**ЛАБОРАТОРНАЯ РАБОТА №3**

**«ПРОГРАММИРОВАНИЕ ОПЕРАЦИЙ НАД СТРУКТУРАМИ**

**И БИНАРНЫМИ ФАЙЛАМИ. СПИСКИ»**

**Цель работы**: Исследование особенностей обработки бинарных файлов, хранящих структурные типы данных, организация списка, обработка данных в списках.

1. **Вариант задания**

Описать структуру с именем STUDENT, содержащую следующие поля:

1. порядковый номер;
2. номер;
3. фамилия и имя;
4. год рождения;
5. год поступления в университет;
6. структура OCENKI, содержащая четыре поля: физика, математика, программирование, история.

Организовать двунаправленный список.

Написать программ, выполняющую следующие действия с помощью функций:

1. ввод с клавиатуры данных в список.
2. Чтение данных из файла и занесение их в список;
3. Вывод данных на экран;
4. корректировку данных в списке по номеру записи;
5. добавление записи в список в необходимое положение(сохранение сортировки).
6. удаление второй и третьей записи;
7. удаление любой записи по ее ид.
8. Сохранение списка в файл.
9. Перенос последнего в начало и обратно.

**2. Алгоритм программы**



Рисунок 1 – Основное тело программы



Рисунок 2 - Алгоритм добавления данных в список



Рисунок 3 – Вывод записей на экран



Рисунок 4 – Считывание записей с файла



Рисунок 5 ­­­­­­­­­­­­­– Поиск и редактирование записи



Рисунок 6 – Удаление второго-третьего элемента

Рисунок 7 – Удаление элемента по id



Рисунок 8 – Добавление элемента в нужную позицию



Рисунок 9 – Обмен информации между первым и последним

**3. Код программы на языке C**

#pragma warning(disable : 4996) ;

#pragma warning(disable : 6031) ;

#include <stdio.h>

#include <string.h>

#include <conio.h>

#include <windows.h>

#include <stdlib.h>

#include "C:\Users\Виктор\source\repos\VicMenuDLL\VicMenuDLL\VicMenuDLL.cpp"; //файл библиотеки меню

#define clear() system("cls")

//добавить новый так, чтобы сохранилась упорядоченность

typedef struct {

int id;

char name[40];

char surname[40];

int birth\_date;

int stud\_date;

struct {

int fiz;

int math;

int it;

int history;

} marks;

} STudent;

typedef struct Persons {

STudent info;

struct Persons\* next;

struct Persons\* prev;

} Person;

const char\* Menu[] = { "Добавить новый элемент",

"Загрузить из файла",

"Записать все в файл ",

"Вывод данных в списке",

"Редактировать запись",

"Добавить элемент с сохранением поледовательности",

"Удалить элемент(любой)",

"Удалить второй и третий",

"Перенести последний в начало",

"Выход" };

const int SizeStudent = sizeof(STudent);

const int MenuSize = 10;

//------------------------------------------------------------------- Область функций ----------------------------------------------------------------

//-----------------------------------функции создания списка----------------------------------------------

void addToList(Person\*\* head, const STudent\* info);//функция добавления в начало списка

int аddNewElement(Person\*\* st);// Функция добавления элемента в конец

STudent GetInfoFromCeyboard(STudent d); //форма считывания с клавиатуры данных

Person \* addNewElemetnSort(Person\* st); //Добавить элемент в нужное место

//----------------------------------Функции работы с файлами----------------------------------------------

int create\_file(FILE\* f, Person\* St); //Функция записи спсика в файл

Person\* loadFromFile\_new(FILE\* f); //Функция загрузки данных из файла

//------------------------------Функции работы с элементами списка----------------------------------------

int correctInfo(FILE\* f); //Редактирование записи по ид

void swap(Person\* st); // перенос ппоследнего в начало

int deleteTwoThree(Person\* st);//удалить второй и третий

Person\* searchElementById(Person\*, int); //Функция поиска элемента по id , возвращает ссылку (положение)

void deleteFromList(Person\*\*, int); //Функция удаления записи из листа. Удалить можно любую запись

//--------------------------------Функции вывода списка на экран------------------------------------------

void printTable(); //Напечатать заголовок таблицы

void printBorder(); //Вертекальный разделитель таблицы

int print\_file(Person\* st); //вывод списка на экран

void printINFO(STudent d, int index);//Вывод одной записи на экран

//-------------------------------Вспомогательные функции -------------------------------------------------

void MenuSelect(int selector, FILE\* f); //Основная функция, которая отвечает за запуск функций в зависимости от полученного значения селектора

int getResponse();//Получить подтверждение от пользователя

//------------------------------------------------------------------- Конец области функций ---------------------------------------------------------

Person\* st = NULL;

int position[] = { 1,1 };

int main(void) {

SetConsoleCP(1251); // Задаем таблицу символов для консоли.

SetConsoleOutputCP(1251);

FILE\* f = fopen("data.dat", "rb+");//Открытие существующего файла для чтения и записи в конец

if (!f) {

f = fopen("data.dat", "wb+"); //Создание нового файла для обновления

if (!f) {

puts("Не могу открыть (создать) файл\n");

return 1;

}

}

// int (\* Create)(Person\*) = аddNewElement; пример присваивания какой-либо переменной ссылки на функцию.

while (1) {//вывод меню и запуск соответствующих функций

clear();

MenuSelect(PrintMenu(Menu, position, MenuSize, 2), f);

}

}

void MenuSelect(int selector, FILE\* f)

{

switch (selector) {

case 1:

аddNewElement(&st);

break;

case 2:

if (getResponse()) {

st = loadFromFile\_new(f);

}

break;

case 3:

create\_file(f, st);

break;

case 4:

print\_file(st);

break;

case 5:

correctInfo(st);

break;

case 6:

st = addNewElemetnSort(st);

break;

case 7:

printf("\nВведите ИД записи -->"); int l; scanf("%d", &l);

deleteFromList(&st, l);

puts("Запись удалена ... Нажмите любую кнопку..."); getch();

clear();

break;

case 8:

if (getResponse()) {

if (deleteTwoThree(st)) printf("\n Записи успешно удалены ... Нажмите любую кнопку"); else printf("\n В списке недостаточно записей ... Нажмите любую кнопку");

getch();

}

break;

case 9:

if (getResponse()) {

swap(st);

printf("\n Записи поменяны местами ... Нажмите любую кнопку"); getch();

}

break;

case 10:

if(getResponse())

exit(666);

break;

}

}

void printBorder() {

for (int i = 0; i < 180; i++)

printf("-");

}

void printTable()

{

char \* TableColsName[] = {" № "," id "," Фамилия "," Имя "," Год рождения "," Год поступления "," Физика "," Математика "," Программирование "," История "};

printBorder(); printf("\n"); char del = '|';

printf("%c %s %c %s %-13c %-8s %12c %18s %13c %s %c %s %c %s %c %s %c %s %c %s %c\n",del, TableColsName[0], del, TableColsName[1], del, TableColsName[2], del, TableColsName[3], del, TableColsName[4], del, TableColsName[5], del, TableColsName[6], del, TableColsName[7], del, TableColsName[8], del, TableColsName[9], del);

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

}

void printINFO(STudent d,int index)

{

char del = '|';

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c %-17d %c %-8d %c %-12d %c %-18d %c %-9d %c",del,index,del,d.id,del,d.surname,del,d.name,del,d.birth\_date,del,d.stud\_date,del,d.marks.fiz,del, d.marks.math, del, d.marks.it, del, d.marks.history, del);

printf("\n");

printBorder();

printf("\n");

}

int create\_file(FILE\* f, Person\* St)

{

if (!St) return 666; fclose(f); f = fopen("data.dat", "wb+");

fseek(f, 0, SEEK\_SET);

while (St->prev) St = St->prev;

for (Person\* Temp = St; Temp; Temp = Temp->next)

{

STudent te = Temp->info;

fwrite(&te, SizeStudent, 1, f);

}

clear();

puts("Данные сохранены. Нажмите любую кнопку ....");

getch();

return 0;

}

int print\_file(Person\* st) {

clear(); int Posid = 1;

printTable();

while (st != NULL) {

printINFO(st->info,Posid);

st = st->next;

Posid++;

}

puts("Нажмите любую клавишу");

getch();

return 0;

}

Person\* searchElementById(Person\* head, int id)

{

Person\* elem = NULL;

if (head)

{

for (Person\* tmp = head; tmp; tmp = tmp->next)

if (tmp->info.id == id)

{

elem = tmp;

break;

}

}

return elem;

}

void deleteFromList(Person\*\* head, int id)

{

Person\* elemToDel = NULL;

if (!(\*head)) return;

elemToDel = searchElementById(\*head, id);

if (elemToDel)

{

if (elemToDel->next) elemToDel->next->prev = elemToDel->prev;

if (elemToDel->prev) elemToDel->prev->next = elemToDel->next;

if (\*head == elemToDel) \*head = elemToDel->next;

free(elemToDel);

}

}

int deleteTwoThree(Person\* st) {

if (!(st)) return 0; Person\* temp; //если список пуст - выйти

while (st->prev) st = st->prev; //возвращаем указатель в начало

if (!(st->next)) return 0; //если элемент только один - выходим

else st = st->next; //иначе переходим на второй элемент

if (!(st->next)) return 0;// если элементов всего два - выходим

else st = st->next;// иначе переходим на третий

if (!(st->next)) temp = NULL; else //если третий последний - то запоминаем пустоту

temp = st->next; //запоминаем указатель на четвертый элемент

st = st->prev; //возвращаемся на второй

free(st->next);//удаляем третий

st = st->prev; // возвращаемся на первый

free(st->next);//удаляем второй

if(temp) temp->prev = st; //у четвертого предыдущим ставим первый , если четвертый существует

st->next = temp;//У первого следующим ставим четертый

return 1;

}

void addToList(Person\*\* head, const STudent\* info)

{

Person\* pointer = NULL;

if (!(\*head)) //если списка не существует - создать его

{

\*head = (Person\*)calloc(sizeof(Person), 1); //освобождаем память

pointer = \*head; //запоминаем

}

else

{

for (pointer = \*head; pointer->next; pointer = pointer->next); //идем до последнего

pointer = ((pointer->next = (Person\*)calloc(sizeof(Person), 1))->prev = pointer)->next; //создаем зависимость ссылок для элемента, сначала выделяем память и запоминаем, потом новому элементу прошлое задаем как текущее положение и потом переходим на этот поинтер.

}

pointer->info = \*info; //записываем данные

}

int аddNewElement(Person\*\* st)

{

STudent d; d.id = NULL;

char\*\* Menu[] = { "Добавить запись","Выход" };

int position[] = { 1,1 }; int flag = 0;

while (1) {

clear();

if (flag)

{

printTable();

printINFO(d,1);

}

switch (PrintMenu(Menu, position, 2, 1)) {

case 1:

{

d = GetInfoFromCeyboard(d);

addToList(st, &d); flag = 1;

}

break;

case 2:

return 1;

}

}

}

Person \* addNewElemetnSort(Person\* st)

{

Person\* temp;

STudent d; d.id = NULL; d = GetInfoFromCeyboard(d);

temp = (Person\*)calloc(sizeof(Person), 1); temp->info = d;

if (!(st)) { //если список пуст

addToList(&st, &d); return st;//добавляем в начало и не морочим голову если список пуст

}

else

{ //иначе

while (st->prev) st = st->prev; //идем в начало списка

while (1)

{

if (st->info.id >= d.id) //остановились на элементе который больше

{

temp->next = st; //темпу записываем адрес этого элемента

temp->prev = st->prev; //предыдущий темпу - предыдущий текущего положения

if (st->prev) st->prev->next = temp; else temp->next->prev = temp; //предыдущему элементу записываем следующий как темп

st = temp;

while (st->prev) st = st->prev; //Возврат в начало

return st;

}

else if ((st->next)) st = st->next; else { addToList(&st, &d); while (st->prev) st = st->prev; return st; //возврат в начало /добавляем в начало и не морочим голову

}

}

}

}

Person\* loadFromFile\_new(FILE\* f)

{

STudent tmp;

Person\* head = NULL;

int count = 1;

fseek(f, 0, SEEK\_SET);

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

while (fread(&tmp, sizeof(STudent), 1, f))

{

addToList(&head, &tmp);

printINFO(tmp,count);

count++;

}

puts("Данные считаны. Нажмите любую кнопку ");

getch();

return head;

}

int correctInfo(Person\* st)

{

char\*\* Menu1[] = { "Найти по номеру","Выйти в главное меню", "Редактировать","Сохранить" }; //меню если элемент для редактирования активен

char\*\* Menu2[] = { "Найти по номеру","Выйти в главное меню" }; //меню если элемента нет

int position[] = { 1,1 }; //массив позиции для меню

STudent d,old; d.id = NULL;

while (1)

{

int select; int Posid = 1; clear();

if (d.id != NULL) //если запись с которо работаем - есть

{

printTable(); //заголовок таблицы

printINFO(d,1); // вывод информации

getch(); //ожидание нажатия кнопки для вывода меню

select = PrintMenu(Menu1, position, 4, 1); // вызов меню с чермя параметрами

} else select = PrintMenu(Menu2, position, 2, 1); // если записи нет - меню с двумя параметрами

switch (select)

{

case 3:

{

d = GetInfoFromCeyboard(d); // получение данных с клавиатуры

break;

}

case 4:

{

clear();

printf("\n--------------------------------------------------------------------------------- Новая запись ------------------------------------------------------------------------------------- \n");

printTable(); //заголовок таблицы

printINFO(d, 1); // вывод информации

printf("\n-------------------------------------------------------------------------------- Старая запись ------------------------------------------------------------------------------------- \n");

printTable(); //заголовок таблицы

printINFO(old, 1); // вывод информации

printf("\n");

if (getResponse()) {

st->info = d;

puts("Данные сохранены . Нажмите любую кнопку");

getch();

}

}

break;

case 1: {

int SerchID; printf("\nВведите идентификационный номер -->"); scanf("%d", &SerchID);

int flag = 0; if (!st) break;

while (st->prev) st = st->prev;

while (1)

{

d = st->info;

if (d.id == SerchID)

{

flag = 1;

old = d;

break;

}

if (st->next == NULL) break;

st = st->next;

Posid++;

}

if (!flag) {

printf("Не найден студент с таким идентификатором .... Нажмите любую кнопку"); getch();

d.id = NULL;

}

break; }

case 2: return 0;

}

}

}

int getResponse() {

printf("\n Вы уверены что хотите выполнить данную команду ? [Y | Any] "); char c = getch();

if ((c == 'y') || (c == 'Y')) return 1; else return 0;

}

void swap(Person\* st)

{

if (!st) return;

while (st->prev) st = st->prev;

if (!(st->next)) return;

STudent temp; Person\* TMP;

while (st->next) st = st->next;

TMP = st;

temp = st->info;

while (st->prev) st = st->prev;

TMP->info = st->info;

st->info = temp;

}

STudent GetInfoFromCeyboard(STudent d)

{

printf("\n");

printTable();

printf("| | "); scanf("%d", &d.id); clear();

printTable(); printf("| | %-6d | ", d.id);

scanf("%s", &d.surname); clear(); char del = '|'; int l = 1;

printTable(); printf("%c %-3d %c %-6d %c %-32s %c", del, l, del, d.id, del, d.surname, del);

scanf("%s", &d.name);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c", del, l, del, d.id, del, d.surname, del, d.name, del);

scanf("%d", &d.birth\_date);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c", del, l, del, d.id, del, d.surname, del, d.name, del, d.birth\_date, del);

scanf("%d", &d.stud\_date);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c %-17d %c", del, l, del, d.id, del, d.surname, del, d.name, del, d.birth\_date, del, d.stud\_date, del);

scanf("%d", &d.marks.fiz);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c %-17d %c %-8d %c", del, l, del, d.id, del, d.surname, del, d.name, del, d.birth\_date, del, d.stud\_date, del, d.marks.fiz, del);

scanf("%d", &d.marks.math);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c %-17d %c %-8d %c %-12d %c", del, l, del, d.id, del, d.surname, del, d.name, del, d.birth\_date, del, d.stud\_date, del, d.marks.fiz, del, d.marks.math, del);

scanf("%d", &d.marks.it);

clear(); printTable();

printf("%c %-3d %c %-6d %c %-32s %c %-30s %c %-14d %c %-17d %c %-8d %c %-12d %c %-18d %c", del, l, del, d.id, del, d.surname, del, d.name, del, d.birth\_date, del, d.stud\_date, del, d.marks.fiz, del, d.marks.math, del, d.marks.it, del);

scanf("%d", &d.marks.history);

return d;

}

**4. Результаты тестирования и отладки программы**

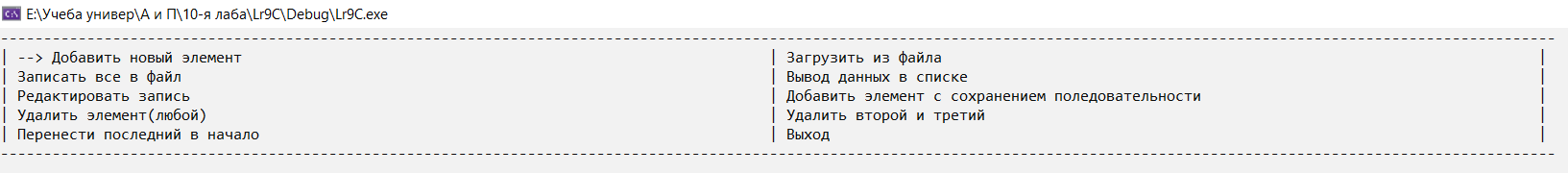


Рисунок 10 – Меню выбора действий



Рисунок 11 – Меню добавления записи

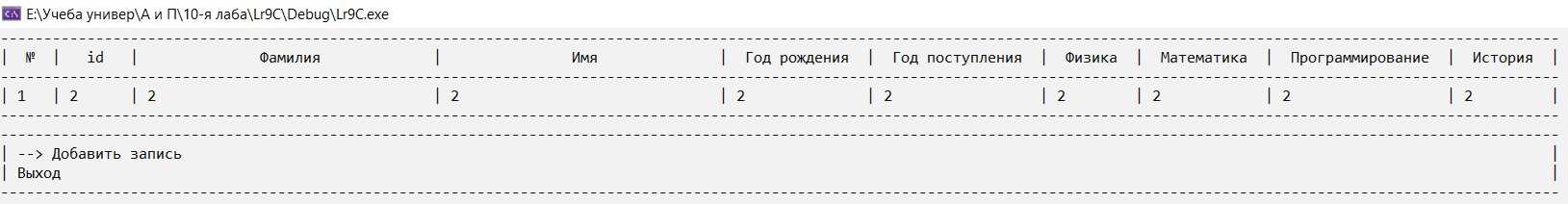


Рисунок 12 – Форма после добавления записи через нее

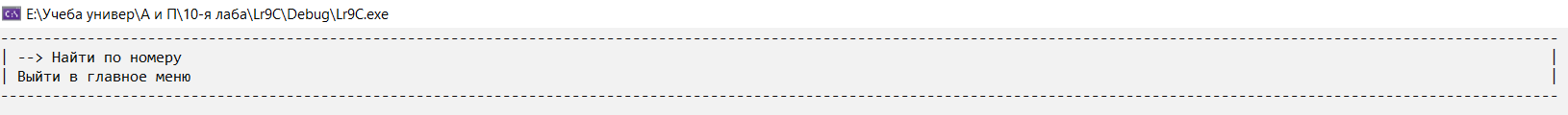


Рисунок 13 – Меню поиска и редактирования записей

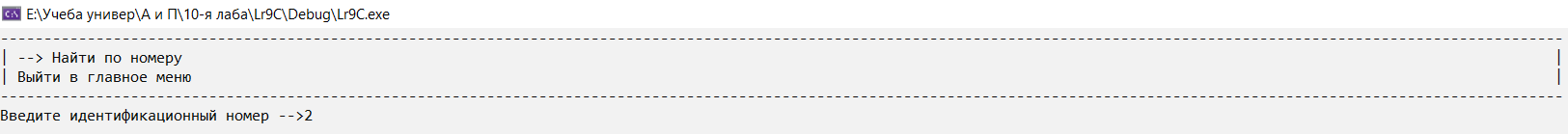


Рисунок 14 – Реализация поиска по номеру

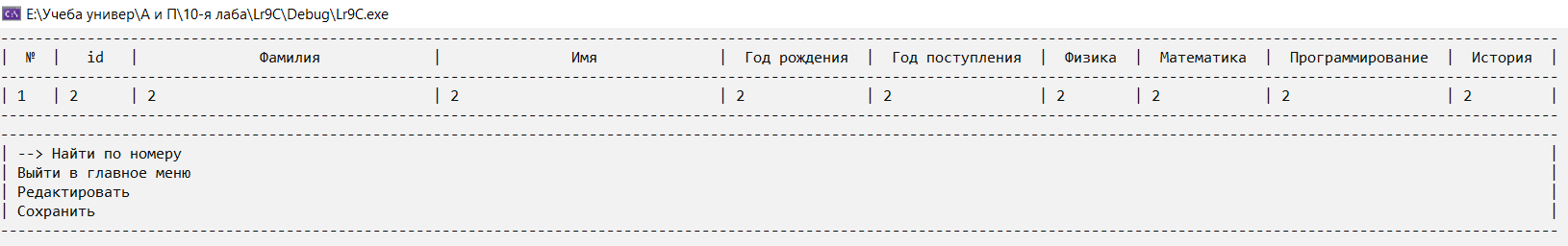


Рисунок 15 – Меню после после того как запись с таким номером найдена

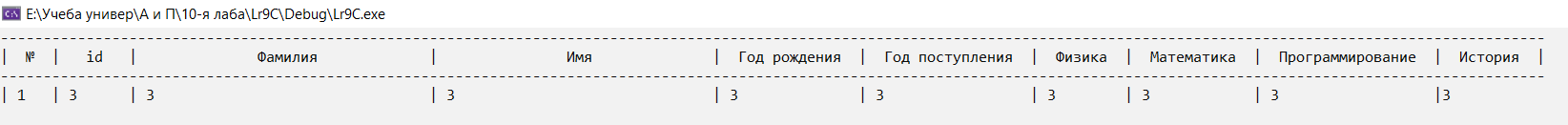


Рисунок 16 – Редактирование записи

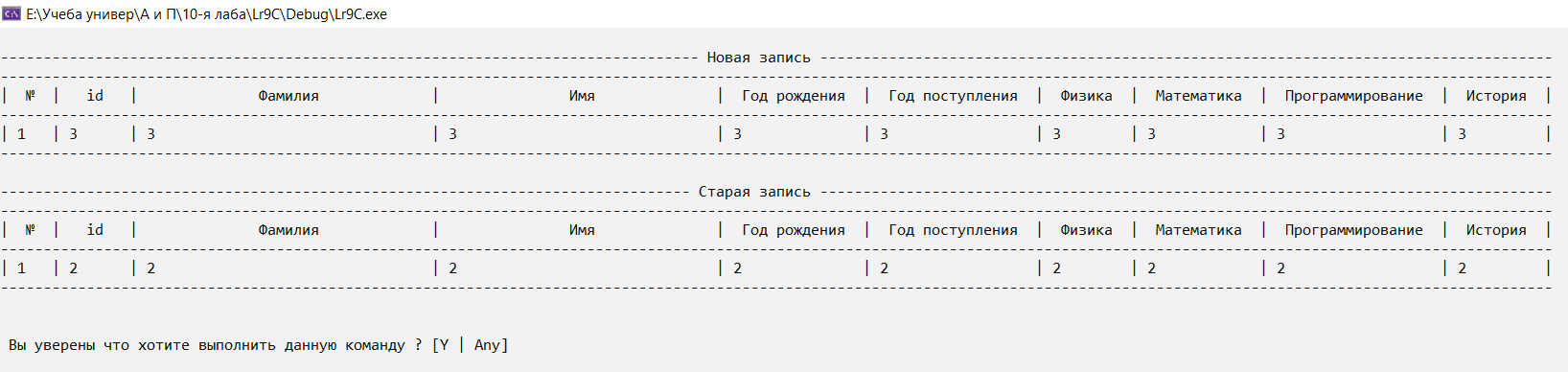


Рисунок 17 – Сохранение записи.

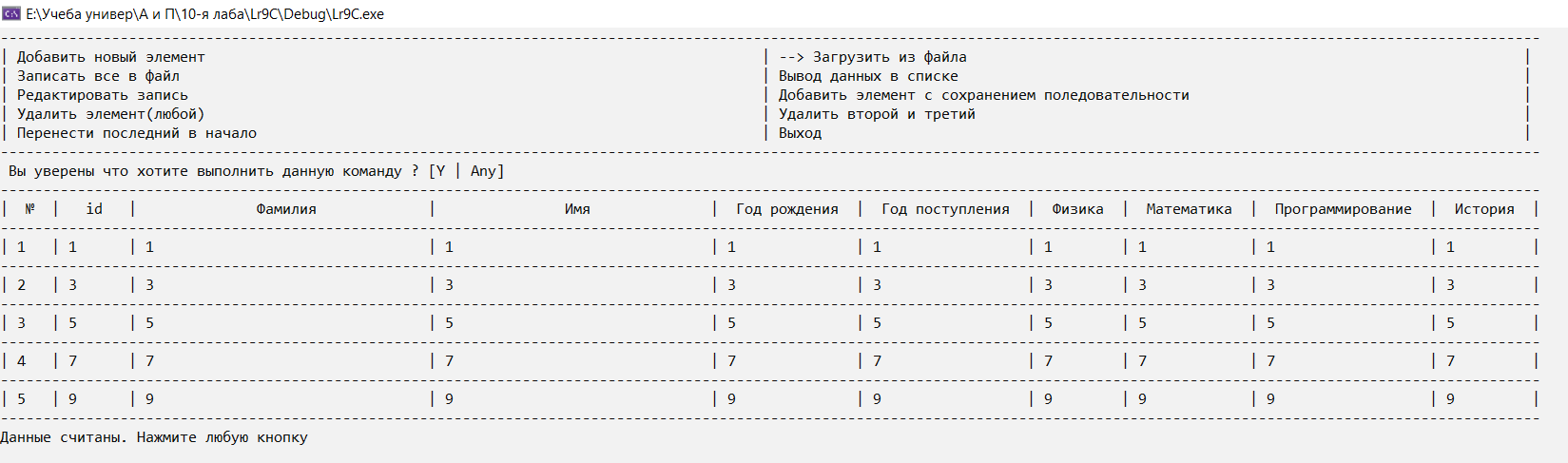


Рисунок 18 – Загрузка данных из файла

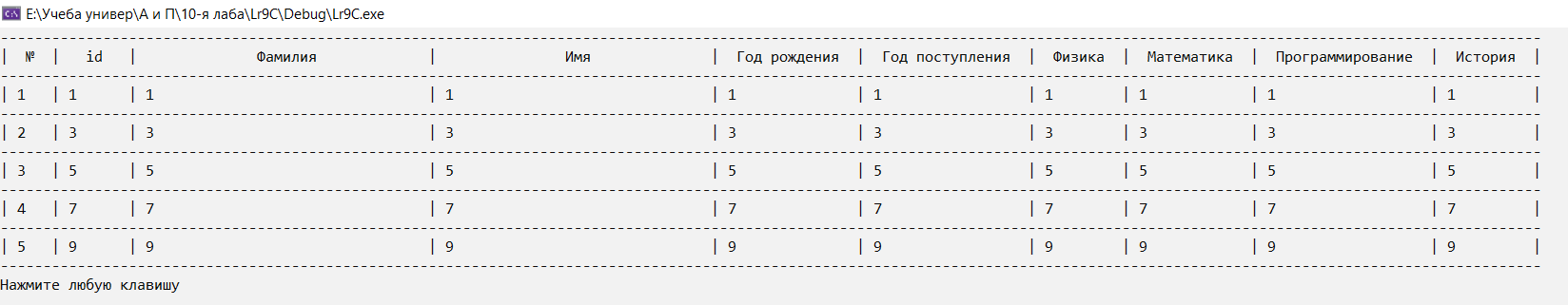
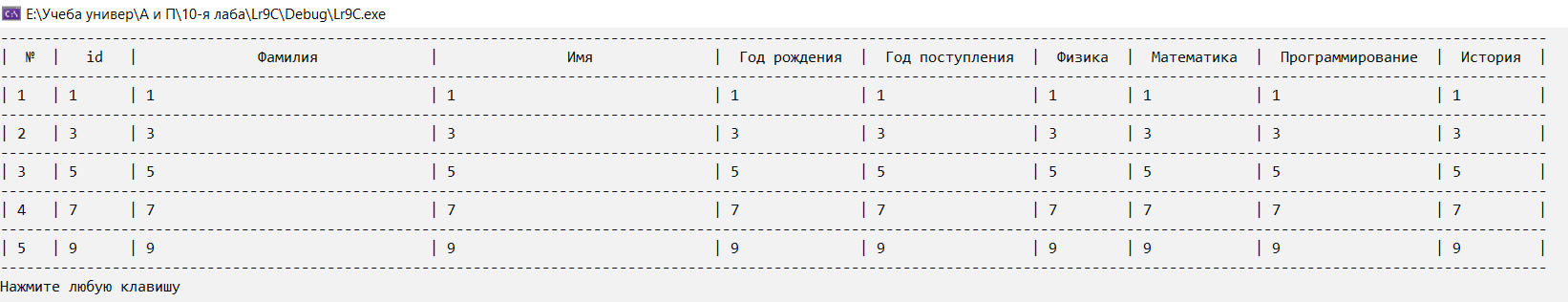
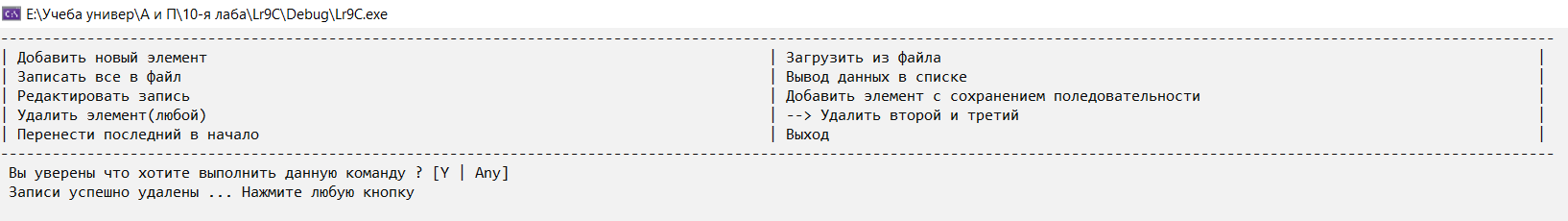


Рисунок 19 – Вывод всего файла на экран





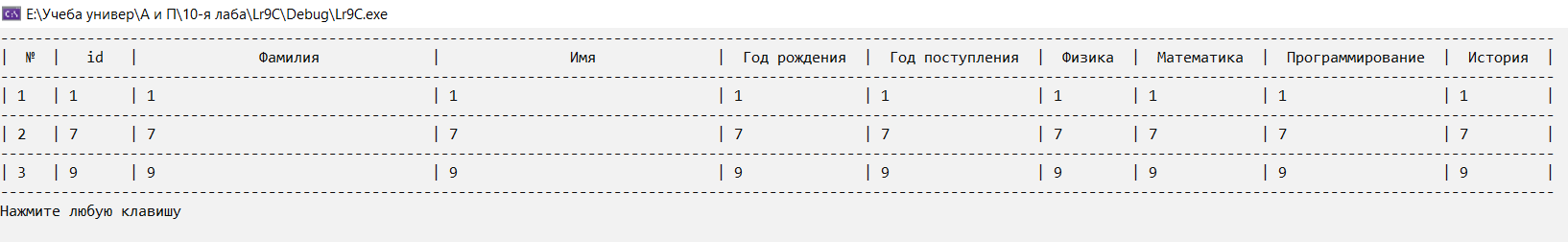
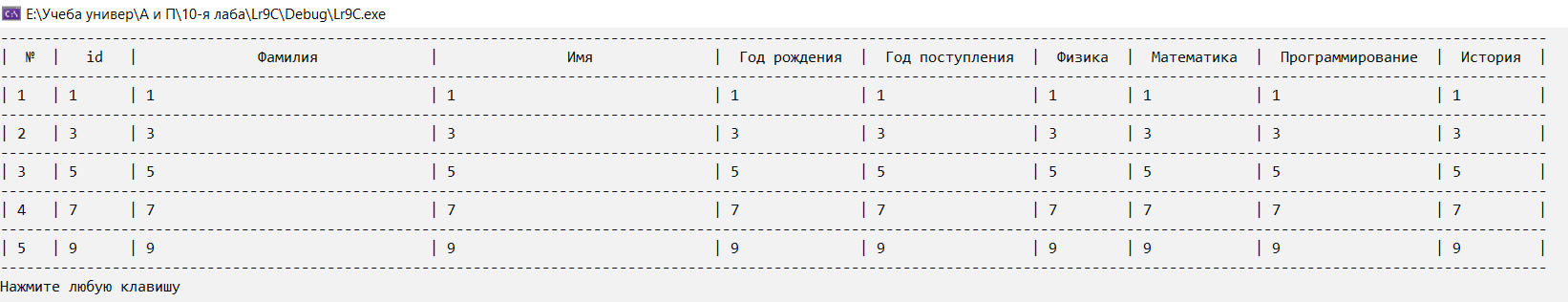
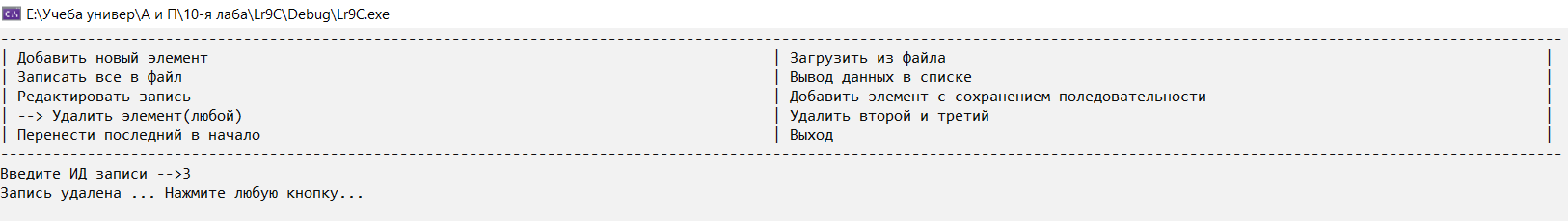


Рисунок 20 – Удаление второй и третьей записи в списке





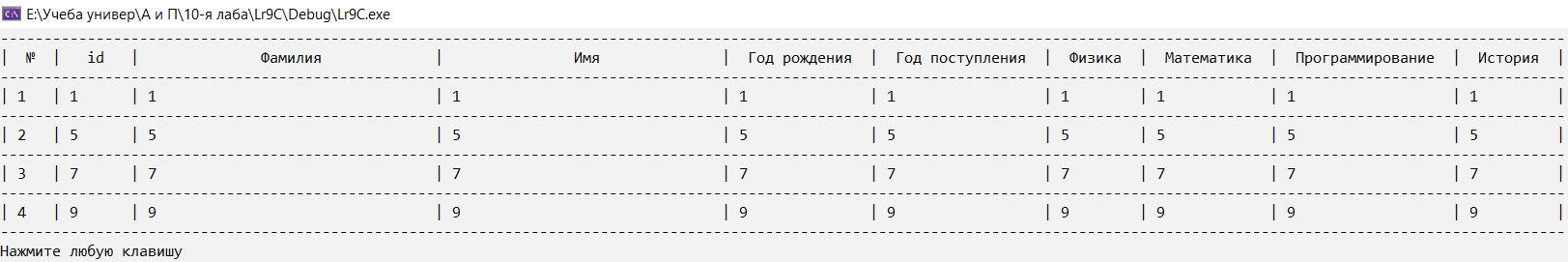
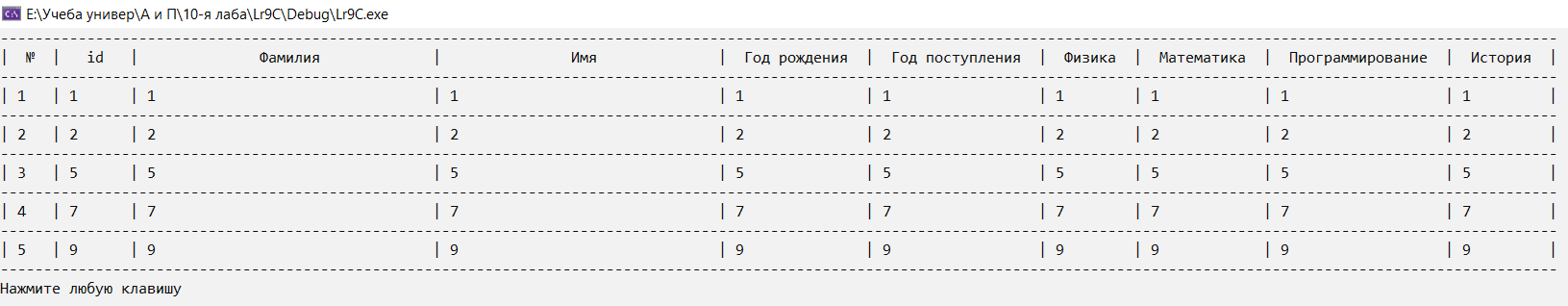


Рисунок 21 – Удаление записи по ид.



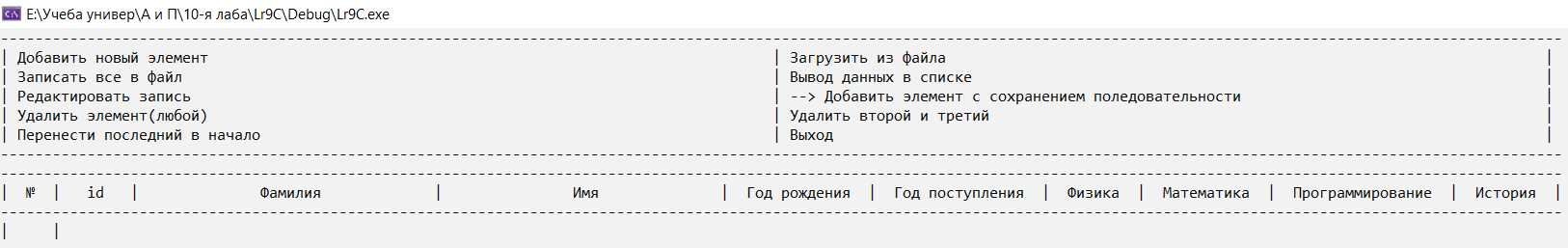
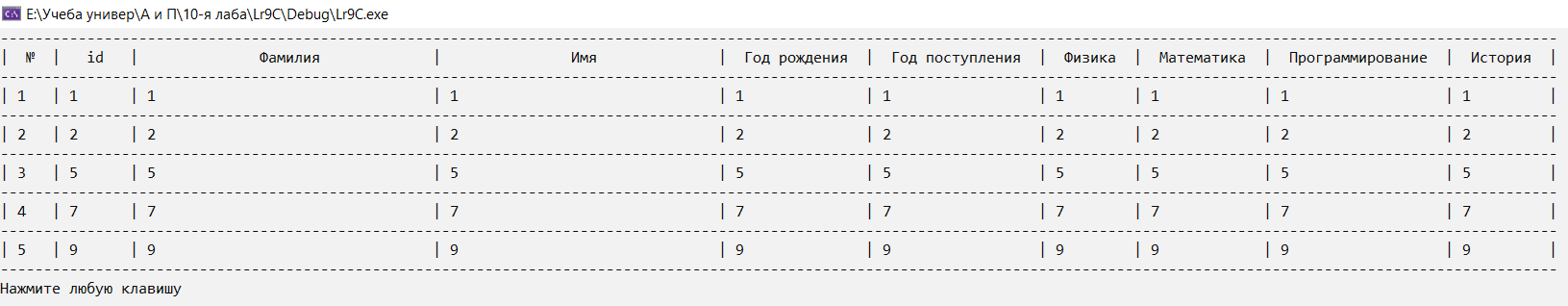
 

Рисунок 22 – Добавление записи в нужную позицию.

**Вывод**

В ходе выполнения данной лабораторной работы были изучены основные операции над бинарными файлами, принцип работы с двунаправленными списками и указателями на элемент, обработки бинарной информации полученной из файла в языке С, с учетом связи указателей. Получены практические навыки работы с файловыми указателями, двунаправленными списками, реализации алгоритмов обработки данных на прямую в элементах списка на языке С. Реализован принцип минималистического графического меню для работы в программе, написана своя библиотека.