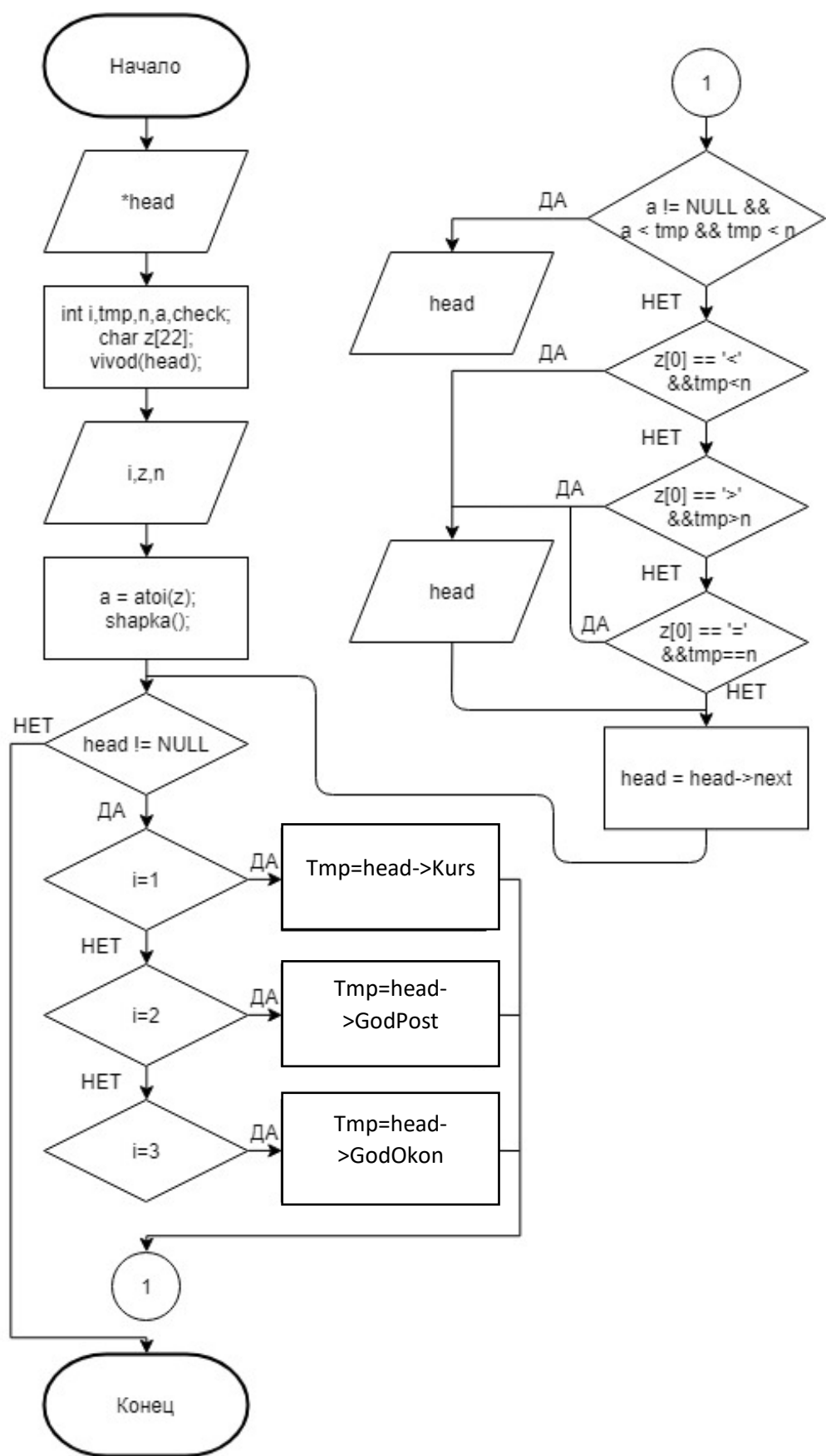


Рисунок 10 - функция фильтрации

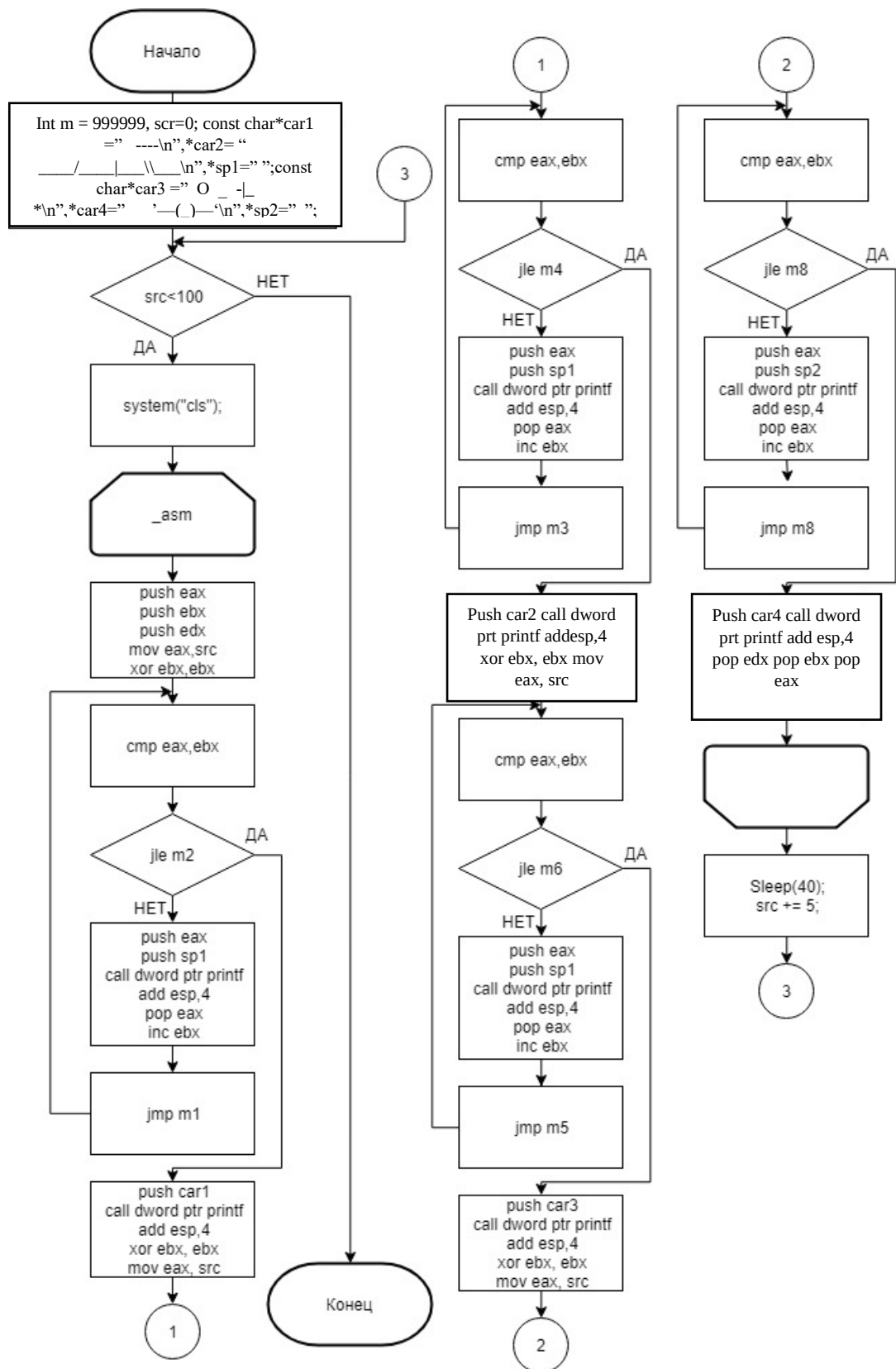


Рисунок 11 - функция анимации

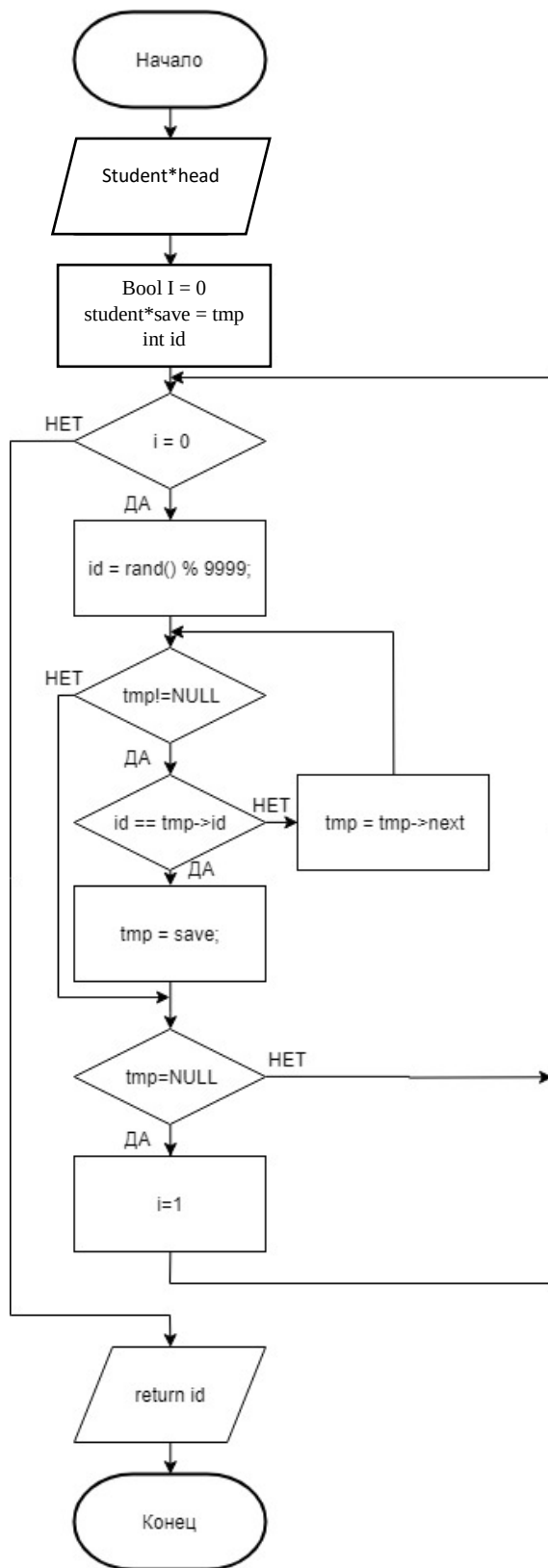


Рисунок 12 - функция генерирования id

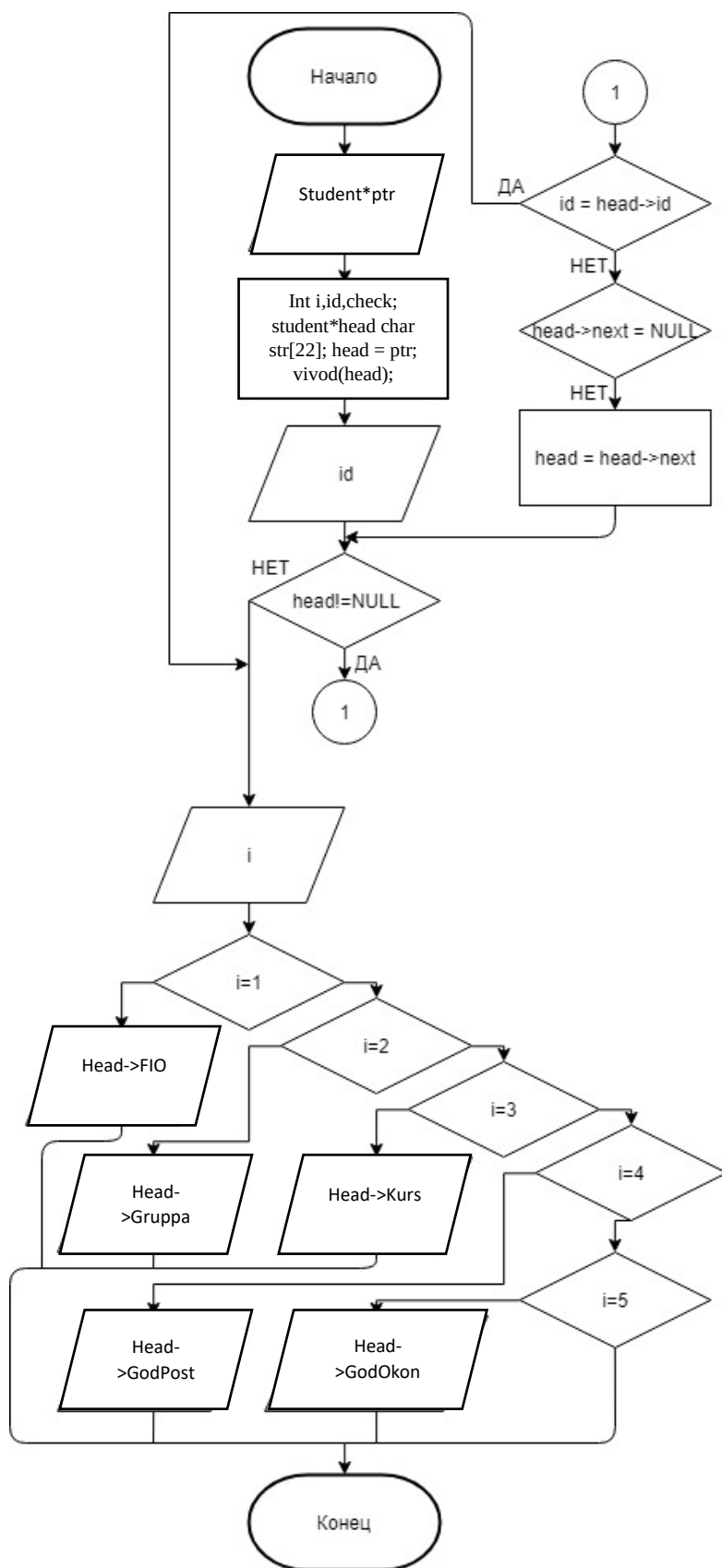


Рисунок 13 - функция редактирования элементов

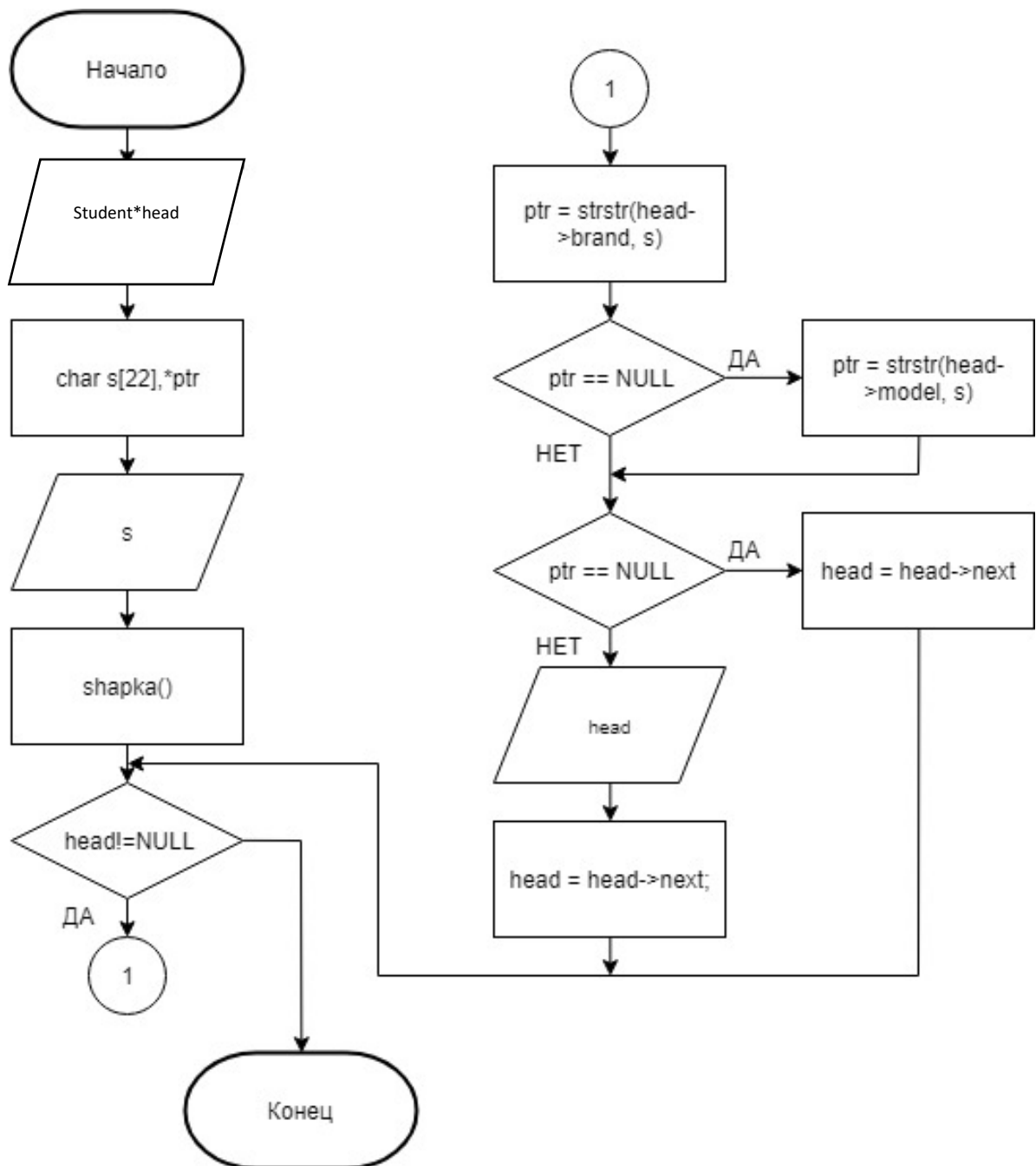


Рисунок 14 - функция поиска

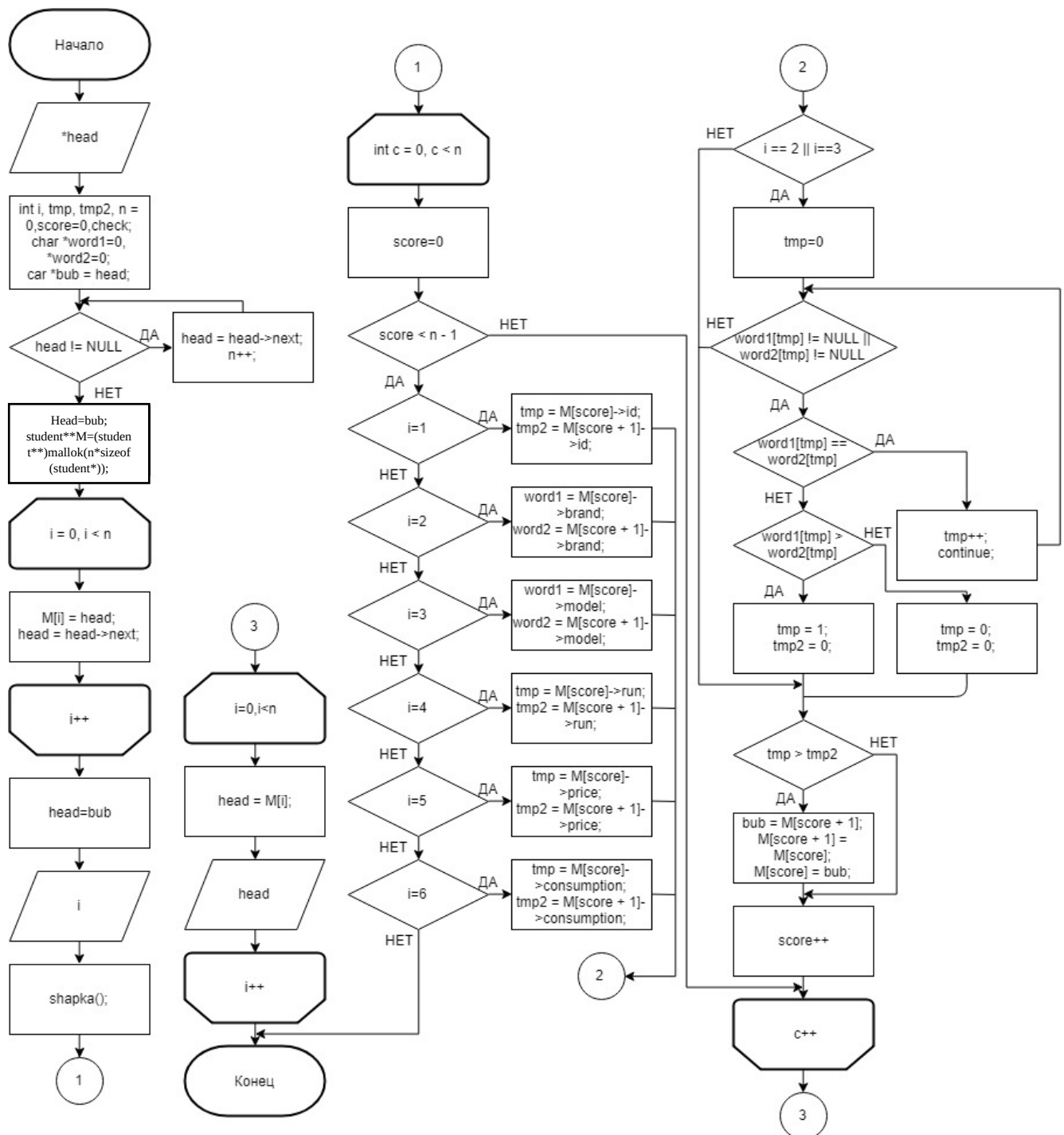


Рисунок 15 - функция сортировки

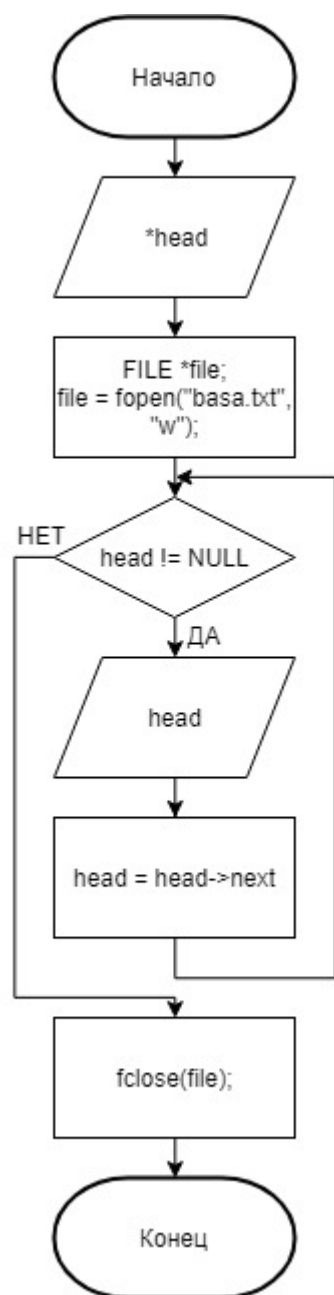


Рисунок 16 - функция загрузки в файл

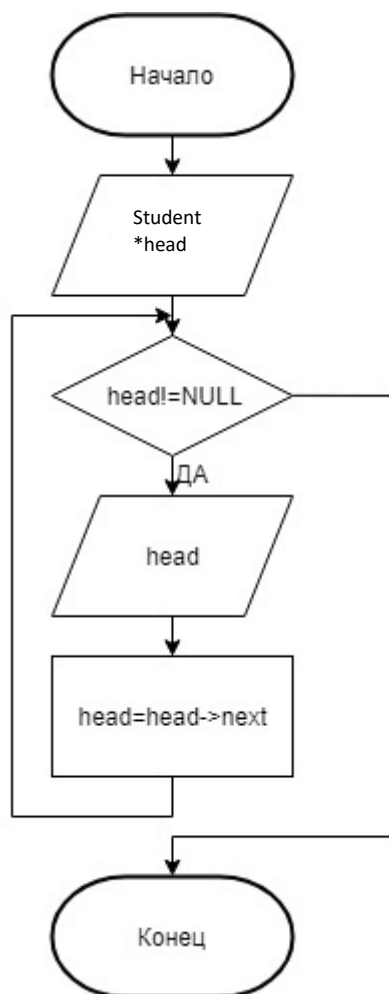
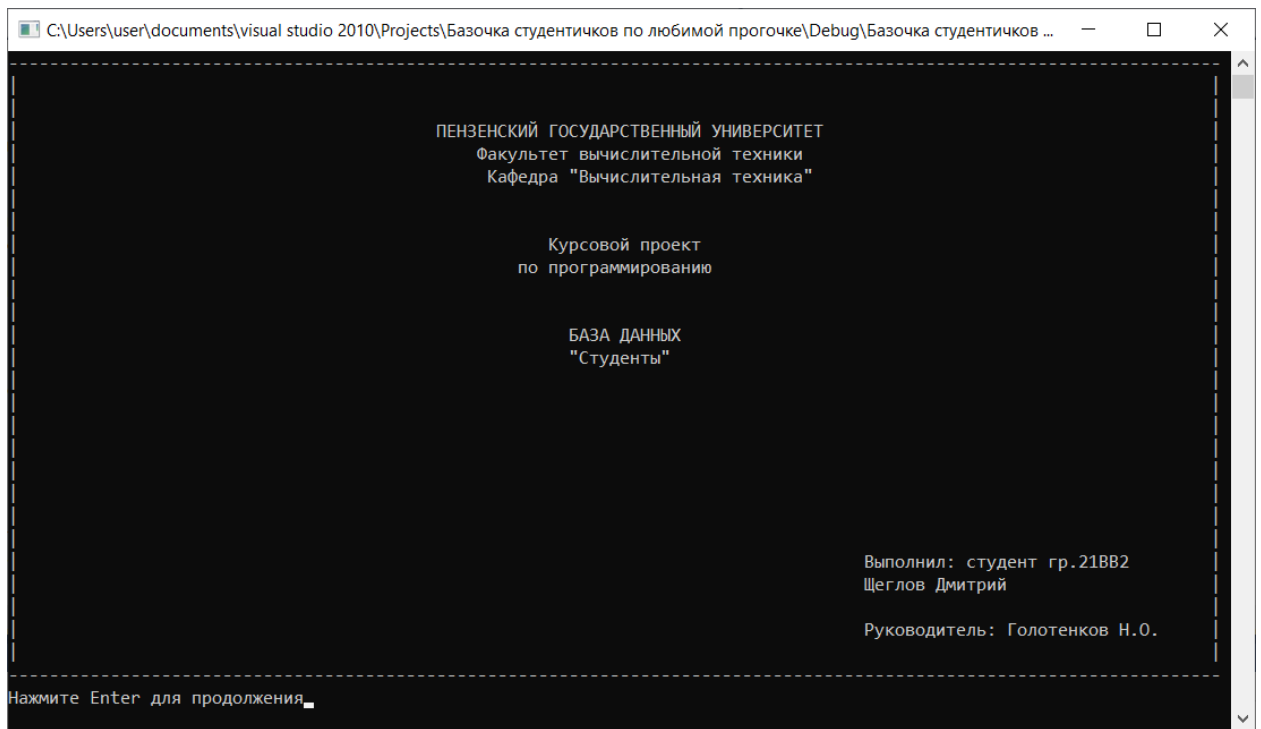


Рисунок 17 - функция вывода базы данных на экран

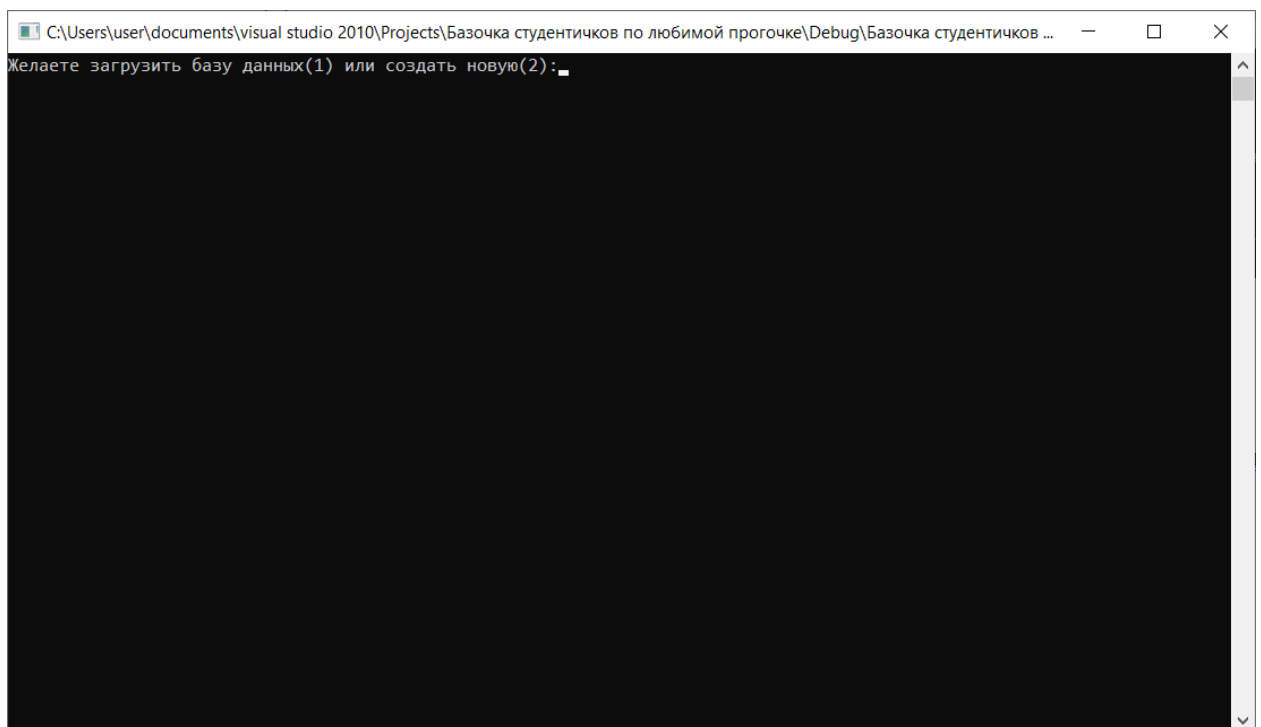
Руководство пользователя

Программа предназначена для хранения информации о студентах. Программа имеет интуитивно понятный интерфейс и поддерживает редактирование записей. При запуске программа выводит на экран название курсовой работы, преподавателя и автора.



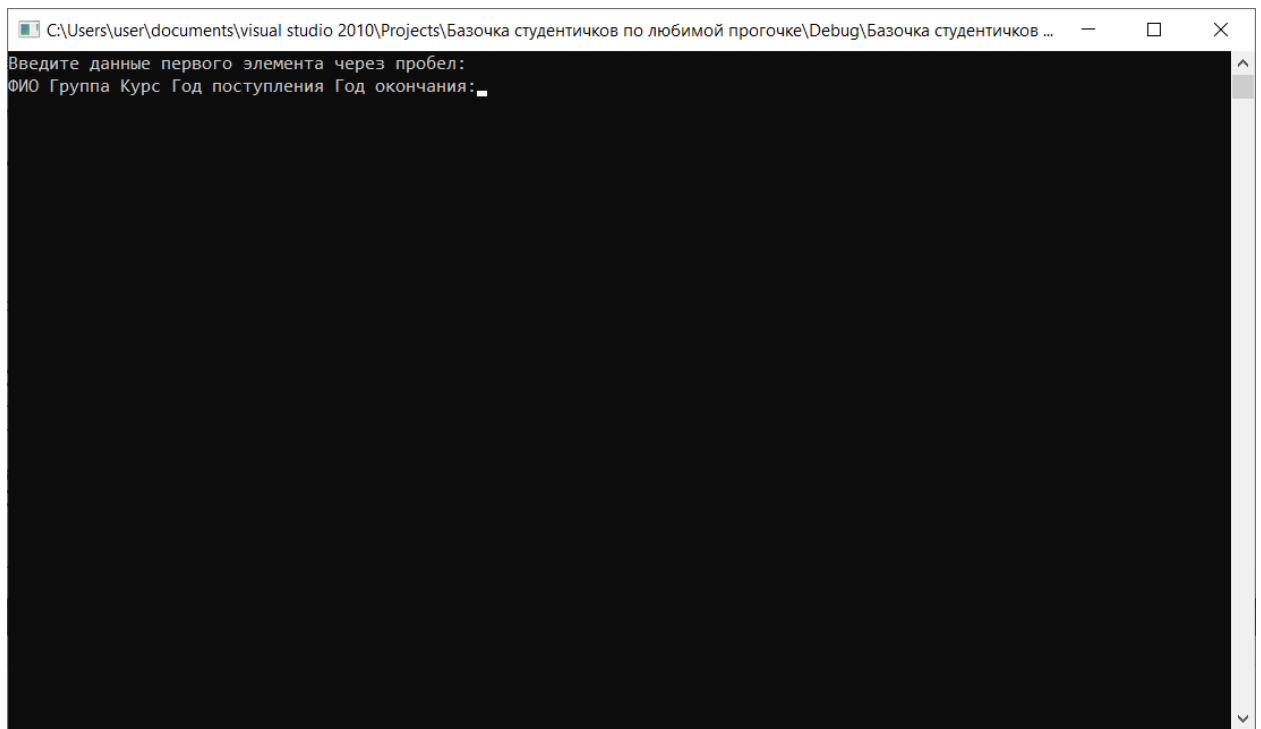
После нажатия клавиши «Enter» появляется вопрос:

«Желаете загрузить базу данных(1) или создать новую(2):»



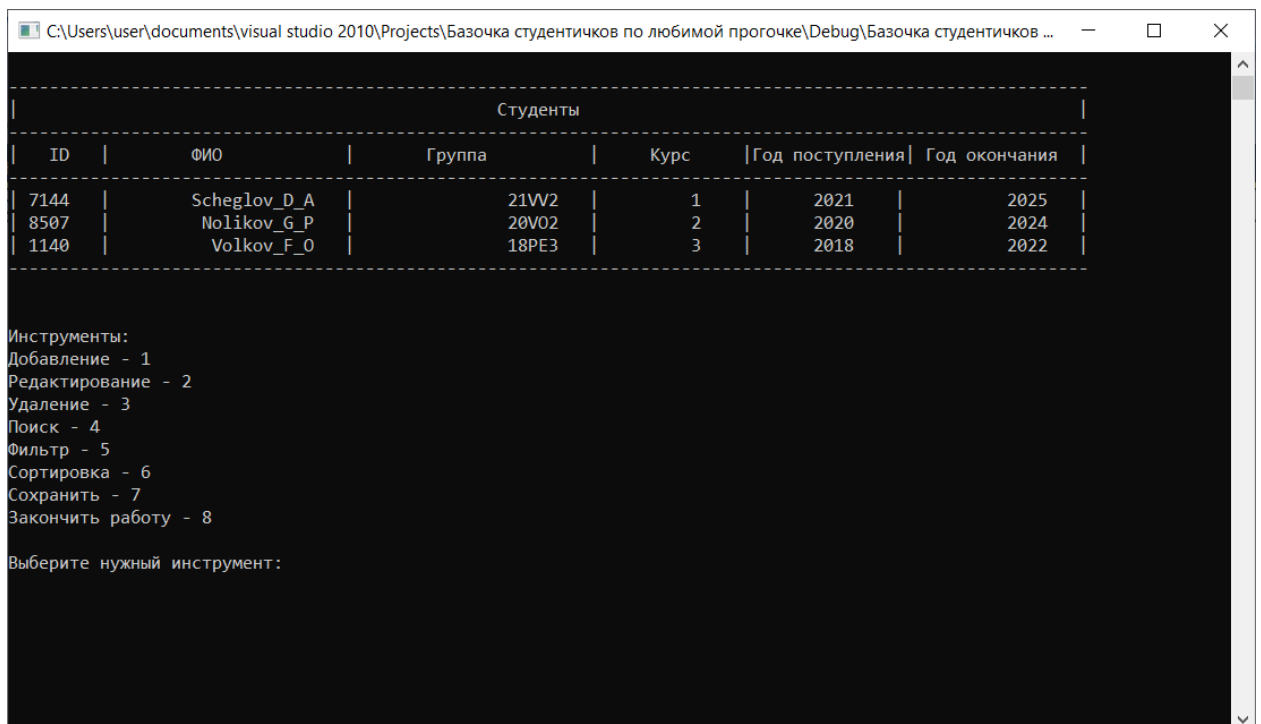
Введите 1, чтобы загрузить базу данных из файла, 2 - чтобы создать новую базу данных.

При выборе пункта 2 появится запрос на ввод данных:



Вводить данные нужно через пробел, затем нажать клавишу Enter.

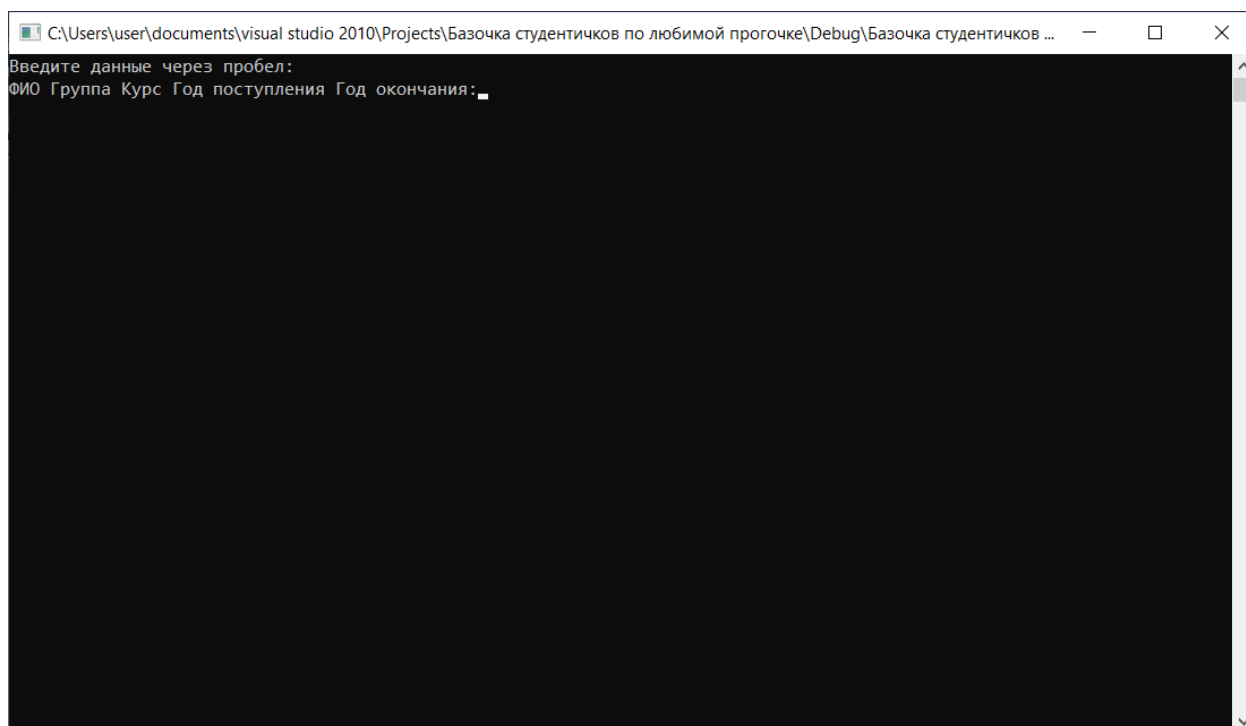
После ввода данных или при выборе пункта 1 программа выведет базу данных и список инструментов на экран:



Чтобы выбрать инструмент, нужно ввести соответствующий номер и нажать Enter.

Добавление

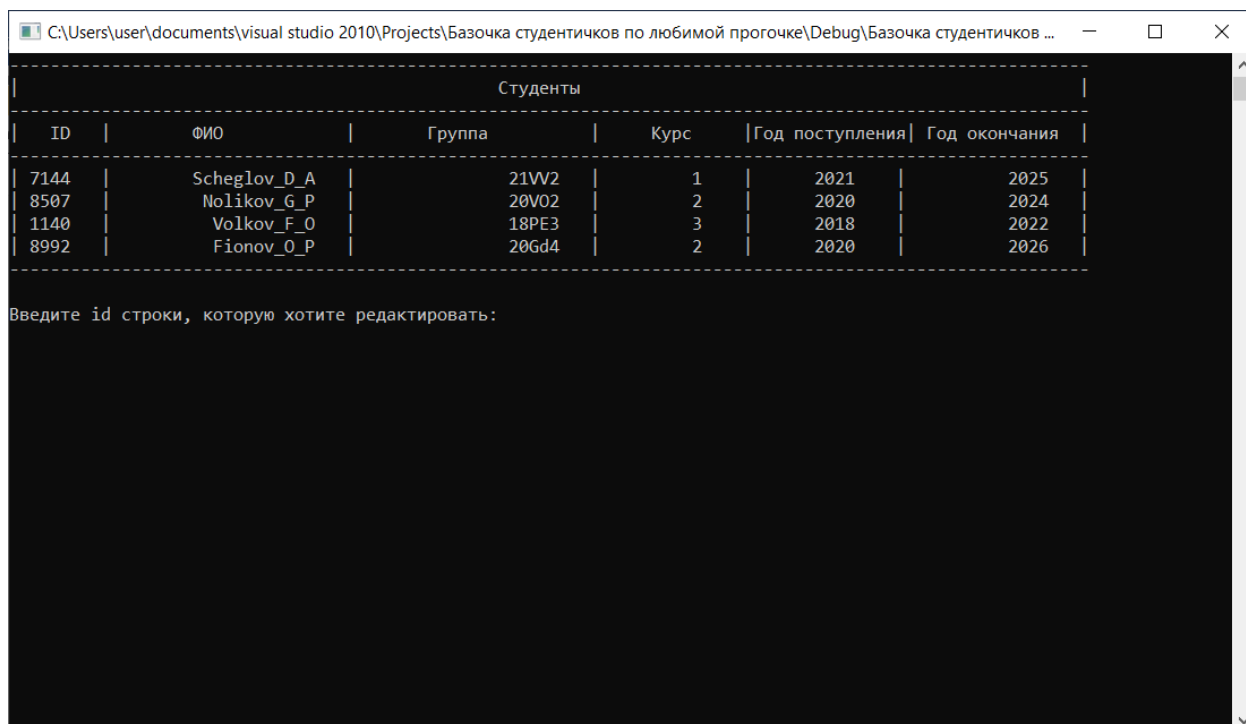
При выборе инструмента «Добавление» появится запрос на ввод данных:



Вводить данные нужно через пробел, затем нажать клавишу Enter.

Редактирование

Сначала нужно ввести id элемента, который вы хотите редактировать:



Затем выбрать данные, которые вы хотите редактировать в этом элементе:

C:\Users\user\documents\visual studio 2010\Projects\Базочка студентиков по любимой прогочке\Debug\Базочка студентиков ...

Студенты						
ID	ФИО	Группа	Курс	Год поступления	Год окончания	
7144	Scheglov_D_A	21VV2	1	2021	2025	
8507	Nolikov_G_P	20V02	2	2020	2024	
1140	Volkov_F_O	18PE3	3	2018	2022	
8992	Fionov_O_P	20Gd4	2	2020	2025	

Введите id строки, которую хотите редактировать:1140

Выберите данные, которые хотите редактировать:
 ФИО-1 Группа-2 Курс-3 Год поступления-4 Год окончания-5:

После этого необходимо ввести новую информацию

C:\Users\user\documents\visual studio 2010\Projects\Базочка студентиков по любимой прогочке\Debug\Базочка студентиков ...

Студенты						
ID	ФИО	Группа	Курс	Год поступления	Год окончания	
7144	Scheglov_D_A	21VV2	1	2021	2025	
8507	Nolikov_G_P	20V02	2	2020	2024	
1140	Volkov_F_O	18PE3	3	2018	2022	
8992	Fionov_O_P	20Gd4	2	2020	2025	

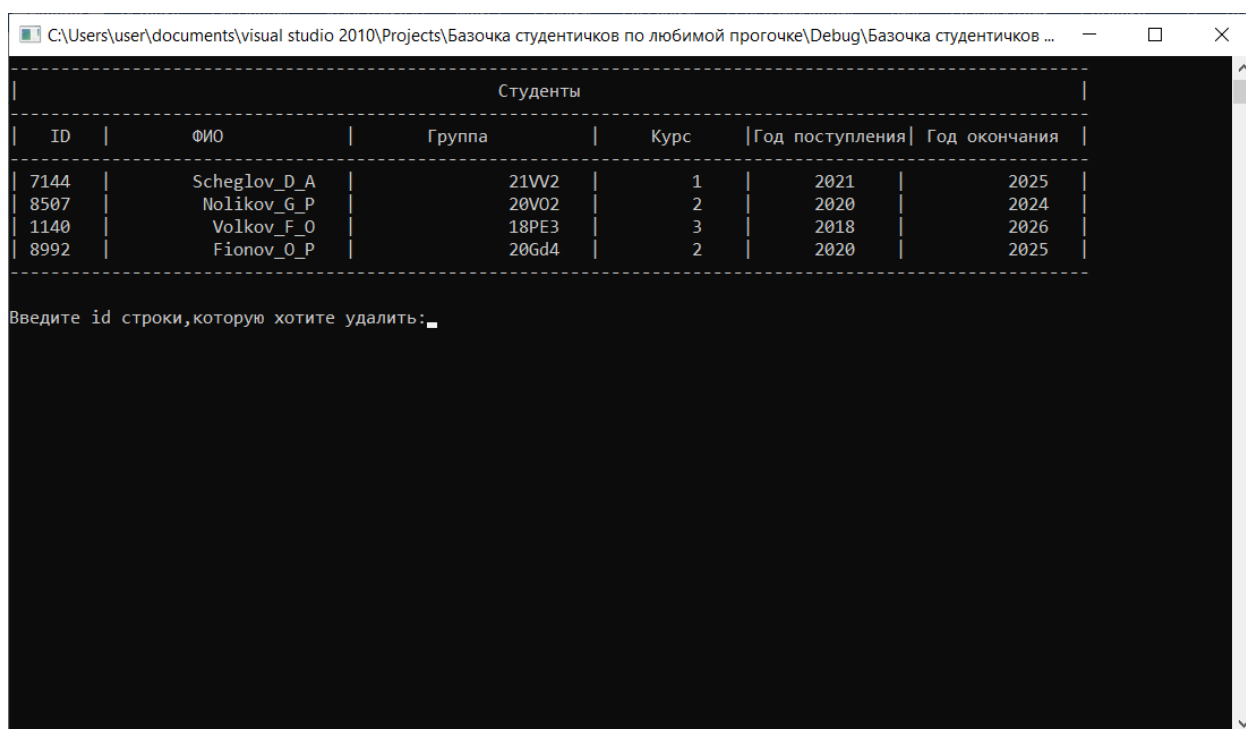
Введите id строки, которую хотите редактировать:1140

Выберите данные, которые хотите редактировать:
 ФИО-1 Группа-2 Курс-3 Год поступления-4 Год окончания-5: 5

Введите данные:

Удаление

Необходимо ввести id строки, которую хотите удалить:



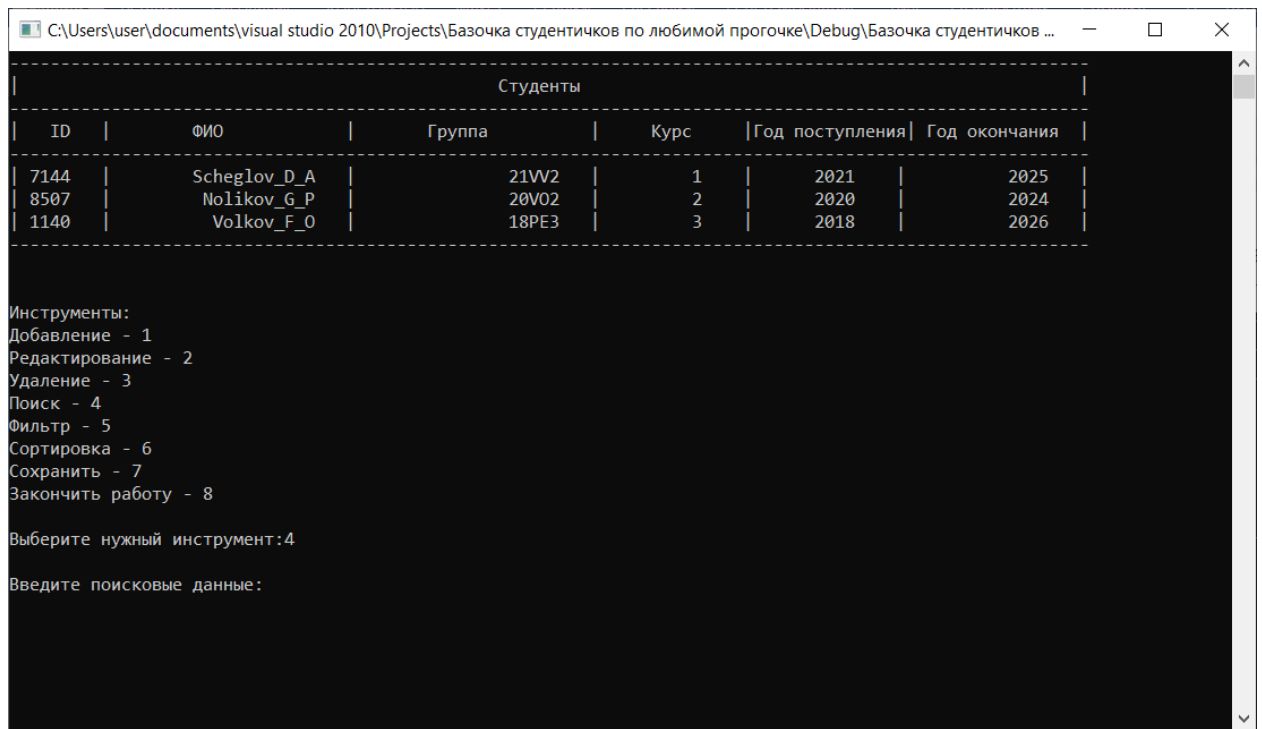
The screenshot shows a Windows command prompt window with the title bar "C:\Users\user\documents\visual studio 2010\Projects\Базочка студентиков по любимой прогочке\Debug\Базочка студентиков ...". The window displays a table of students with the following data:

ID	ФИО	Группа	Курс	Год поступления	Год окончания
7144	Scheglov_D_A	21W2	1	2021	2025
8507	Nolikov_G_P	20V02	2	2020	2024
1140	Volkov_F_O	18PE3	3	2018	2026
8992	Fionov_O_P	20Gd4	2	2020	2025

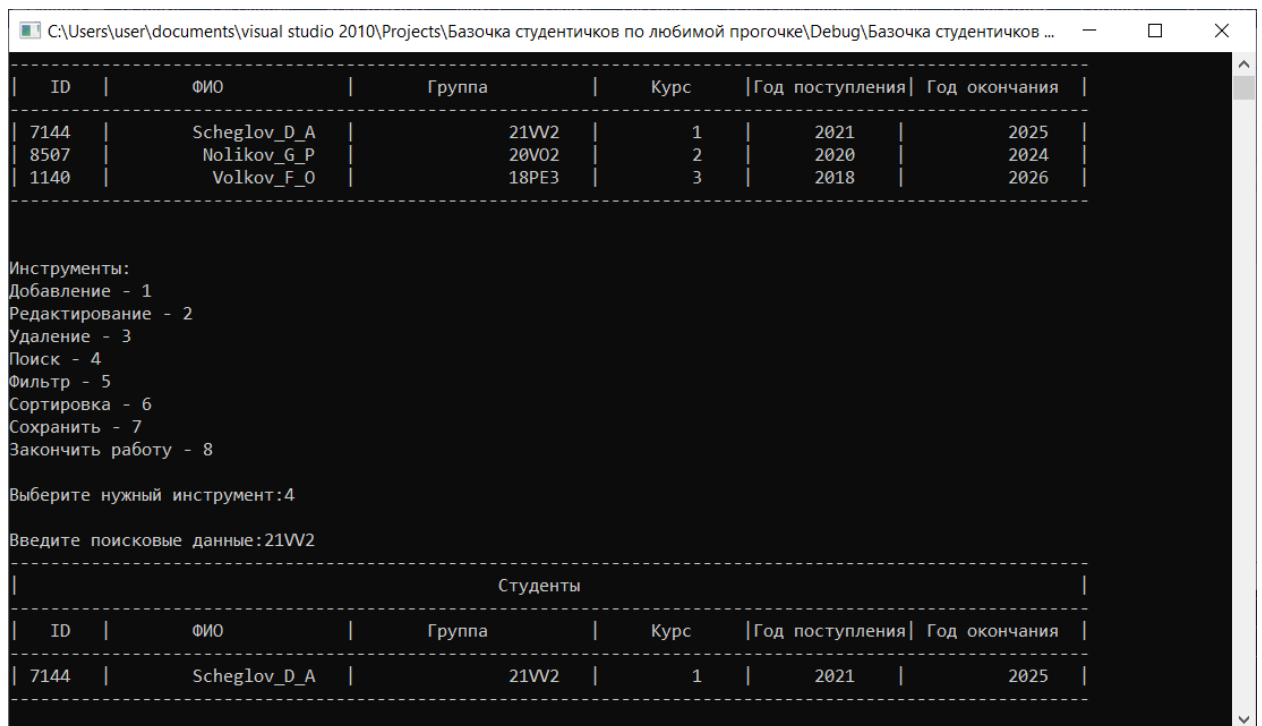
Below the table, the prompt "Введите id строки, которую хотите удалить:_" is displayed, followed by a cursor.

Поиск

Нужно ввести ФИО или группу студента, который вы хотите найти:

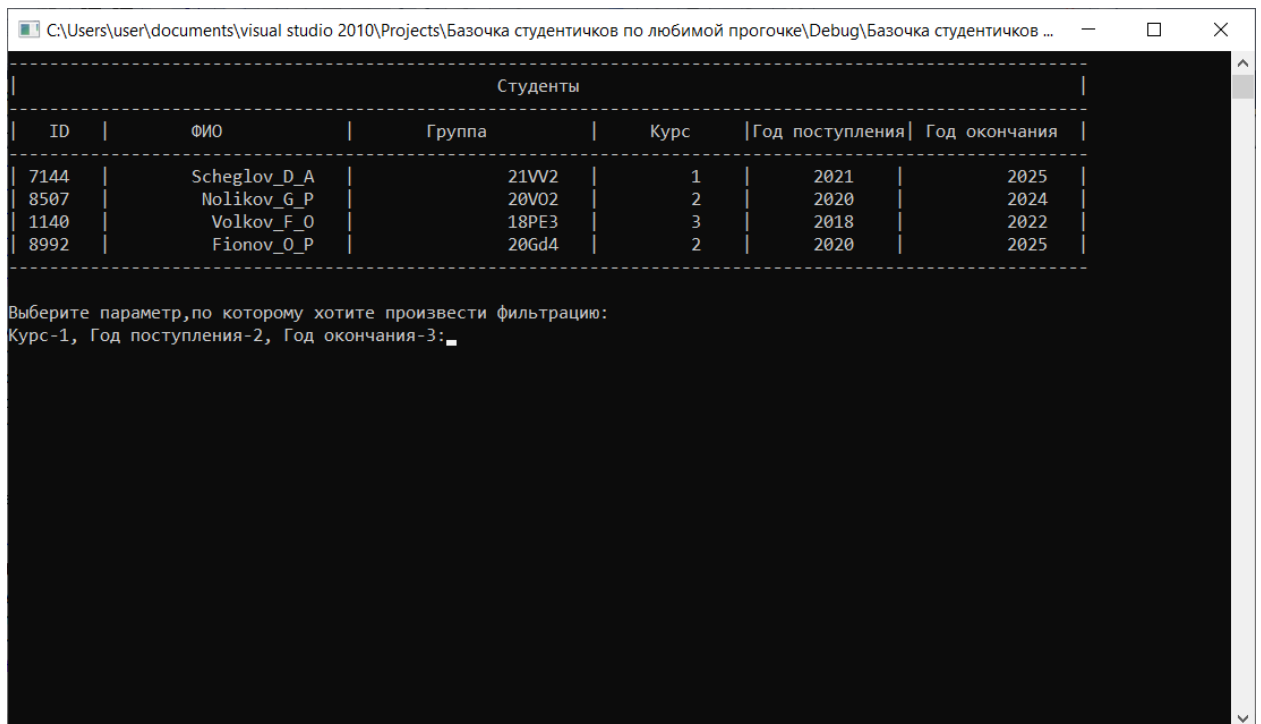


Найденные студенты выведутся на экран:

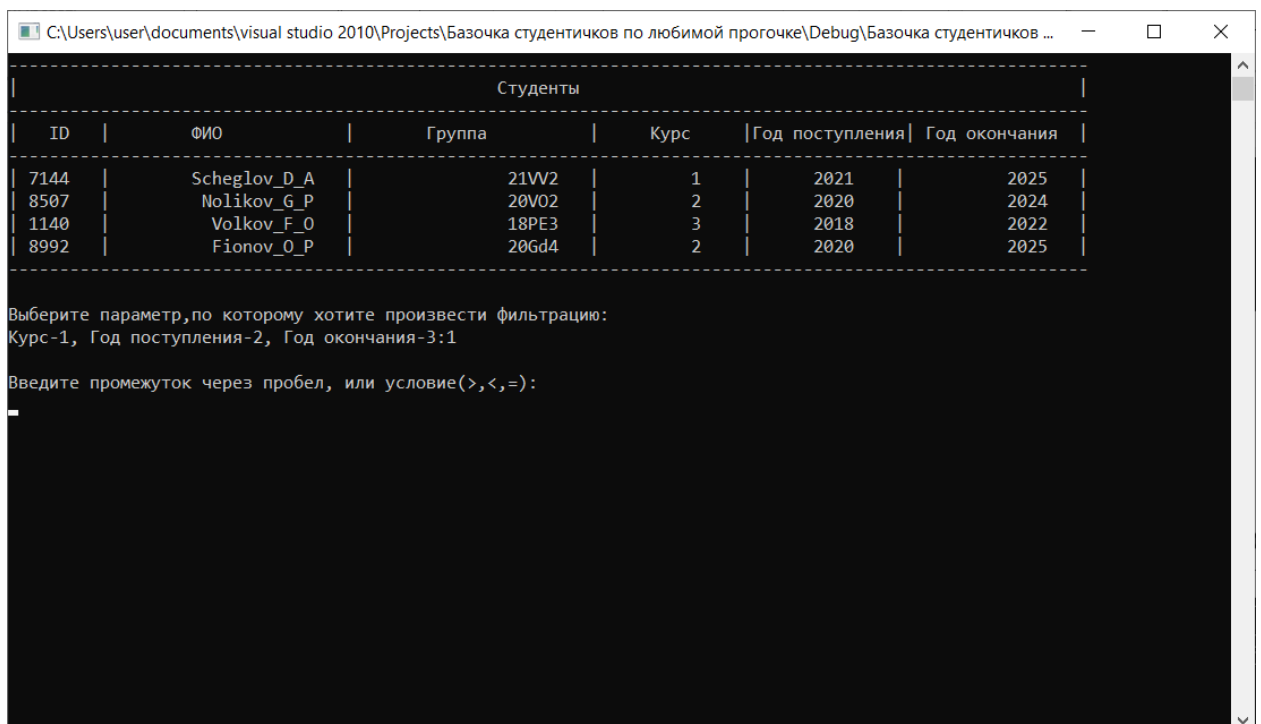


Фильтр

Сначала нужно выбрать параметр, по которому хотите производить фильтрацию:



Затем нужно ввести числовой промежуток ([число] [число]):



Либо математическое условие ($[>, <, =]$ [число]):

ID	ФИО	Группа	Курс	Год поступления	Год окончания
7144	Scheglov_D_A	21VV2	1	2021	2025
8507	Nolikov_G_P	20V02	2	2020	2024
1140	Volkov_F_O	18PE3	3	2018	2022
8992	Fionov_O_P	20Gd4	2	2020	2025

Выберите параметр, по которому хотите произвести фильтрацию:
Курс-1, Год поступления-2, Год окончания-3:1

Введите промежуток через пробел, или условие(>,<,<=):
< 3

После этого отфильтрованные данные будут выведены на экран:

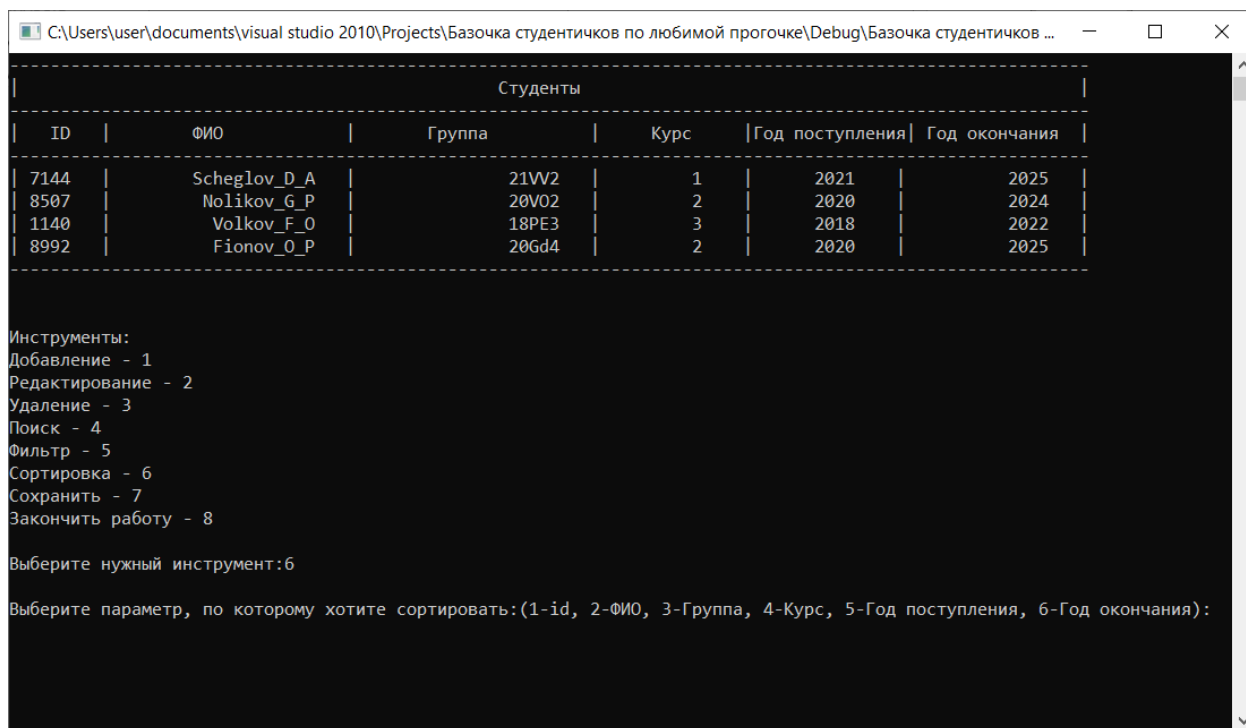
ID	ФИО	Группа	Курс	Год поступления	Год окончания
8507	Nolikov_G_P	20V02	2	2020	2024
8992	Fionov_O_P	20Gd4	2	2020	2025

Выберите параметр, по которому хотите произвести фильтрацию:
Курс-1, Год поступления-2, Год окончания-3:1

Введите промежуток через пробел, или условие(>,<,<=):
< 3

Сортировка

Необходимо ввести цифру, соответствующую параметру, по которому вы хотите провести сортировку:



Затем отсортированные данные будут выведены на экран:

```
C:\Users\user\documents\visual studio 2010\Projects\Базочка студентиков по любимой прогочке\Debug\Базочка студентиков ...
| 8507 | Nolikov_G_P | 20V02 | 2 | 2020 | 2024 |
| 1140 | Volkov_F_O | 18PE3 | 3 | 2018 | 2022 |
| 8992 | Fionov_O_P | 20Gd4 | 2 | 2020 | 2025 |
-----
Инструменты:
Добавление - 1
Редактирование - 2
Удаление - 3
Поиск - 4
Фильтр - 5
Сортировка - 6
Сохранить - 7
Закончить работу - 8
Выберите нужный инструмент:6
Выберите параметр, по которому хотите сортировать:(1-id, 2-ФИО, 3-Группа, 4-Курс, 5-Год поступления, 6-Год окончания):1
-----
Студенты
-----
| ID | ФИО | Группа | Курс | Год поступления | Год окончания |
-----
| 1140 | Volkov_F_O | 18PE3 | 3 | 2018 | 2022 |
| 7144 | Scheglov_D_A | 21VV2 | 1 | 2021 | 2025 |
| 8507 | Nolikov_G_P | 20V02 | 2 | 2020 | 2024 |
| 8992 | Fionov_O_P | 20Gd4 | 2 | 2020 | 2025 |
-----
```

Закончить работу

Введите 1, чтобы сохранить изменения и закрыть программу или 0, чтобы закрыть программу без сохранения:

```
C:\Users\user\documents\visual studio 2010\Projects\Базочка студентиков по любимой прогочке\Debug\Базочка студентиков ...
-----
Студенты
-----
| ID | ФИО | Группа | Курс | Год поступления | Год окончания |
-----
| 7144 | Scheglov_D_A | 21VV2 | 1 | 2021 | 2025 |
| 8507 | Nolikov_G_P | 20V02 | 2 | 2020 | 2024 |
| 1140 | Volkov_F_O | 18PE3 | 3 | 2018 | 2022 |
| 8992 | Fionov_O_P | 20Gd4 | 2 | 2020 | 2025 |
-----
Хотите сохранить изменения(1-да,0-нет):
```

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Получены базовые навыки программирования на языке Си и Ассемблер. Изучены основные возможности среды программирования Visual Studio 2010. Получены навыки отладки и тестирования программ.

В рамках выполнения курсовой работы была написана программа необходимая для ведения базы данных автомобилей. Программа предоставляет хорошие возможности для редактирования данных. Она обеспечивает быструю обработку базу данных.

В дальнейшем программу можно улучшить, добавив полноценный графический интерфейс, улучшить навигацию по базе данных путем добавления постраничного перемещения и сортировки, добавить вывод сообщения об ошибках и подсказок при неверных действиях пользователя, а также оптимизировать код для более быстрого выполнения операций чтения и записи базы данных.

Листинг программы

Файл: «type.h»

```
#pragma once

typedef struct student
{
    char FIO[22], Gruppa[22];
    int Kurs, GodPost, GodOkon, id;
    student *prev, *next;
};
```

Файл: «load.h»

```
#pragma once

student* download(student *head);
void upload(student *head);
```

Файл: «sort_fil.h»

```
#pragma once

void filter(student* head);
void sortirovka(student* head);
```

Файл: «add_del.h»

```
#pragma once
```

```
void add_end(student *head);  
student* deleter(student* head);  
int redact(student *head);
```

Файл: «GIF.h»

```
#pragma once
```

```
void gif(void);  
void shapka(void);  
void zastavka(void);
```

Файл: «headers.h»

```
#pragma once  
#include <iostream>  
#include <stdio.h>  
#include <stdlib.h>  
#include <conio.h>  
#include <malloc.h>  
#include <locale.h>  
#include <stddef.h>  
#include <time.h>
```

Файл: «kursovik.cpp»

```
// kursovik.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается  
// выполнение программы.  
//
```

```
#include "stdafx.h"  
#include "pch.h"  
#include "headers.h"  
#include "type.h"  
#include "load.h"  
#include "add_del.h"  
#include "sort_fil.h"  
#include "gif.h"  
#define _CRT_SECURE_NO_WARNINGS  
#pragma warning (disable:4996)
```

```
FILE *file;
```

```
void clean_stdin(void)  
{  
    int c;  
    do {  
        c = getchar();  
    } while (c != '\n' && c != EOF);  
}
```

```
int check_overflow(student *tmp)  
{  
    if (strlen(tmp->FIO) > 20 || strlen(tmp->Gruppa) > 20)  
    {  
        printf("\nОшибка!");  
        clean_stdin();  
        getch();  
        return 1;  
    }  
    return 0;  
}
```

```

}

int check_scanf(int check, int i)
{
    if (check != i)
    {
        printf("\nОшибка!");
        getch();
        clean_stdin();
        return 1;
    }
    return 0;
}

void vivod(student *head)
{
    shapka();
    while (head != NULL)
    {
        printf("| %04d   |%20s   |%20s   |%10d   |%10d   |   %10d   |\n", head-
>id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
        head = head->next;
    }
    printf("-----\n");
}

int id_unique(student* tmp)
{
    bool i = 0;
    student *save = tmp;
    int id;
    while (i == 0)
    {
        id = rand() % 9999;
        while (tmp!=NULL)
        {
            if (id == tmp->id)
            {
                tmp = save;
                break;
            }
            tmp = tmp->next;
        }
        if (tmp == NULL)
            i = 1;
    }
    return id;
}

student* creation(void)
{
    int check;
    student *tmp = (student*)malloc(sizeof(student));
    tmp->prev = NULL;
    tmp->next = NULL;
    printf("Введите данные первого элемента через пробел:\nФИО Группа Курс Год
поступления Год окончания:");
    check = scanf("%s %s %d %d %d", &tmp->FIO, &tmp->Gruppa, &tmp->Kurs, &tmp-
>GodPost, &tmp->GodOkon);
    if (check_scanf(check,5)==1|| check_overflow(tmp) == 1)
    {
        free(tmp);
        return nullptr;
    }
}

```

```

    }
    tmp->id = rand() % 9999;
    file = fopen("basa.txt", "w");
    fprintf(file, "%d %s %s %d %d %d\n", tmp->id, tmp->FIO, tmp->Gruppa, tmp->Kurs,
tmp->GodPost, tmp->GodOkon);
    fclose(file);
    return tmp;
}

void search(student* head)
{
    char s[22], *ptr;
    printf("\nВведите поисковые данные:");
    scanf("%21s", s);
    if (strlen(s) > 20)
    {
        printf("\nОшибка");
        clean_stdin();
        getch();
        return;
    }
    shapka();
    while (head != NULL)
    {
        ptr = strstr(head->FIO, s);
        if (ptr == NULL)
        {
            ptr = strstr(head->Gruppa, s);
        }
        if (ptr == NULL)
        {
            head = head->next;
            continue;
        }
        else
        {
            printf("| %04d |%20s |%20s |%10d |%10d | %10d |\n",
head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
            head = head->next;
            continue;
        }
    }
    printf("-----\n");
    getch();
}

int main()
{
    setlocale(LC_ALL, "Rus");
    srand(time(NULL));
    student *head = nullptr;
    int i, check;
    zastavka();
    clean_stdin();
m1:    system("cls");
    printf("Желаете загрузить базу данных(1) или создать новую(2):");
    check = scanf("\n%d", &i);
    if (check <= 1)
    {
        goto m1;
    }
    if (i != 1 && i != 2)
    {

```

```

        clean_stdin();
        goto m1;
    }
    gif();
    system("cls");
    if (i == 2)
    {
        head = creation();
        if (head == nullptr)
        {
            goto m1;
        }
        vivod(head);
    }
    else
    {
        head = download(head);
        if (head == nullptr)
        {
            goto m1;
        }
    }
}

while (i != 8)
{
    printf("\n\nИнструменты:\nДобавление - 1\nРедактирование - 2\nУдаление -
3\nПоиск - 4\nФильтр - 5\nСортировка - 6\nСохранить - 7\nЗакончить работу - 8\n");
    m2: printf("\nВыберите нужный инструмент:");
    check=scanf("%d", &i);
    if (check_scanf(check,1) == 1)
    {
        goto m2;
    }
    if (i < 1 || i > 8)
    {
        clean_stdin();
        goto m2;
    }
    switch (i)
    {
    case 1:
    {
        add_end(head);
        break;
    }
    case 2:
    {
        redact(head);
        break;
    }
    case 3:
    {
        head = deleter(head);
        break;
    }
    case 4:
    {
        search(head);
        break;
    }
    case 5:
    {
        filter(head);
        break;
    }

```

```

    }
    case 6:
    {
        sortirovka(head);
        break;
    }
    case 7:
    {
        upload(head);
        break;
    }
    default:
    {
        i = 8;
        break;
    }
}
system("cls");
vivod(head);
clean_stdin();
}
m3: printf("\nХотите сохранить изменения(1-да,0-нет):");
check=scanf("%d", &i);
if (check_scanf(check,1) == 1)
{
    goto m3;
}
if (i != 0 && i != 1)
    goto m3;
if (i==1)
    upload(head);

return 0;
}

```

Файл: «add_del.cpp»

```

#include "stdafx.h"
#include "pch.h"
#include "type.h"
#include "headers.h"
#define _CRT_SECURE_NO_WARNINGS
#pragma warning (disable:4996)

void vivod(student *head);
int check_overflow(student *tmp);
void clean_stdin(void);
int id_unique(student* tmp);
int check_scanf(int check, int i);

void add_end(student *head)
{
    int check,i;
    student *tmp = (student*)malloc(sizeof(student));
    system("cls");
    printf("Введите данные через пробел:\nФИО Группа Курс Год поступления Год\n");
    check = scanf("%21s %21s %21d %21d %d", &tmp->FIO, &tmp->Gruppa, &tmp->Kurs, &tmp->GodPost, &tmp->GodOkon);
    if (check_scanf(check,5)==1||check_overflow(tmp)==1)
    {
        free(tmp);
    }
}

```

```

        return;
    }
    tmp->id = id_unique(head);
    while (head->next != NULL)
        head = head->next;
    tmp->prev = head;
    head->next = tmp;
    tmp->next = NULL;
    system("cls");
}

student* deleter(student* head)
{
    int id,check;
    student *save = head;
    system("cls");
    vivod(head);
    printf("\nВведите id строки, которую хотите удалить:");
    check=scanf("%d", &id);
    if (check_scanf(check,1) == 1)
    {
        return 0;
    }
    while (head!=NULL)
    {
        if (id==head->id)
        {
            if (head->next == NULL)
            {
                head->prev->next = NULL;
                free(head);
                break;
            }
            if (head->prev == NULL)
            {
                head = head->next;
                free(head->prev);
                head->prev = NULL;
                save = head;
                break;
            }
            head->next->prev = head->prev;
            head->prev->next = head->next;
            free(head);
            break;
        }
        else
        {
            head = head->next;
            if (head == NULL)
            {
                printf("\nСтрока не найдена!");
                getch();
                break;
            }
        }
    }
    return save;
}

int redact(student *ptr)
{
    int i,id,check;
    student *head;

```

```

char str[22];
system("cls");
head = ptr;
vivod(head);
printf("\nВведите id строки, которую хотите редактировать:");
check=scanf("%d", &id);
if (check_scanf(check,1) == 1)
{
    return 0;
}
while (head!=NULL)
{
    if (id == head->id)
        break;
    if (head->next == NULL)
    {
        printf("\nСтрока не найдена!");
        getch();
        clean_stdin();
        return 0;
    }
    else
    {
        head = head->next;
    }
}
m2: printf("\nВыберите данные, которые хотите редактировать:\nФИО-1 Группа-2 Курс-3
Год поступления-4 Год окончания-5: ");
check=scanf("%d", &i);
if (check_scanf(check,1) == 1)
{
    return 0;
}
if (i < 1 || i>5)
    goto m2;
printf("\nВведите данные:");
switch (i)
{
case 1:
{
    scanf("%21s", str);
    if (strlen(str) > 20)
    {
        printf("\nОшибка");
        getch();
        return 0;
    }
    for (int n = 0; n < 20; n++)
    {
        if (str != 0)
            head->FIO[n] = str[n];
        else break;
    }
    break;
}
case 2:
{
    scanf("%21s", str);
    if (strlen(str) > 20)
    {
        printf("\nОшибка");
        getch();
        return 0;
    }
}
}

```

```

        for (int n = 0; n < 20; n++)
        {
            if (str != 0)
                head->Gruppa[n] = str[n];
            else break;
        }
        break;
    }
    case 3:
    {
        check=scanf("%d", &head->Kurs);
        if (check_scanf(check, 1) == 1)
            return 0;
        break;
    }
    case 4:
    {
        check=scanf("%d", &head->GodPost);
        if (check_scanf(check, 1) == 1)
            return 0;
        break;
    }
    case 5:
    {
        check=scanf("%d", &head->GodOkon);
        if (check_scanf(check, 1) == 1)
            return 0;
        break;
    }
    default:
    {
        printf("\nЭлемент не найден!");
        getch();
        break;
    }
}
return 0;
}

```

Файл: «load.cpp»

```

#include "stdafx.h"
#include "pch.h"
#include "headers.h"
#include "type.h"
#include "GIF.h"
#define _CRT_SECURE_NO_WARNINGS
#pragma warning (disable:4996)

student* download(student *head)
{
    FILE *file;
    file = fopen("basa.txt", "r");
    if (file == nullptr)
    {
        printf("Файл не существует!\n");
        return nullptr;
    }
    int i = 0;
    printf("\n");
    shapka();
    while (!feof(file))
    {
        student *tmp = (student*)malloc(sizeof(student));
    }
}

```

```

        if (i == 0)
        {
            tmp->prev = NULL;
            tmp->next = NULL;
        }
        else
        {
            tmp->prev = head;
            head->next = tmp;
            tmp->next = NULL;
        }
        head = tmp;
        i++;
        fscanf(file, "%d %s %s %d %d %d\n",&tmp->id, &tmp->FIO, &tmp->Gruppa, &tmp->Kurs, &tmp->GodPost, &tmp->GodOkon);
        printf("| %04d |%20s |%20s |%10d |%10d | %10d | \n",tmp->id, tmp->FIO, tmp->Gruppa, tmp->Kurs, tmp->GodPost, tmp->GodOkon);
    }
    fclose(file);
    printf("-----\n");
    while (head->prev != NULL)
        head = head->prev;
    return head;
}

void upload(student *head)
{
    FILE *file;
    file = fopen("basa.txt", "w");
    while (head != NULL)
    {
        fprintf(file, "%d %s %s %d %d %d\n",head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
        head = head->next;
    }
    fclose(file);
}

```

Файл: «sort_fil.cpp»

```

#include "stdafx.h"
#include "pch.h"
#include "headers.h"
#include "type.h"
#include "GIF.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <malloc.h>
#include <locale.h>
#include <stddef.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning (disable:4996)

void vivod(student *head);
void clean_stdin(void);
int check_scanf(int check, int i);

void filter(student* head)
{

```

```

int i,tmp,n,a,check;
char z[22];
system("cls");
vivod(head);
printf("\nВыберите параметр,по которому хотите произвести фильтрацию:\nКурс-1, Год
поступления-2, Год окончания-3:");
check = scanf("%d", &i);
if (check_scanf(check,1) == 1)
{
    return;
}
if (i < 1 || i>3)
    return;
printf("\nВведите промежуток через пробел, или условие(>,<,:)");
check=scanf("%21s%d", &z, &n);
if (check_scanf(check,2) == 1)
{
    return;
}
if (strlen(z) > 20)
{
    printf("\nОшибка");
    getch();
    return;
}
a = atoi(z);
printf("\n");
if (a > n)
{
    printf("\nНеверно задано условие!");
    getch();
    return;
}
shapka();
while (head != NULL)
{
    switch (i)
    {
        case 1:
        {
            tmp = head->Kurs;
            break;
        }
        case 2:
        {
            tmp = head->GodPost;
            break;
        }
        case 3:
        {
            tmp = head->GodOkon;
            break;
        }
        default:
        {
            printf("\nЭлемент не найден!");
            getch();
            return;
            break;
        }
    }
    if (a != NULL)
    {
        if (a < tmp && tmp < n)

```

```

        {
            printf("| %04d |%20s |%20s |%10d |%10d | %10d\n", head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
            goto m3;
        }
    }
    if (z[0] == '<')
        if (tmp < n)
            printf("| %04d |%20s |%20s |%10d |%10d | %10d\n", head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
        if (tmp > n)
            printf("| %04d |%20s |%20s |%10d |%10d | %10d\n", head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
        if (z[0] == '=')
            if (tmp == n)
                printf("| %04d |%20s |%20s |%10d |%10d | %10d\n", head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
m3:    head = head->next;
    }
    printf("-----\n");
    getch();
}

void sortirovka(student *head)
{
    int i, tmp, tmp2, n = 0, score=0, check;
    char *word1=0, *word2=0;
    student *bub = head, *bub2;
    while (head != NULL)
    {
        head = head->next;
        n++;
    }
    head = bub;
    student **M = (student**)malloc(n * sizeof(student*));
    for (i = 0; i < n; i++)
    {
        M[i] = head;
        head = head->next;
    }
    head = bub;
m1:    printf("\nВыберите параметр, по которому хотите сортировать:(1-id, 2-ФИО, 3-Группа, 4-Курс, 5-Год поступления, 6-Год окончания):");
    check=scanf("%d", &i);
    if (check_scanf(check,1) == 1)
    {
        return;
    }
    if (i < 1 || i>6)
        goto m1;
    shapka();
    for (int c=0; c<n; c++)
    {
        score = 0;
        while (score < n - 1)
        {
            switch (i)
            {
                case 1:
                {
                    tmp = M[score]->id;
                    tmp2 = M[score + 1]->id;
                    break;
                }
            }
        }
    }
}

```

```

}
case 2:
{
    word1 = M[score]->FIO;
    word2 = M[score + 1]->FIO;
    break;
}
case 3:
{
    word1 = M[score]->Gruppa;
    word2 = M[score + 1]->Gruppa;
    break;
}
case 4:
{
    tmp = M[score]->Kurs;
    tmp2 = M[score + 1]->Kurs;
    break;
}
case 5:
{
    tmp = M[score]->GodPost;
    tmp2 = M[score + 1]->GodPost;
    break;
}
case 6:
{
    tmp = M[score]->GodOkon;
    tmp2 = M[score + 1]->GodOkon;
    break;
}
default:
{
    printf("\nОшибка");
    return;
    break;
}
}
}
if (i == 2 || i==3)
{
    tmp = 0;
    while (word1[tmp] != NULL || word2[tmp] != NULL)
    {
        if (word1[tmp] == word2[tmp])
        {
            tmp++;
            continue;
        }
        else
        {
            if (word1[tmp] > word2[tmp])
            {
                tmp = 1;
                tmp2 = 0;
            }
            else
            {
                tmp = 0;
                tmp2 = 0;
            }
            break;
        }
    }
}
}

```

```

        if (tmp > tmp2)
        {
            bub2 = M[score + 1];
            M[score + 1] = M[score];
            M[score] = bub2;
        }
        score++;
    }
}
for (i = 0; i < n; i++)
{
    head = M[i];
    printf("| %04d   |%20s   |%20s   |%10d   |%10d   |   %10d   |\n", head->id, head->FIO, head->Gruppa, head->Kurs, head->GodPost, head->GodOkon);
}
printf("-----\n");
getch();
}

```

Файл: «GIF.cpp»

```

#include "stdafx.h"
#include "pch.h"
#include "headers.h"
//#include <iostream>
//#include <conio.h>

void gif(void)
{
    int m = 999999, src = 0;
    const char *student1 = " ( ) \n", *student2 = " /|/   \n", *sp1 = " ";
    const char *student3 = " | \n", *student4 = " ,|,\n", *sp2 = "_";
    while (src < 100)
    {
        system("cls");
        __asm
        {
            push eax
            push ebx
            push edx

            mov eax,src
            xor ebx,ebx
m1:      cmp eax,ebx
            jle m2
            push eax
            push sp1
            call dword ptr printf
            add esp,4
            pop eax
            inc ebx
            jmp m1

m2:      push student1
            call dword ptr printf
            add esp,4

            xor ebx, ebx
            mov eax, src
m3:      cmp eax,ebx
            jle m4
            push eax
            push sp1
            call dword ptr printf

```

```

        add esp,4
        pop eax
        inc ebx
        jmp m3

m4:    push student2
        call dword ptr printf
        add esp,4

        xor ebx, ebx
        mov eax, src
m5:    cmp eax,ebx
        jle m6
        push eax
        push sp1
        call dword ptr printf
        add esp,4
        pop eax
        inc ebx
        jmp m5

m6:    push student3
        call dword ptr printf
        add esp,4

        xor ebx, ebx
        mov eax, src
m7:    cmp eax, ebx
        jle m8
        push eax
        push sp2
        call dword ptr printf
        add esp,4
        pop eax
        inc ebx
        jmp m7

m8:    push student4
        call dword ptr printf
        add esp,4

        pop edx
        pop ebx
        pop eax
    }
    for (int n = 0; n < m; n++)
    {
    }
    src += 1;
}

void shapka(void)
{
    printf("-----\n");
    printf("|                                     Студенты\n");
    printf("-----\n");
    printf("|      ID      |      ФИО      |      Группа      |      Курс\n");
    printf("| Год поступления | Год окончания | \n");
    printf("-----\n");
}

```

[illegible]

