

Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Пензенский государственный университет»

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по курсу «Программирование»

на тему «База данных Телефонная книга»

01.06.2026
от
Г. Генералова А.А.

Выполнил:

студент группы 25BVB2

Доблер Б. С.

Принял:

к.т.н., доцент Генералова А.А.

Пенза 2026

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

М.А. Митрохин

«11» февраля 2026 г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Добнеру Богдану Сергеевичу Группа 25BBB2

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

Исходные данные (технические требования) на проектирование

Программа "База данных телефонная книга"
Добавление, удаление, редактирование, поиск, сортировка записей; Фильтрация записей по определенному критерию; Создание новой базы данных; Сохранение текущей базы данных; Загрузка сохраненных баз данных.

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 "11" февраля 2026

Дата защиты проекта "01" июня 26

Руководитель Т. Усманов А. А.

Задание получил "11" февраля 2026г.

Студент Д. Доблер Богдан Сергеевич

Содержание

Введение	5
Постановка задачи	6
Выбор решения	7
Описание разработки программы	8
Отладка и тестирование.....	9
Описание программы	10
Руководство пользователя.....	18
Заключение	30
Приложение А Листинги программы	32
Приложение В Результаты работы программы	57
Приложение С Результат работы программы.....	58

Введение

В современном мире деловые и личные связи играют ключевую роль в жизни каждого человека. Ежедневно мы сталкиваемся с необходимостью хранить, обрабатывать и быстро находить контактные данные: номера телефонов, адреса электронной почты, информацию о принадлежности к определенной группе или организации. В условиях постоянного роста объема информации ручное ведение записей становится неэффективным и приводит к потере важных данных.

Именно для решения этой задачи и были созданы компьютерные базы данных. База данных — это совокупность взаимосвязанных данных, организованных по определенным правилам, которая позволяет легко добавлять, изменять, удалять и искать информацию. Технологии баз данных лежат в основе практически всех современных информационных систем: от простых списков контактов в мобильном телефоне до сложных корпоративных систем управления взаимоотношениями с клиентами.

Данная курсовая работа посвящена разработке информационно-поисковой системы «Телефонная книга». Программа представляет собой консольное приложение, реализованное на языке C с использованием структур, динамических списков и функций работы с файлами [1]. Основной целью разработки является создание удобного инструмента для ведения базы контактов, который позволяет не только хранить информацию, но и эффективно её обрабатывать: выполнять сортировку, фильтрацию и поиск по различным критериям. Особое внимание в работе уделено использованию ассемблерных вставок для реализации низкоуровневых операций (очистка экрана, звуковой сигнал, обработка строк), что позволяет глубже понять принципы работы вычислительной техники и программного обеспечения [2].

Разработанная программа может быть использована как в личных целях, так и в качестве основы для более сложных информационных систем.

Постановка задачи

Необходимо разработать программу для работы с базой данных «Телефонная книга». Программа должна иметь интуитивно понятный интерфейс. Необходимо определить пункты меню, построить алгоритмы для работы с односвязным списком, реализовать фильтрацию и сортировку, изучить функции для работы с файлами.

Требования:

- 1) Многомодульность: Программа должна быть поделена на логические модули.
- 2) Использование сложных типов данных: Динамический список, структуры.
- 3) Режим работы: Текстовый.
- 4) Интерфейс: Меню на основе клавиатуры.
- 5) Заставка: Приветственный экран.

Выбор решения

При запуске программы открывается приветственный экран. Далее появляется главное меню, состоящее из пунктов:

- 1) Добавить контакт: Пользователь вводит данные (имя, телефон, *email*, группа, избранное). Программа ищет удаленную запись ($ID = -1$) и перезаписывает её, либо создает новую.
- 2) Показать все контакты: Вывод всей базы данных в табличном виде.
- 3) Поиск и фильтрация:
 - Поиск по ID.
 - Поиск по имени (частичное совпадение).
 - Поиск по группе.
 - Фильтр "Избранные".
- 4) Сортировка:
 - По имени (А-Я).
 - По ID (убывание).
- 5) Удалить контакт: Пользователь вводит ID, программа помечает запись как удаленную ($ID = -1$).
- 6) Редактировать контакт: Пользователь находит запись по ID и вводит новые данные.
- 7) Сохранить в файл: Сохранение базы данных в текстовый файл.
- 8) Загрузить из файла: Загрузка данных из файла.
- 9) Выход.

Навигация осуществляется с помощью цифровых клавиш.

Описание разработки программы

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

- 1) `main.c` – основной файл программы. Реализует главный цикл, обработку выбора пользователя через `switch`, настройку русского языка и управление памятью.
- 2) `consts.c` – файл с константами. Содержит строковые сообщения для вывода пользователю («Операция выполнена успешно!», «Телефонная книга пуста» и т.д.) и переменную `GETCHAR_HANDLER` для обработки буфера ввода.
- 3) `consts.h` – заголовочный файл с константами. Определяет размеры строковых буферов, коды цветов и объявляет внешние переменные из `consts.c`.
- 4) `phonebook.h` – заголовочный файл с определением структуры `Contact`. Содержит поля: `ID`, имя, телефон, `email`, группа, статус избранного и указатель на следующий элемент.
- 5) `list.c` – модуль для работы с односвязным списком. Содержит функции инициализации элемента, добавления в конец, логического удаления по `ID`, поиска по `ID`, подсчёта активных записей, освобождения памяти и ассемблерную функцию обмена указателей [3].
- 6) `list.h` – заголовочный файл для модуля `list.c`. Содержит объявления всех функций работы со списком.
- 7) `file_io.c` – модуль для работы с файлами. Реализует сохранение базы данных в файл и загрузку базы данных из файла [4].
- 8) `file_io.h` – заголовочный файл для модуля `file_io.c`. Содержит объявления функций сохранения и загрузки.
- 9) `ui.c` – модуль пользовательского интерфейса. Реализует все диалоговые окна: добавление, редактирование, вывод всех контактов, меню поиска и сортировки. Также содержит ассемблерные функции очистки экрана и звукового сигнала, а также функцию цветного вывода [5].

- 10) `ui.h` – заголовочный файл для модуля `ui.c`. Содержит объявления всех функций интерфейса.
- 11) `utils.c` – модуль вспомогательных функций. Содержит ассемблерную функцию удаления символов новой строки, ассемблерную функцию сравнения строк и функции сравнения для сортировки (по имени и по ID) [6].
- 12) `utils.h` – заголовочный файл для модуля `utils.c`. Содержит объявления всех вспомогательных функций.

По порядку реализации модулей можно понять, что использовался нисходящий подход программирования. Этот метод основан на том, что сначала проектируются основные компоненты программы (структуры и функции работы с ними), а затем уже дорабатываются детали интерфейса и вспомогательные функции. Это позволяет упростить тестирование, так как каждый модуль тестируется сразу после его реализации, что помогает в создании более структурированных программ [7].

Отладка и тестирование

В качестве среды разработки была выбрана программа `Microsoft Visual Studio 2022`. Программа обладает всеми необходимыми средствами для отладки: точки останова, трассировка, анализ переменных.

Тестирование проводилось во время разработки и после. Основные проблемы были связаны с корректной обработкой строк, а также с правильным освобождением памяти.

Описание программы

Приложение main.exe является основным модулем. При запуске выводится заставка. После этого пользователю предлагается главное меню (таблица 1, таблица 2, таблица 3).

Таблица 1 — Работа главного меню

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1, Enter	Выбран пункт «Добавить контакт»	Запускается диалог добавления контакта.
2, Enter	Выбран пункт «Показать все контакты»	На экран выводятся все контакты из базы.
3, Enter	Выбран пункт «Поиск и фильтрация»	Открывается меню поиска (Таблица 2).
4, Enter	Выбран пункт «Сортировка»	Открывается меню сортировки (Таблица 3).
5, Enter	Выбран пункт «Удалить контакт»	Запускается диалог удаления (ввод ID).
6, Enter	Выбран пункт «Редактировать контакт»	Запускается диалог изменения записи.
7, Enter	Выбран пункт «Сохранить в файл»	Запись базы данных в файл.
8, Enter	Выбран пункт «Загрузить из файла»	Загрузка базы данных из файла.
0, Enter	Выбран пункт «Выход»	Завершение работы программы.

Таблица 2 — Поиск и фильтрация

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1, Enter	Поиск по ID	Выводится контакт с указанным ID.
2, Enter	Поиск по имени	Выводятся контакты, в имени которых есть подстрока.
3, Enter	Поиск по группе	Выводятся контакты, принадлежащие указанной группе.
4, Enter	Показать избранные	Выводятся только контакты с флагом isFavorite=1.

Таблица 3 — Сортировка

Клавиши, событие	вызывающее	Действие пользователя	Действие программы
1, Enter		Сортировать по имени (А-Я)	Контакты сортируются по алфавиту.
2, Enter		Сортировать по ID (убывание)	Контакты сортируются от большого ID к меньшему.

Далее на рисунке 1 представлена схема взаимодействия программы (рисунок 1).

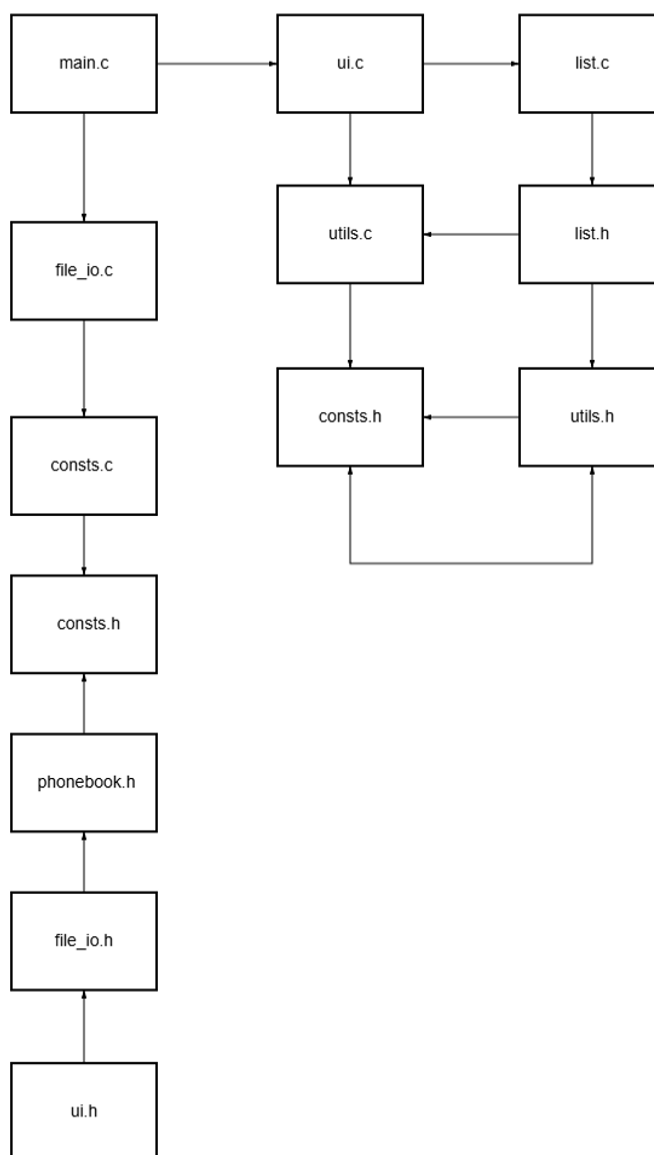


Рисунок 1 - Схема взаимодействия программы

Ниже, на рисунке 2 представлена схема данных (рисунок 2).



Рисунок 2 - Схема данных

Схема программы. Функция Main (рисунок 3)

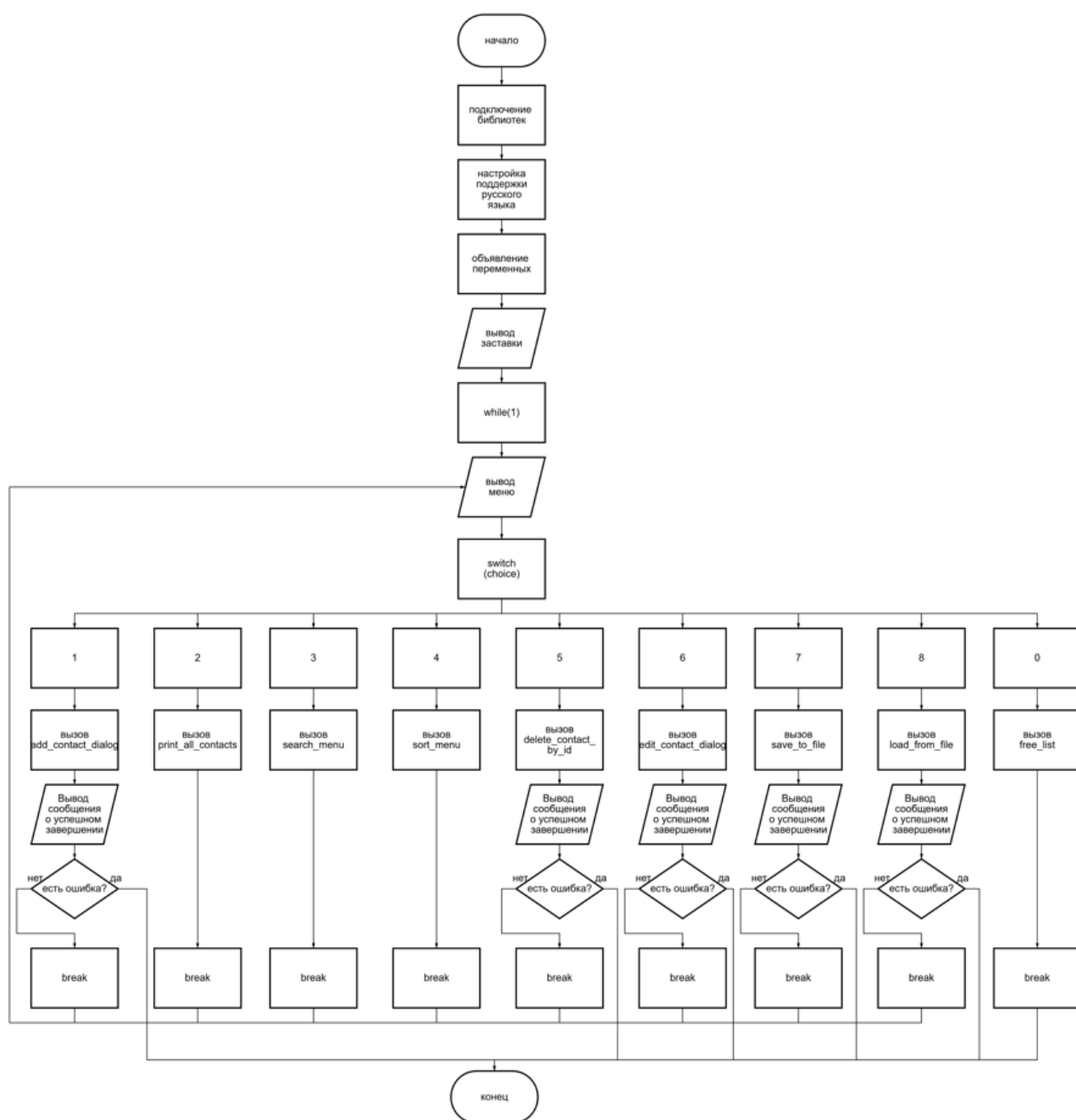


Рисунок 3 - Функция main

Схема программы. Функция ui.c (рисунок 4)

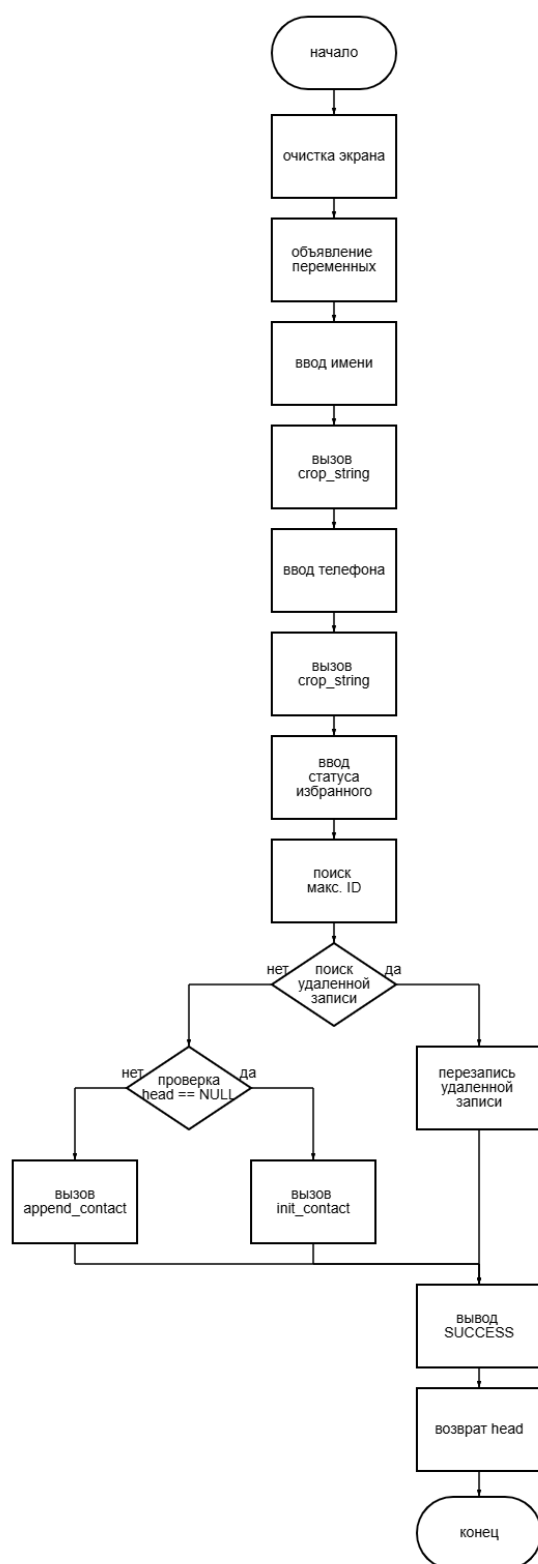


Рисунок 4 - Функция ui.c

Схема программы. Функция list.c (рисунок 5)

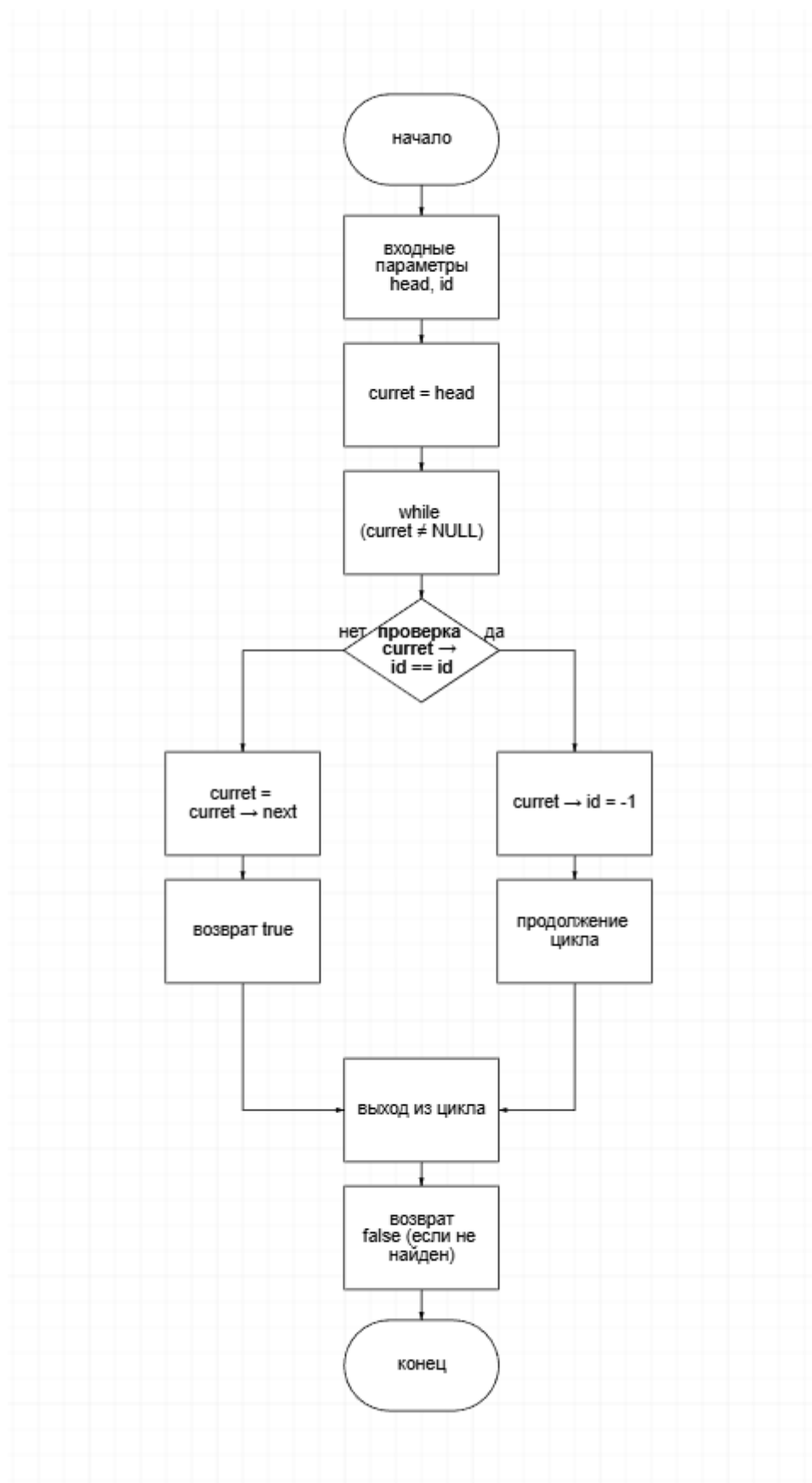


Рисунок 5 – Функция list.c

Ниже приведена иерархическая схема программы (рисунок 6).

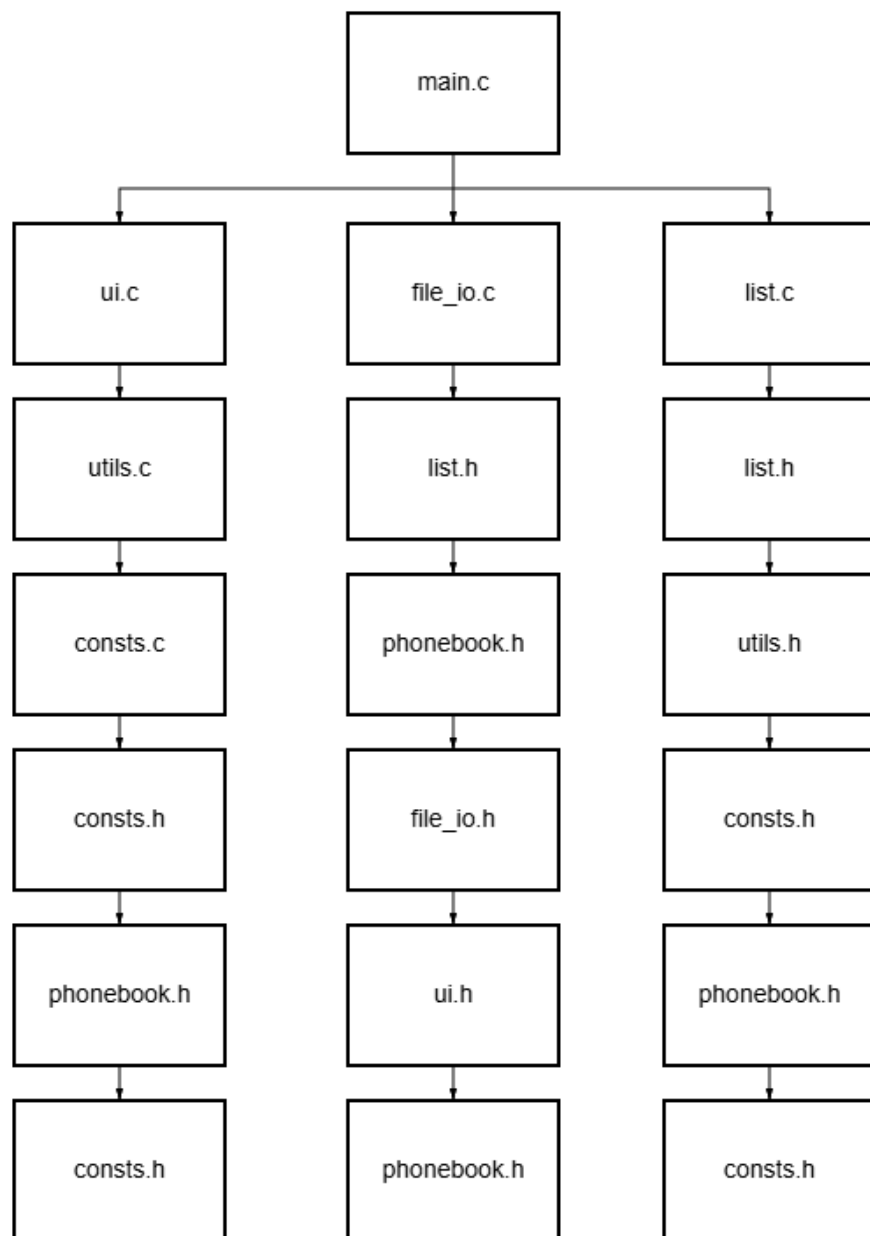


Рисунок 6 – иерархическая схема программы

На схеме показана структура подчинения модулей. Главный модуль main.c вызывает ui.c, file_io.c и list.c. Модуль ui.c использует list.c и utils.c. Все модули нижнего уровня используют consts.h и phonebook.h.

Руководство пользователя

Программа предназначена для хранения информации о контактах. Программа имеет интуитивно понятный интерфейс и поддерживает такие операции, как добавление новой записи в базу данных, удаление записи из базы данных, изменение записи в базе данных, поиск записи в базе данных, сохранение существующей базы данных в файл, импорт базы данных из файла, сортированный вывод записей базы данных и фильтрация записей в базе данных.

Для открытия и вывода на экран существующего каталога необходимо выбрать пункт меню «Показать все контакты» (рисунок 7, рисунок 8).

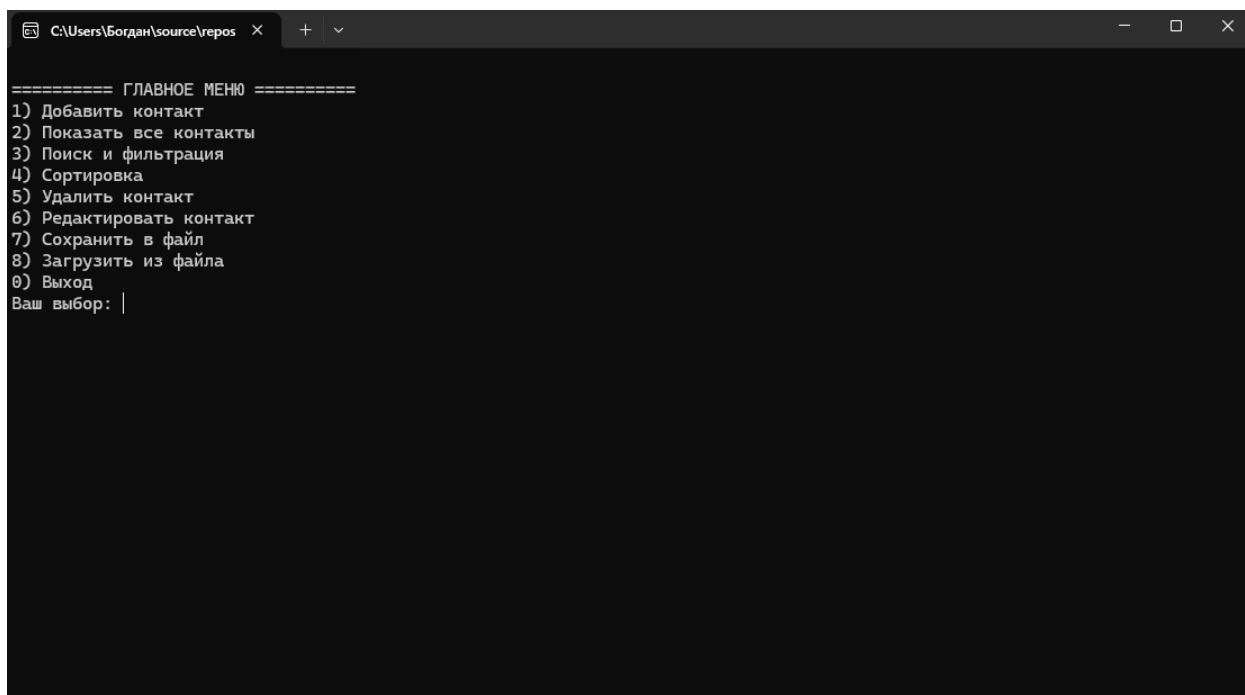
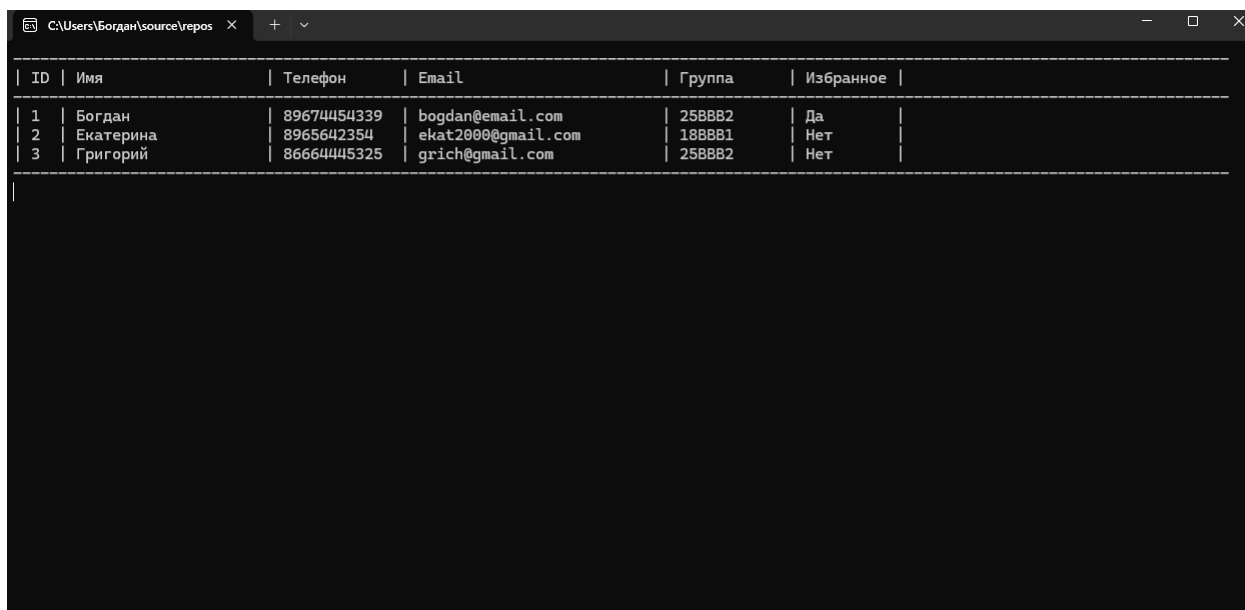


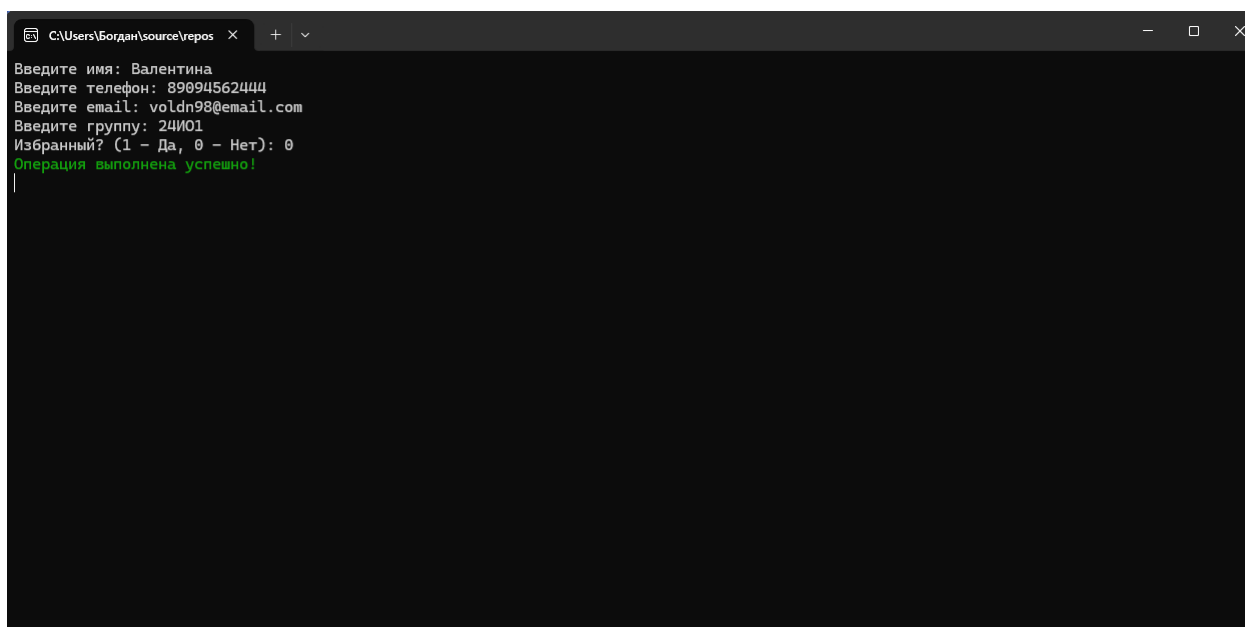
Рисунок 7 – Главное меню программы



ID	Имя	Телефон	Email	Группа	Избранное
1	Богдан	89674454339	bogdan@email.com	258882	Да
2	Екатерина	8965642354	ekat2000@gmail.com	188881	Нет
3	Григорий	86664445325	grich@gmail.com	258882	Нет

Рисунок 8 – Показать все контакты

Для добавления записи в каталог необходимо выбрать пункт меню «Добавить контакт». Далее необходимо заполнить форму структуры из пяти пунктов: ввести имя контакта, нажать клавишу Enter, ввести номер телефона, нажать клавишу Enter, ввести адрес электронной почты, нажать клавишу Enter, ввести группу, нажать клавишу Enter, ввести статус избранного (1 — Да, 0 — Нет), нажать клавишу Enter (рисунок 9). В случае, если не возникли ошибки, на экране появится надпись «Операция выполнена успешно!».



```

C:\Users\Богдан\source\repos x + v
Введите имя: Валентина
Введите телефон: 89094562444
Введите email: voldn98@email.com
Введите группу: 24ИЮ1
Избранный? (1 - Да, 0 - Нет): 0
Операция выполнена успешно!

```

Рисунок 9 – Добавление контакта

Для удаления записи из каталога необходимо выбрать пункт меню «Удалить контакт» (рисунок 10). Далее необходимо ввести ID удаляемой записи (рисунок 11) и нажать клавишу Enter. В случае, если не возникли ошибки, на экране появится надпись «Операция выполнена успешно!» (рисунок 12). Если контакт с указанным ID не найден, программа завершит работу.

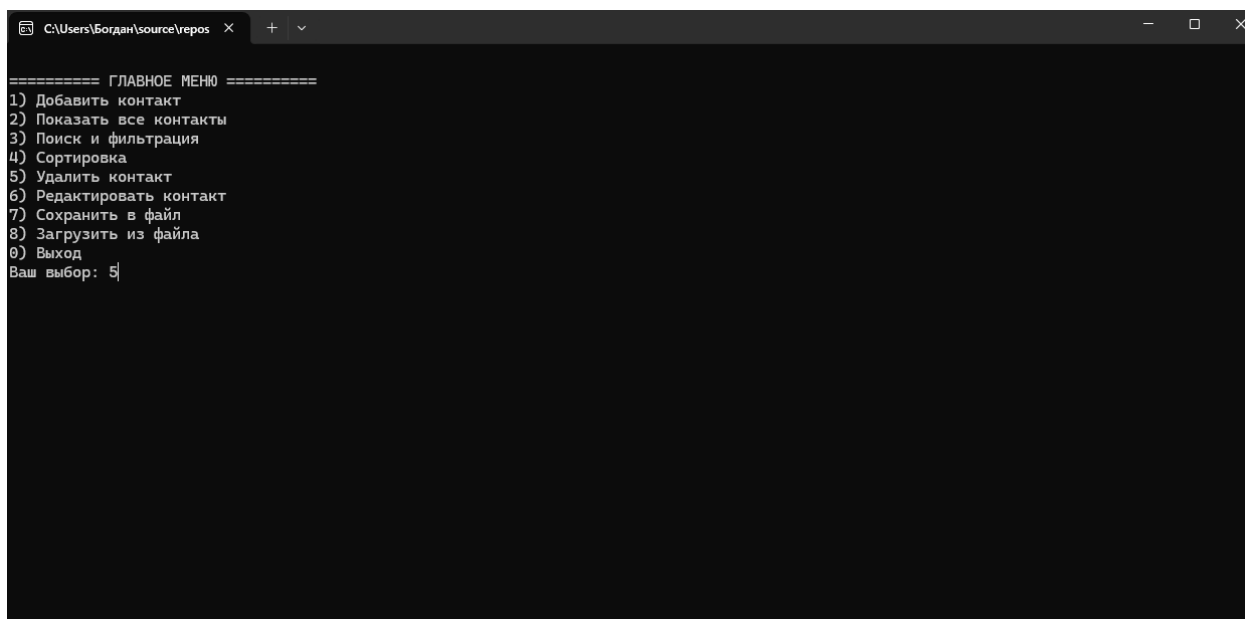


Рисунок 10 –Пункт удалить контакт

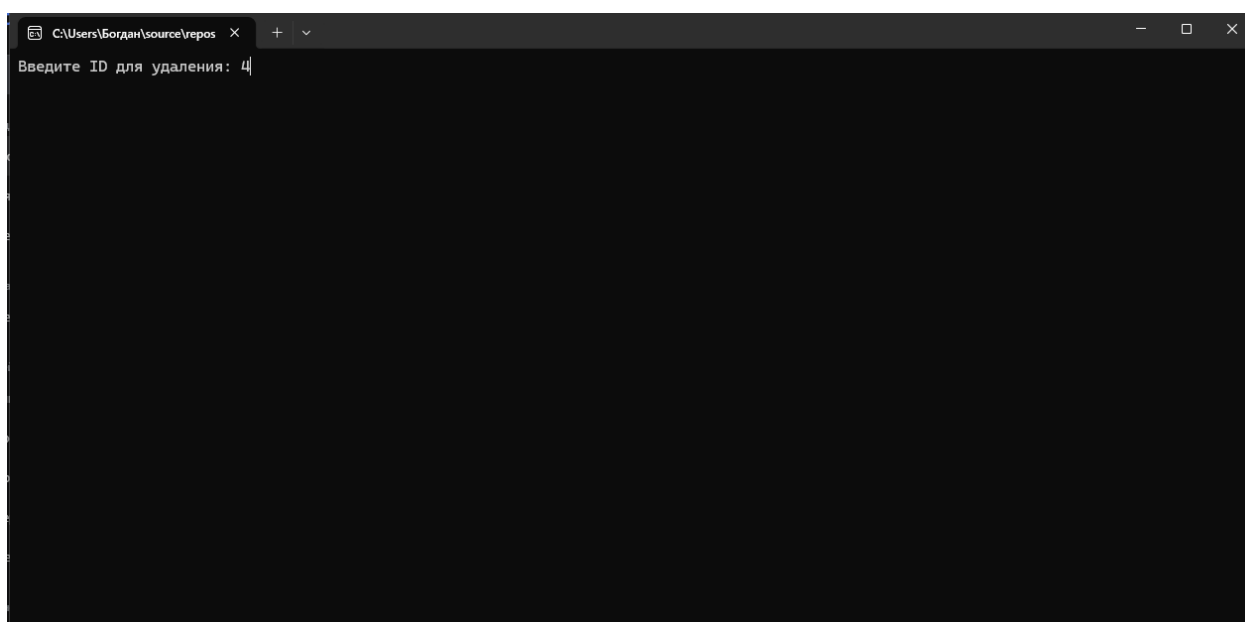


Рисунок 11 – ввод ID

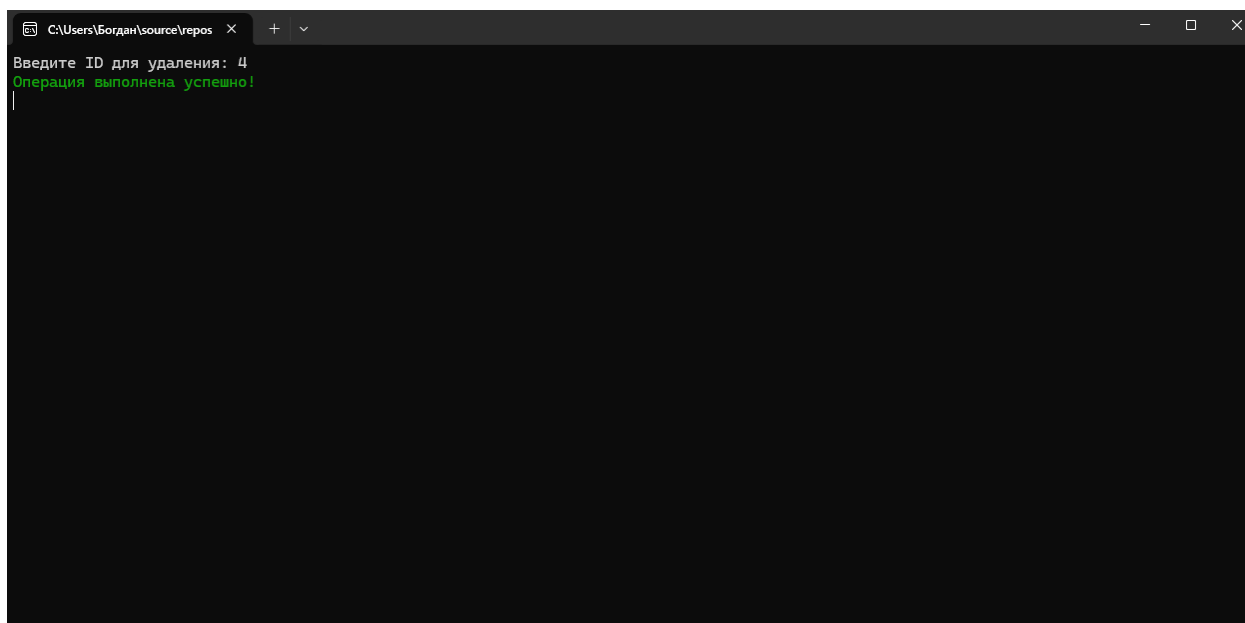


Рисунок 12 – Удаление контакта

Для изменения записи необходимо выбрать пункт меню «Редактировать контакт» (рисунок 13). Далее необходимо ввести ID изменяемой записи и ввести новые данные для экземпляра из пяти пунктов: ввести новое имя, нажать клавишу Enter, ввести новый телефон, нажать клавишу Enter, ввести новый email, нажать клавишу Enter, ввести новую группу, нажать клавишу Enter, ввести новый статус избранного (1 — Да, 0 — Нет), нажать клавишу Enter. В случае, если не возникли ошибки, на экране появится надпись «Операция выполнена успешно!» (рисунок 14).

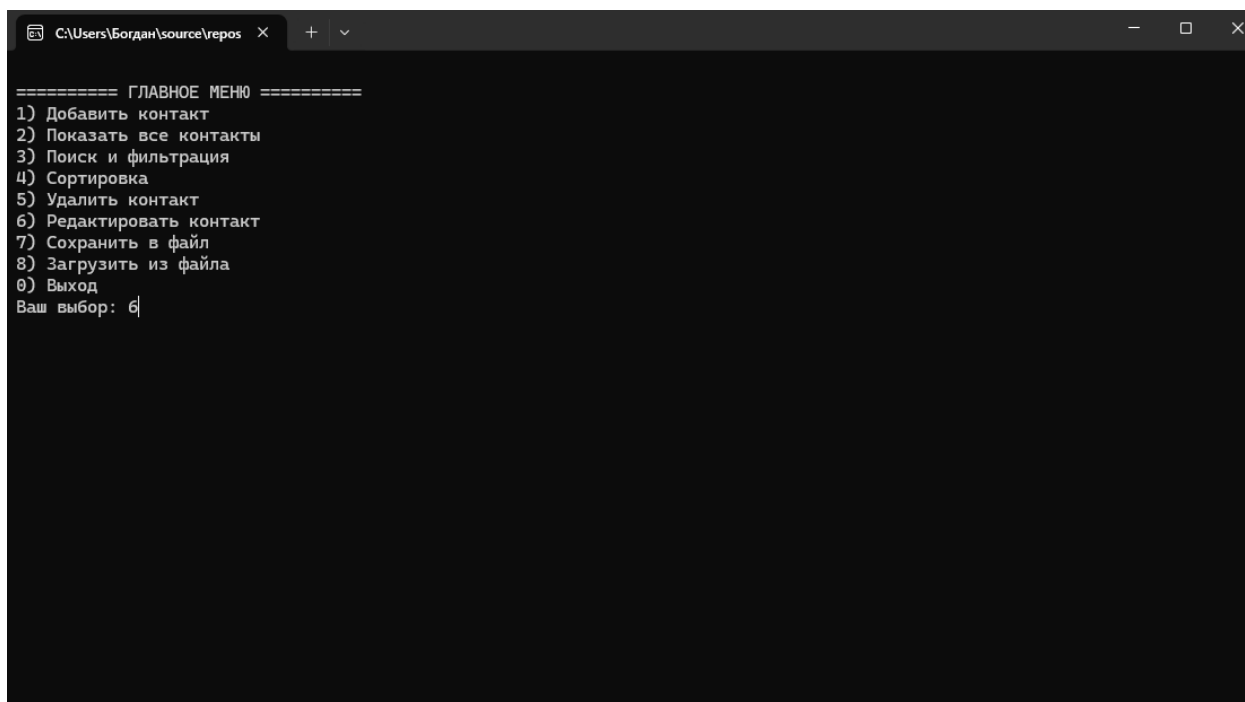


Рисунок 13 – Пункт Редактировать контакт

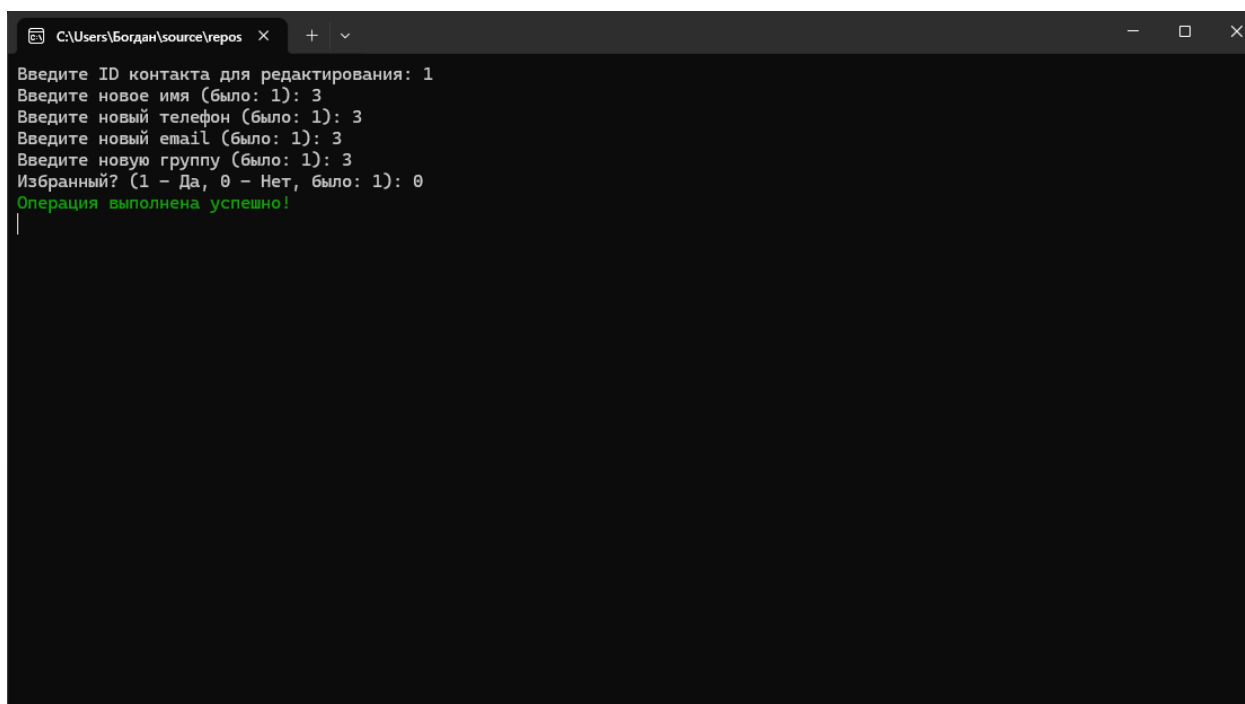


Рисунок 14 – Редактирование контакта

Для того чтобы сохранить текущую базу данных в файл, необходимо выбрать пункт меню «Сохранить в файл» (рисунок 15). Далее ввести имя файла, в который будет записана база данных. Если файл уже существует, то программа удалит информацию из файла и запишет в него каталог. В случае,

если не возникли ошибки, на экране появится надпись «Операция выполнена успешно!» (рисунок 16).

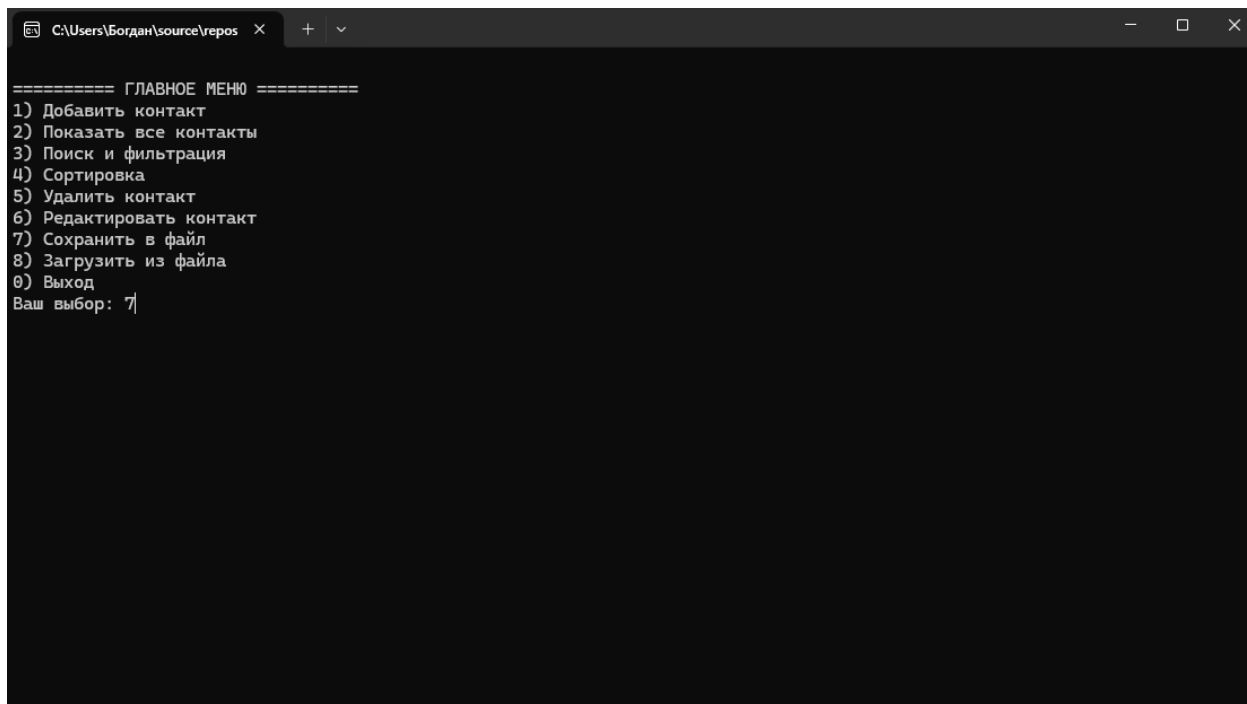


Рисунок 15 – Пункт сохранить в файл

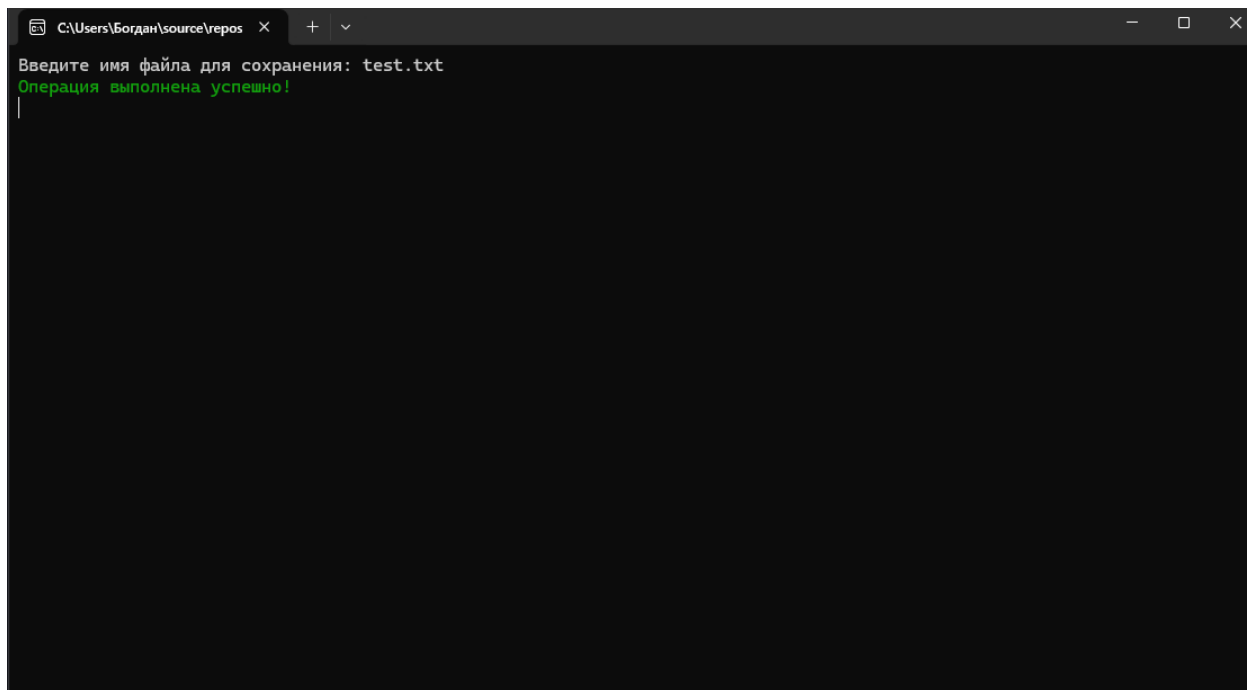


Рисунок 16 – Сохранение базы данных в файл

Для того чтобы импортировать базу данных из файла, необходимо выбрать пункт меню «Загрузить из файла» (рисунок 17). Далее ввести имя

файла, из которого произойдет чтение данных, нажать клавишу Enter. Текущая база данных будет заменена на новую из файла. В случае ошибки программа завершит работу. В случае, если не возникли ошибки, на экране появится надпись «Операция выполнена успешно!» (рисунок 18).

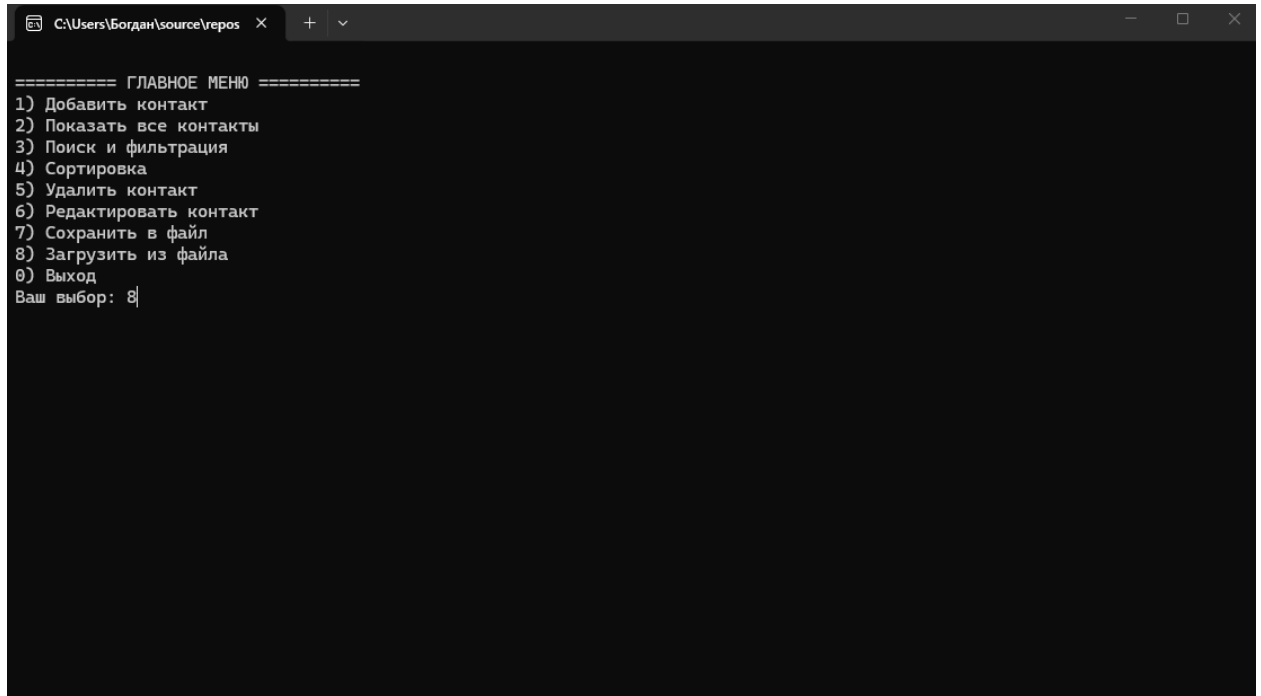


Рисунок 17 – Пункт загрузить из файла

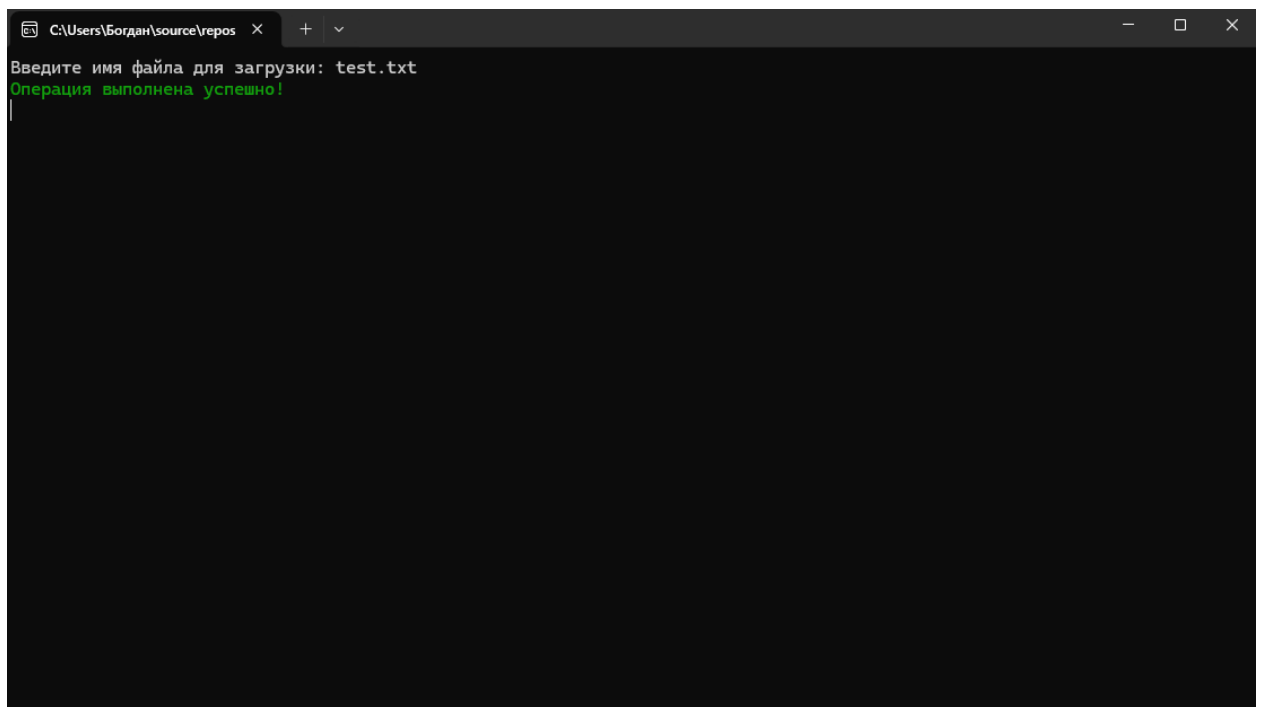


Рисунок 18 – Загрузка базы данных из файла

Для того чтобы записи были выведены в отсортированном виде, необходимо в главном меню выбрать пункт «Сортировка» (рисунок 19). В открывшемся меню выбрать один из двух параметров сортировки: «По имени (А-Я)» или «По ID (убывание)», нажать клавишу Enter (рисунок 20).

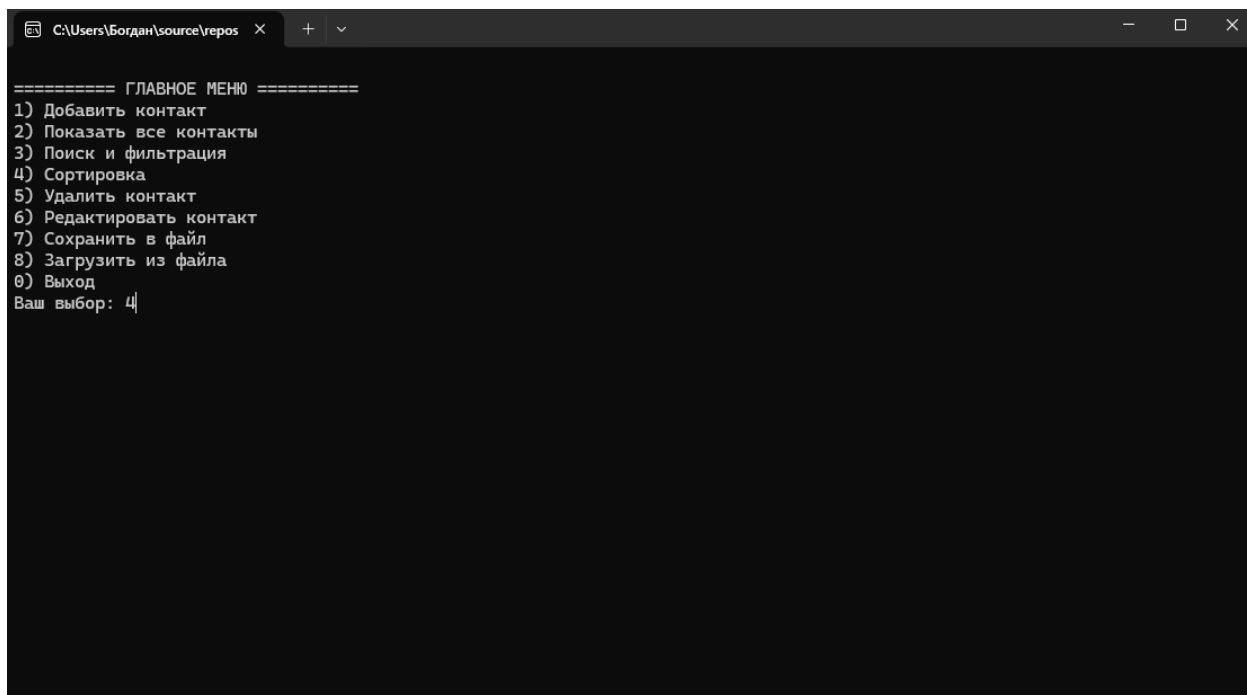


Рисунок 19 – Пункт сортировка

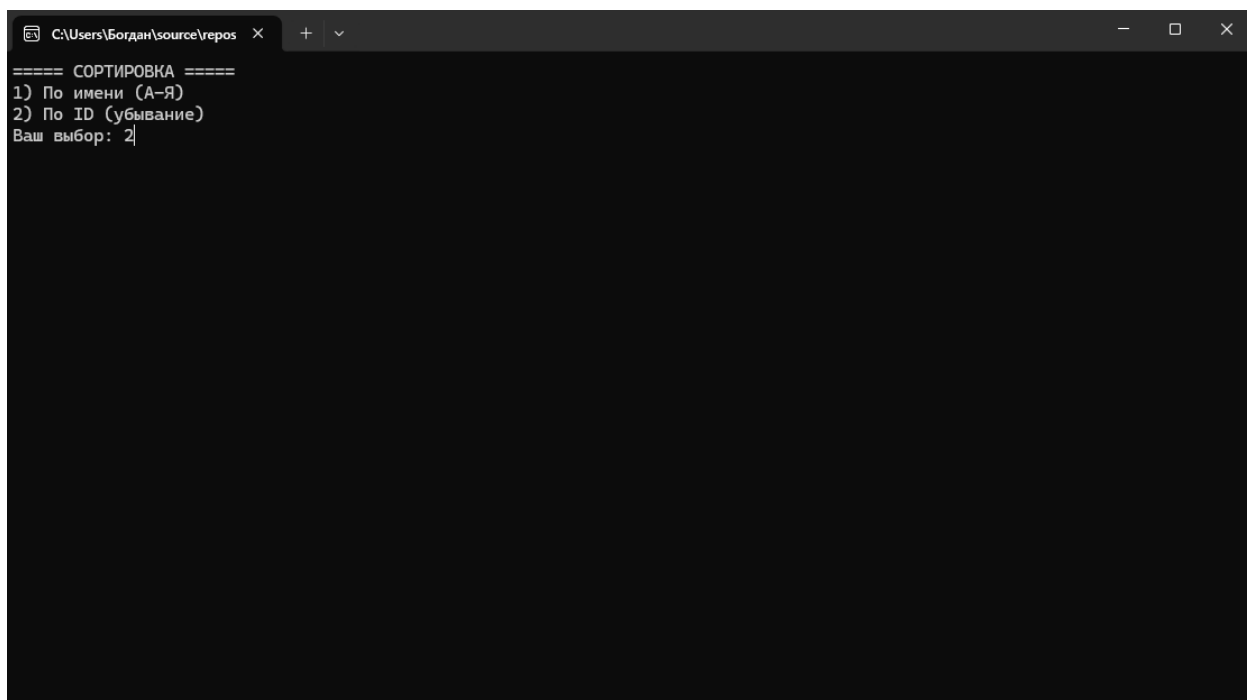


Рисунок 20 – Меню сортировки

Для того чтобы найти определенную запись или найти группу записей по критерию, необходимо выбрать пункт меню «Поиск и фильтрация» (рисунок 21). В открывшемся меню выбрать один из четырех пунктов: «Поиск по ID», «Поиск по имени», «Поиск по группе», «Показать только избранные» (рисунок 22).

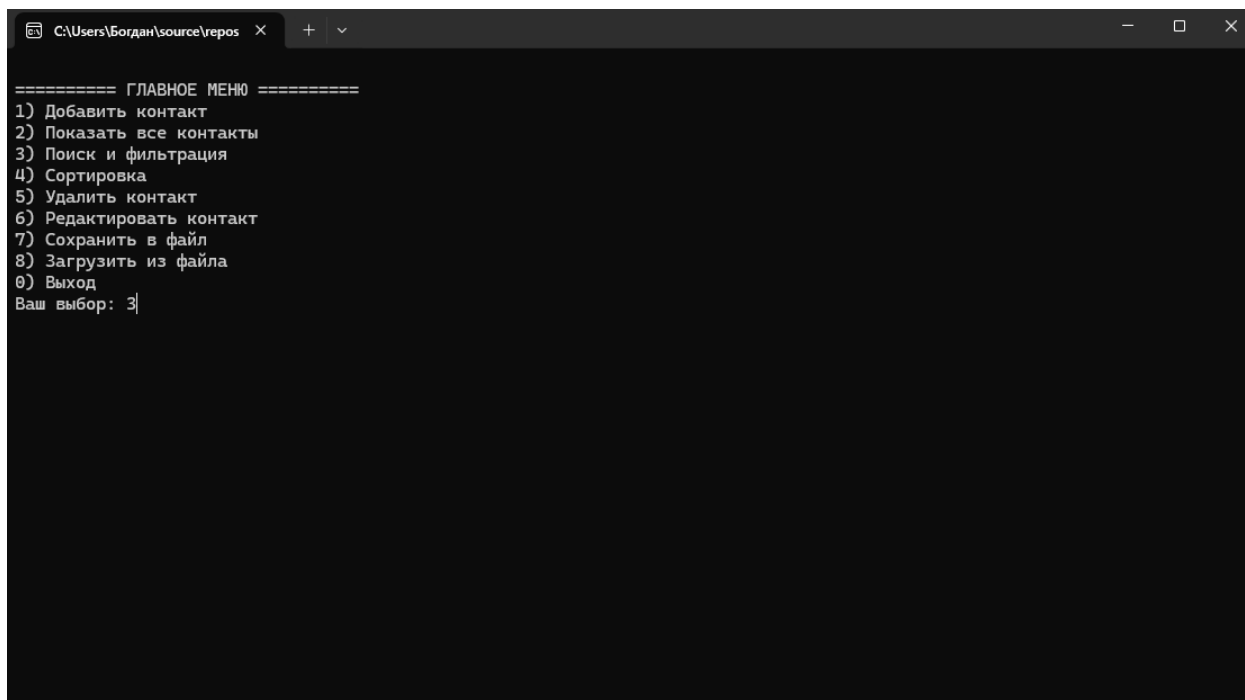


Рисунок 21 – Пункт поиск и фильтрация

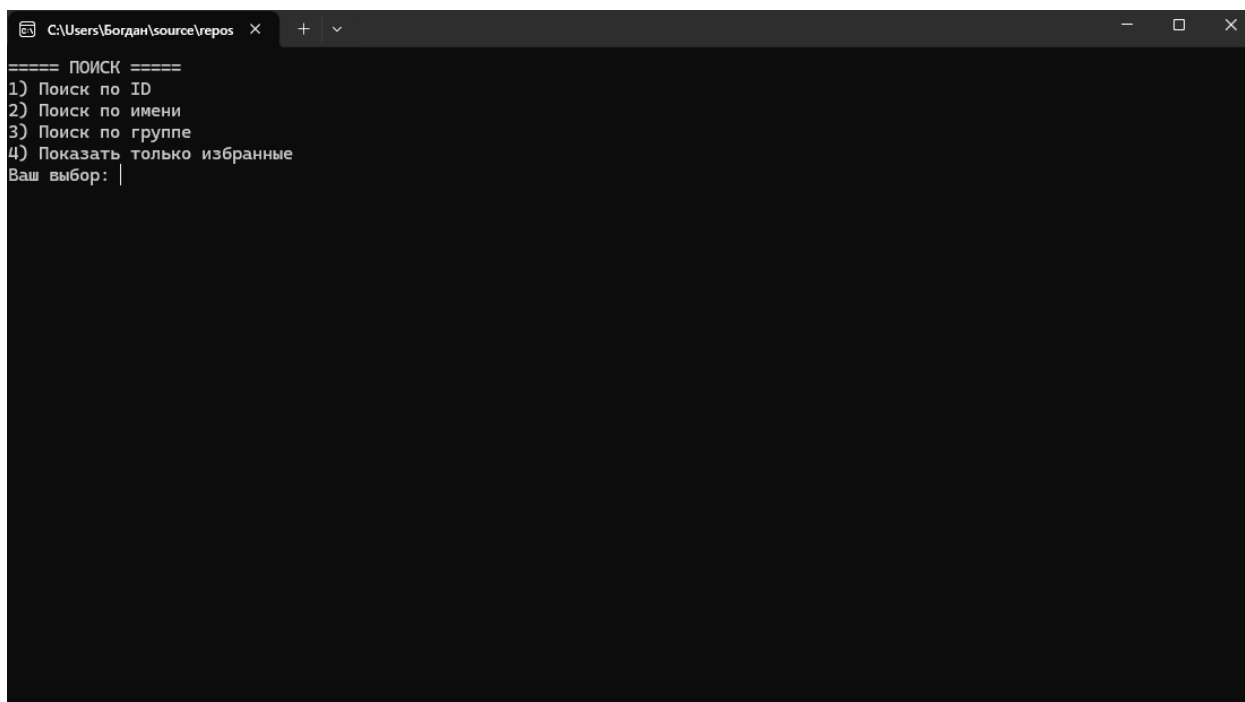


Рисунок 22 – Меню поиска и фильтрации

Поиск по ID предназначен для поиска конкретной записи. Пользователю следует ввести ID искомой записи, нажать клавишу Enter. В случае, если запись находится в базе данных, программа выведет её на экран, в противном случае выдаст сообщение «Контакт с указанным ID не найден».

Поиск по имени предназначен для поиска записей по имени. Пользователь может ввести как полное имя, так и его начальную часть, далее нужно нажать клавишу Enter. Если записи с данным именем есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст сообщение «Телефонная книга пуста».

Поиск по группе предназначен для фильтрации записей по группам. Пользователь может ввести как полное название группы, так и его начальную часть, далее нужно нажать клавишу Enter. Если записи с данной группой есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст сообщение «Телефонная книга пуста» (рисунок 23).

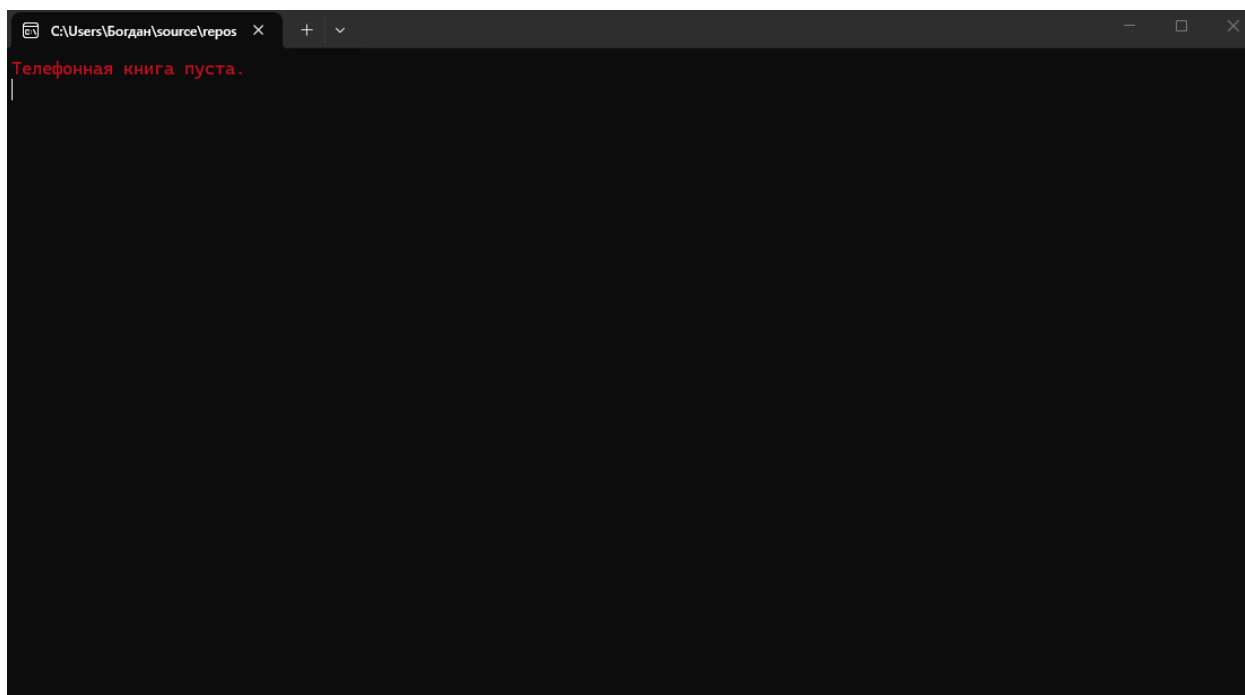


Рисунок 23 – Сообщение «телефонная книга пуста»

Показать только избранные предназначен для вывода только тех записей, которые отмечены как избранные. Программа выводит все контакты, у которых флаг избранного установлен в 1.

Для выхода из программы необходимо выбрать пункт меню «Выход» (рисунок 24) и нажать клавишу Enter. Программа завершит работу, освободив память, занятую списком контактов. На экране появится сообщение «Программа завершена. До свидания!» (рисунок 25).

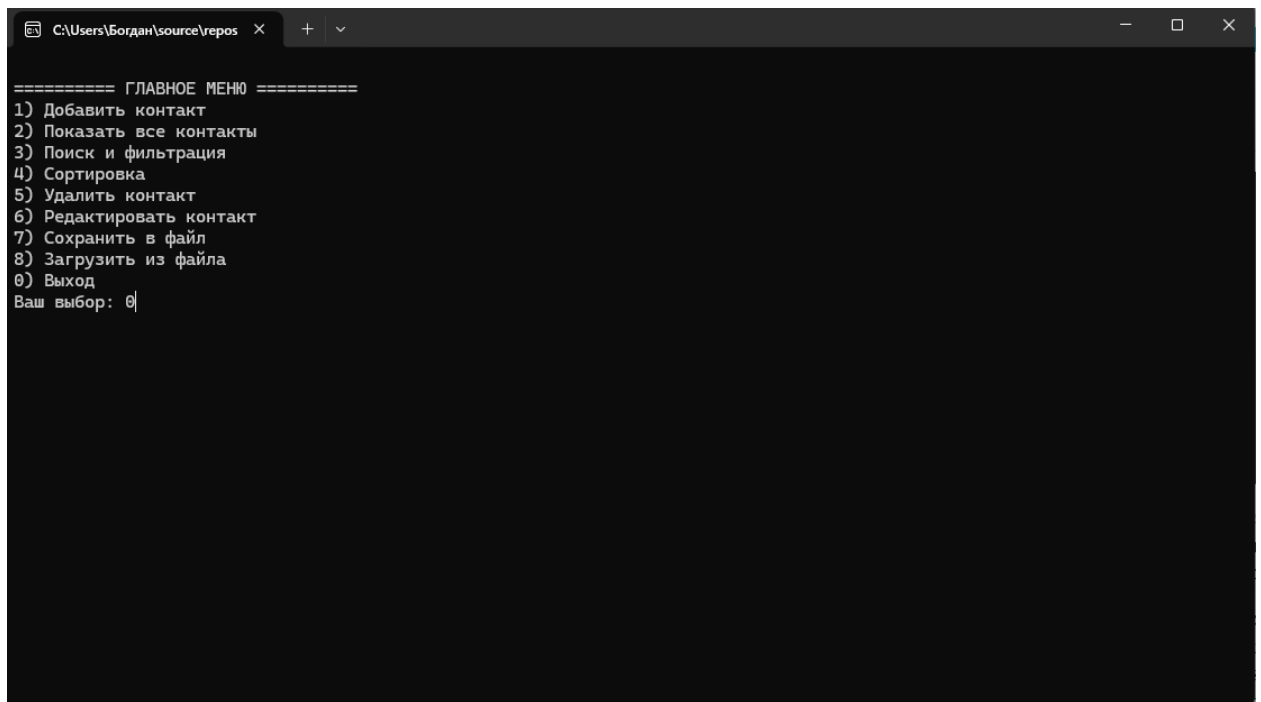


Рисунок 24 – Пункт выход

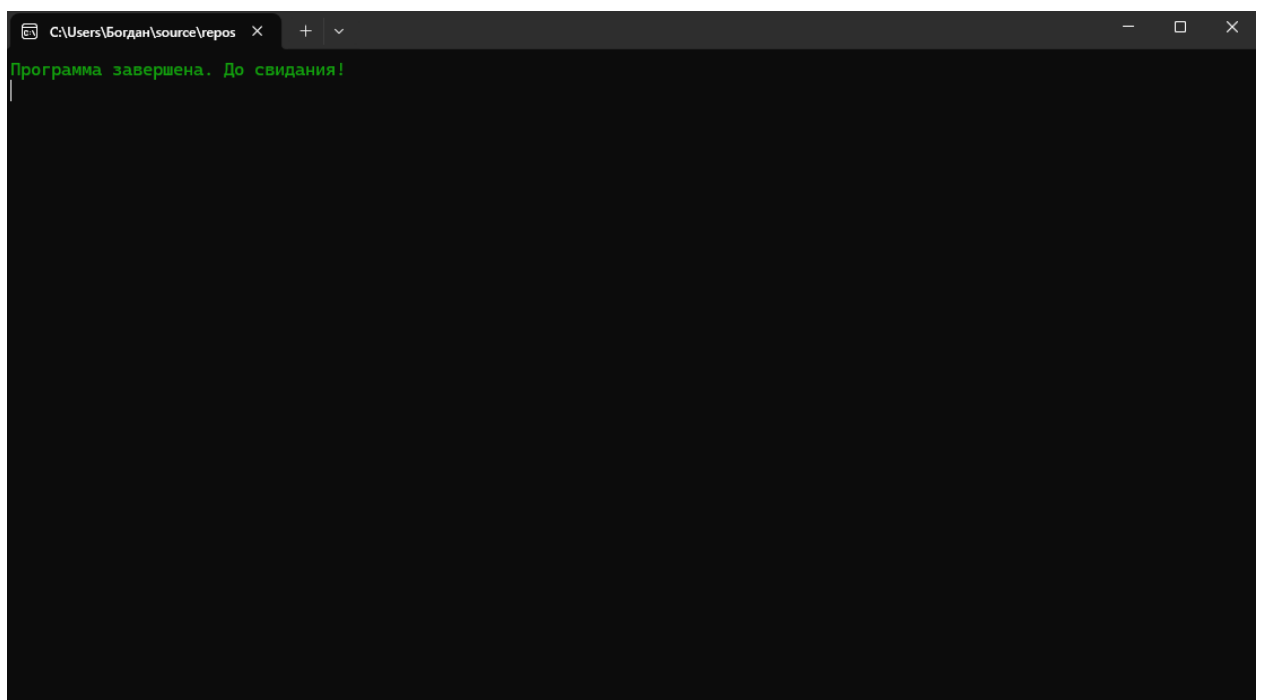


Рисунок 25 – Завершение программы

Заключение

В ходе выполнения данной курсовой работы были получены навыки разработки многомодульных программ, изучены функции работы с консолью, файлами, оперативной памятью (динамические списки). Также были закреплены навыки отладки и тестирования в Visual Studio.

Была написана программа для работы с базой данных "Телефонная книга", которая выполняет все заявленные функции: добавление, поиск, сортировку, удаление и редактирование записей, а также их сохранение и загрузку.

В дальнейшем программу можно улучшить путем добавления графического интерфейса и поддержки работы с мышью.

Список используемых источников

1. Керниган Б., Ритчи Д. Язык программирования С. – 1985 г.
2. MSDN.
3. Прата С. Язык программирования С++. – 2019 г.
4. Тюгашев А.А. Языки программирования. Учебное пособие. – 2018 г.
5. Аблязов Р. Программирование на ассемблере на платформе x86-64. – 2016 г.
6. Трофимов С.В. Структуры данных и алгоритмы в программировании. – 2018 г.
7. Шилдт Г. Полный справочник по языку С. – 2005 г.

Приложение А Листинги программы

Файл main.c

```
#include <stdio.h>
#include <locale.h>
#include <windows.h>
#include "phonebook.h"
#include "list.h"
#include "ui.h"
#include "file_io.h"
#include "consts.h"

int main() {
    setlocale(LC_ALL, "Rus");
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

    show_splash();

    Contact* phonebook = NULL;
    char choice;
    char filename[MAX_FILENAME];
    int id;
    bool running = true;

    while (running) {
        system("cls");
        show_main_menu();
        choice = getchar();

        GETCHAR_HANDLER = getchar();
```

```

switch (choice) {
case '1':
phonebook = add_contact_dialog(phonebook);
break;
case '2':
print_all_contacts(phonebook);
break;
case '3':
search_menu(phonebook);
break;
case '4':
sort_menu(&phonebook);
break;
case '5':
system("cls");
printf("Введите ID для удаления: ");
scanf_s("%d", &id);

        GETCHAR_HANDLER = getchar();
if (delete_contact_by_id(phonebook, id))
color_print(SUCCESS, COLOR_GREEN);
else {
beep_asm();
color_print(NOT_FOUND, COLOR_RED);
        }

        GETCHAR_HANDLER = getchar();
break;
case '6':
edit_contact_dialog(phonebook);
break;
case '7':
system("cls");

```

```

printf("Введите имя файла для сохранения: ");
fgets(filename, MAX_FILENAME, stdin);
crop_string(filename);
if (save_to_file(phonebook, filename))
color_print(SUCCESS, COLOR_GREEN);
else {
beep_asm();
color_print(FILE_ERROR, COLOR_RED);

        }

        GETCHAR_HANDLER = getchar();

break;

case '8':

system("cls");

printf("Введите имя файла для загрузки: ");
fgets(filename, MAX_FILENAME, stdin);
crop_string(filename);

        {

                Contact* loaded = load_from_file(filename);

if (loaded) {
free_list(phonebook);
phonebook = loaded;
color_print(SUCCESS, COLOR_GREEN);

                }

else {
beep_asm();
color_print(FILE_ERROR, COLOR_RED);

                }

        }

        GETCHAR_HANDLER = getchar();

break;

case '0':

```

```

running = false;
break;
default:
beep_asm();
color_print("Неверный ввод. Попробуйте снова.", COLOR_RED);
GETCHAR_HANDLER = getchar();
        }
    }

free_list(phonebook);
system("cls");
color_print("Программа завершена. Досвидания!", COLOR_GREEN);
getchar();
return 0;
}

```

Файл phonebook.h

```

#ifndef PHONEBOOK_H
#define PHONEBOOK_H

#include <stdbool.h>
#include "consts.h"

typedef struct Contact {
    int id;
    char name[LEN_NAME];
    char phone[LEN_PHONE];
    char email[LEN_EMAIL];
    char group[LEN_GROUP];
    bool isFavorite;
    struct Contact* next;
} Contact;

```

```
#endif
```

Файл consts.c

```
#include "consts.h"
```

```
charSUCCESS[] = "Операция выполнена успешно!";
```

```
charFAIL[] = "При выполнении операции произошла ошибка.";
```

```
char EMPTY_LIST[] = "Телефонная книга пуста.";
```

```
char NOT_FOUND[] = "Контакт с указанным ID не найден.";
```

```
char FILE_ERROR[] = "Ошибка при работе с файлом.";
```

```
char GETCHAR_HANDLER;
```

Файл consts.h

```
#ifndef CONSTS_H
```

```
#define CONSTS_H
```

```
#define LEN_NAME      30
```

```
#define LEN_PHONE     20
```

```
#define LEN_EMAIL     30
```

```
#define LEN_GROUP     20
```

```
#define MAX_FILENAME 50
```

```
#define COLOR_RED     1
```

```
#define COLOR_GREEN   2
```

```
#define COLOR_DEFAULT 0
```

```
extern char SUCCESS[];
```

```
extern char FAIL[];
```

```
extern char EMPTY_LIST[];
```

```
extern char NOT_FOUND[];
```

```
extern char FILE_ERROR[];
```

```
extern char GETCHAR_HANDLER;
```

```
#endif
```

Файл file_io.c

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include "file_io.h"
```

```
#include "list.h"
```

```
boolsave_to_file(Contact* head, const char* filename) {
```

```
    FILE* file;
```

```
    if (fopen_s(&file, filename, "w") != 0) return false;
```

```
    fprintf_s(file, "-START-\n");
```

```
    Contact* current = head;
```

```
    while (current != NULL) {
```

```
        if (current->id != -1) {
```

```
            fprintf_s(file, "%d\n%s\n%s\n%s\n%s\n%d\n-CONTINUE-\n",
```

```
                current->id, current->name, current->phone,
```

```
                current->email, current->group, current->isFavorite);
```

```
        }
```

```
        current = current->next;
```

```
    }
```

```
    fclose(file);
```

```
    return true;
```

```
}
```

```
Contact* load_from_file(const char* filename) {
```

```
    FILE* file;
```

```
    if (fopen_s(&file, filename, "r") != 0) return NULL;
```

```

        Contact* head = init_contact(0, "", "", "", "", false);
if (!head) { fclose(file); return NULL; }

char buffer[256];
fgets(buffer, sizeof(buffer), file);

int id, isFav;
char name[LEN_NAME], phone[LEN_PHONE], email[LEN_EMAIL],
group[LEN_GROUP];

while (fscanf_s(file, "%d\n", &id) == 1) {
    fgets(name, LEN_NAME, file); name[strcspn(name, "\n")] = 0;
    fgets(phone, LEN_PHONE, file); phone[strcspn(phone, "\n")] = 0;
    fgets(email, LEN_EMAIL, file); email[strcspn(email, "\n")] = 0;
    fgets(group, LEN_GROUP, file); group[strcspn(group, "\n")] = 0;
    fscanf_s(file, "%d\n", &isFav);
    fgets(buffer, sizeof(buffer), file);
    append_contact(head, id, name, phone, email, group,
        (bool)isFav);
    }
fclose(file);

    Contact* real_head = head->next;
free(head);
return real_head;
}

```

Файл file_io.h

```

#ifndef FILE_IO_H
#define FILE_IO_H

#include "phonebook.h"

```

```
boolsave_to_file(Contact* head, const char* filename);
Contact* load_from_file(const char* filename);
```

```
#endif
```

Файл list.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "list.h"
```

```
Contact* init_contact(int id, const char* name, const char*
phone,
const char* email, const char* group, boolisFav) {
    Contact* new_contact = (Contact*)malloc(sizeof(Contact));
    if (new_contact) {
        new_contact->id = id;
        strcpy_s(new_contact->name, LEN_NAME, name);
        strcpy_s(new_contact->phone, LEN_PHONE, phone);
        strcpy_s(new_contact->email, LEN_EMAIL, email);
        strcpy_s(new_contact->group, LEN_GROUP, group);
        new_contact->isFavorite = isFav;
        new_contact->next = NULL;
    }
    return new_contact;
}
```

```
void append_contact(Contact* head, int id, const char* name,
const char* phone, const char* email, const char* group,
boolisFav) {
    Contact* new_contact = init_contact(id, name, phone, email,
group, isFav);
    if (!new_contact) return;
```

```

    Contact* current = head;
while (current->next != NULL)
current = current->next;
current->next = new_contact;
}

booldelete_contact_by_id(Contact* head, int id) {
    Contact* current = head;
while (current != NULL) {
    if (current->id == id) {
        current->id = -1;
        return true;
    }
    current = current->next;
}
return false;
}

Contact* find_contact_by_id(Contact* head, int id) {
    Contact* current = head;
while (current != NULL) {
    if (current->id == id && id != -1)
        return current;
    current = current->next;
}
return NULL;
}

intcount_active_contacts(Contact* head) {
    int count = 0;
    Contact* current = head;

```

```

while (current != NULL) {
    if (current->id != -1) count++;
    current = current->next;
}

return count;
}

void free_list(Contact* head) {
    Contact* tmp;
    while (head != NULL) {
        tmp = head;
        head = head->next;
        free(tmp);
    }
}

void swap_pointers(Contact** a, Contact** b) {
    __asm {
        moveax, a
        movebx, b
        movecx, [eax]
        movedx, [ebx]
        mov[eax], edx
        mov[ebx], ecx
    }
}

```

Файл list.h

```

#ifndef LIST_H
#define LIST_H

#include "phonebook.h"

```

```

Contact* init_contact(int id, const char* name, const char*
phone,
const char* email, const char* group, boolisFav);

voidappend_contact(Contact* head, int id, const char* name,
const char* phone, const char* email, const char* group,
boolisFav);

booldelete_contact_by_id(Contact* head, int id);
Contact* find_contact_by_id(Contact* head, int id);
intcount_active_contacts(Contact* head);
voidfree_list(Contact* head);
voidswap_pointers(Contact** a, Contact** b); // ассемблерная

#endif

```

Файл ui.c

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#include "ui.h"
#include "list.h"
#include "utils.h"
#include "consts.h"

// очисткаэкрана
voidclrscr_asm() {
    HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
    CONSOLE_SCREEN_BUFFER_INFO csbi;
    DWORD count;

```

```

    DWORD cellCount;

    COORD homeCoords = { 0, 0 };

    if (hConsole == INVALID_HANDLE_VALUE) return;
    if (!GetConsoleScreenBufferInfo(hConsole, &csbi)) return;
    cellCount = csbi.dwSize.X * csbi.dwSize.Y;
    if (!FillConsoleOutputCharacter(hConsole, (TCHAR)' ', cellCount,
    homeCoords, &count)) return;

    if (!FillConsoleOutputAttribute(hConsole, csbi.wAttributes,
    cellCount, homeCoords, &count)) return;

    SetConsoleCursorPosition(hConsole, homeCoords);
}

// звуковой сигнал
void beep_asm() {
    __asm {
    mov al, 0B6h
    out 43h, al
    mov ax, 1193
    out 42h, al
    mov al, ah
    out 42h, al
    in al, 61h
    or al, 3
    out 61h, al
    mov ecx, 0FFFFh
        _delay :
    loop _delay
    in al, 61h
    and al, 0FCh
    out 61h, al

```

```

    }
}

void color_print(const char* text, int color) {
    HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE);
    switch (color) {
    case COLOR_RED: SetConsoleTextAttribute(console,
        FOREGROUND_RED); break;
    case COLOR_GREEN: SetConsoleTextAttribute(console,
        FOREGROUND_GREEN); break;
    default: SetConsoleTextAttribute(console,
        FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE);
    }
    puts(text);
    SetConsoleTextAttribute(console,
        FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE);
}

void show_splash() {
    clrscr_asm();
    printf("=====\n");
    printf("          КУРСОВАЯ РАБОТА\n");
    printf("    по дисциплине \"Программирование\"\n");
    printf("на тему: \"База данных Телефонная книга\"\n");
    printf("=====\n\n");
    printf("Выполнил: студент группы 25BBB2 Доблер Б. С.\n");
    printf("Приняли: к.т.н. Генералова А. А.\n\n");
    color_print("Нажмите любую клавишу для продолжения...",
        COLOR_GREEN);
    GETCHAR_HANDLER = getchar();
}

```

```

voidshow_main_menu() {
printf("\n===== ГЛАВНОЕ МЕНЮ =====\n");
printf("1) Добавить контакт\n");
printf("2) Показать все контакты\n");
printf("3) Поиск и фильтрация\n");
printf("4) Сортировка\n");
printf("5) Удалить контакт\n");
printf("6) Редактировать контакт\n");
printf("7) Сохранить в файл\n");
printf("8) Загрузить из файла\n");
printf("0) Выход\n");
printf("Ваш выбор: ");
}

Contact* add_contact_dialog(Contact* head) {
clrscr_asm();

char name[LEN_NAME], phone[LEN_PHONE], email[LEN_EMAIL],
group[LEN_GROUP];

intisFav;

printf("Введите имя: ");
fgets(name, LEN_NAME, stdin);
crop_string(name);

printf("Введите телефон: ");
fgets(phone, LEN_PHONE, stdin);
crop_string(phone);

printf("Введите email: ");
fgets(email, LEN_EMAIL, stdin);
crop_string(email);

```

```

printf("Введите группу: ");
fgets(group, LEN_GROUP, stdin);
crop_string(group);

printf("Избранный? (1 - Да, 0 - Нет): ");
scanf_s("%d", &isFav);
    GETCHAR_HANDLER = getchar();

// Поиск максимального ID
intmax_id = 0;
Contact* curr = head;
while (curr) {
    if (curr->id > max_id) max_id = curr->id;
    curr = curr->next;
}

// Пытаемся перезаписать удалённую запись
curr = head;
bool replaced = false;
while (curr) {
    if (curr->id == -1) {
        curr->id = max_id + 1;
        strcpy_s(curr->name, LEN_NAME, name);
        strcpy_s(curr->phone, LEN_PHONE, phone);
        strcpy_s(curr->email, LEN_EMAIL, email);
        strcpy_s(curr->group, LEN_GROUP, group);
        curr->isFavorite = isFav;
        replaced = true;
        break;
    }
}

```

```

curr = curr->next;

    }

if (!replaced) {
if (!head)
head = init_contact(max_id + 1, name, phone, email, group,
isFav);
else
append_contact(head, max_id + 1, name, phone, email, group,
isFav);

    }

color_print(SUCCESS, COLOR_GREEN);

    GETCHAR_HANDLER = getchar();
return head;
}

voidedit_contact_dialog(Contact* head) {
clrscr_asm();
if (!head) {
color_print(EMPTY_LIST, COLOR_RED);

    GETCHAR_HANDLER = getchar();
return;

    }

int id;

printf("Введите ID контакта для редактирования: ");
scanf_s("%d", &id);

    GETCHAR_HANDLER = getchar();

    Contact* contact = find_contact_by_id(head, id);

```

```

if (!contact) {
beep_asm();    // <-- АССЕМБЛЕР: звукошибки
color_print(NOT_FOUND, COLOR_RED);

    GETCHAR_HANDLER = getchar();
return;
    }

printf("Введите новое имя (было: %s): ", contact->name);
fgets(contact->name, LEN_NAME, stdin);
crop_string(contact->name);

printf("Введите новый телефон (было: %s): ", contact->phone);
fgets(contact->phone, LEN_PHONE, stdin);
crop_string(contact->phone);

printf("Введите новый email (было: %s): ", contact->email);
fgets(contact->email, LEN_EMAIL, stdin);
crop_string(contact->email);

printf("Введите новую группу (было: %s): ", contact->group);
fgets(contact->group, LEN_GROUP, stdin);
crop_string(contact->group);

printf("Избранный? (1 - Да, 0 - Нет, было: %d): ", contact-
->isFavorite);
int isFav;
scanf_s("%d", &isFav);

    GETCHAR_HANDLER = getchar();
contact->isFavorite = isFav;

color_print(SUCCESS, COLOR_GREEN);

```

```

    GETCHAR_HANDLER = getchar();
}

void print_all_contacts(Contact* head) {
    clrscr_asm();
    if (!head) {
        color_print(EMPTY_LIST, COLOR_RED);
        GETCHAR_HANDLER = getchar();
        return;
    }

    printf("-----\n");
    printf("| %-2s | %-20s | %-12s | %-26s | %-11s | %-9s |\n",
        "ID", "Имя", "Телефон", "Email", "Группа", "Избранное");
    printf("-----\n");

    Contact* curr = head;
    while (curr) {
        if (curr->id != -1) {
            printf("| %-2d | %-20s | %-12s | %-26s | %-11s | %-9s |\n",
                curr->id, curr->name, curr->phone, curr->email, curr->group,
                curr->isFavorite ? "Да" : "Нет");
        }
        curr = curr->next;
    }

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

    GETCHAR_HANDLER = getchar();
}

```

```

}

void search_menu(Contact* head) {
    clrscr_asm();
    if (!head) {
        color_print(EMPTY_LIST, COLOR_RED);
        GETCHAR_HANDLER = getchar();
    }
    return;
}

printf("==== ПОИСК ==== \n");
printf("1) Поиск по ID \n");
printf("2) Поиск по имени \n");
printf("3) Поиск по группе \n");
printf("4) Показать только избранные \n");
printf("Ваш выбор: ");

char choice;
choice = getchar();
    GETCHAR_HANDLER = getchar();

if (choice == '1') {
    int id;
    printf("Введите ID: ");
    scanf_s("%d", &id);
        GETCHAR_HANDLER = getchar();
        Contact* c = find_contact_by_id(head, id);
    clrscr_asm();
    if (c) {
        printf("Найден контакт: \n");
    }
}

```

```

printf("ID: %d\nИмя: %s\nТелефон: %s\nEmail: %s\nГруппа:
%s\nИзбранный: %s\n",

        c->id, c->name, c->phone, c->email, c->group,
c->isFavorite ? "Да" : "Нет");

    }

else {
beep_asm();
color_print(NOT_FOUND, COLOR_RED);

    }

    }

    GETCHAR_HANDLER = getchar();
}

void sort_menu(Contact** head) {
clrscr_asm();
if (!(*head)) {
color_print(EMPTY_LIST, COLOR_RED);

    GETCHAR_HANDLER = getchar();
return;

    }

int count = count_active_contacts(*head);

    Contact** arr = (Contact**)malloc(count * sizeof(Contact*));

    Contact* curr = *head;

int idx = 0;
while (curr) {
if (curr->id != -1)
arr[idx++] = curr;
curr = curr->next;
}

```

```

printf("==== СОРТИРОВКА ====\n");
printf("1) По имени (А-Я)\n");
printf("2) По ID (убывание)\n");
printf("Вашвыбор: ");

char choice = getchar();

    GETCHAR_HANDLER = getchar();

if (choice == '1')
qsort(arr, count, sizeof(Contact*), compare_by_name);
else if (choice == '2')
qsort(arr, count, sizeof(Contact*), compare_by_id_desc);

clrscr_asm();

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

printf("| %-2s | %-20s | %-12s | %-26s | %-11s | %-9s |\n",
"ID", "Имя", "Телефон", "Email", "Группа", "Избранное");

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

for (inti = 0; i< count; i++) {
printf("| %-2d | %-20s | %-12s | %-26s | %-11s | %-9s |\n",
arr[i]->id, arr[i]->name, arr[i]->phone, arr[i]->email,
arr[i]->group, arr[i]->isFavorite ? "Да" : "Нет");
    }

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

free(arr);

    GETCHAR_HANDLER = getchar();
}

```

Файл ui.h

```
#ifndef UI_H
#define UI_H

#include "phonebook.h"

voidclrscr_asm();
voidbeep_asm();
voidcolor_print(const char* text, int color);
voidshow_splash();
voidshow_main_menu();

Contact* add_contact_dialog(Contact* head);
voidedit_contact_dialog(Contact* head);
voidprint_all_contacts(Contact* head);
voidsearch_menu(Contact* head);
voidsort_menu(Contact** head);

#endif
```

Файл utils.c

```
#include <string.h>
#include "utils.h"
#include "phonebook.h"

voidcrop_string(char* str) {
    __asm {
        movedi, str
        movecx, -1
        xoreax, eax
        repnescasb
        notecx
    }
```

```

dececx
cmpecx, 0
j1 done

movedi, str
addedi, ecx
decedi

cmp byte ptr[edi], 10
jnecheck_r
mov byte ptr[edi], 0
decedi

check_r :
cmpedi, str
j1 done
cmp byte ptr[edi], 13
jne done
mov byte ptr[edi], 0

done :
    }
}

intasm_strncmp(const char* s1, const char* s2) {
int result;
    __asm {
movesi, s1
movedi, s2
        _loop :
mov al, [esi]

```

```

movbl, [edi]
cmp al, bl
jne _not_equal
test al, al
jz _equal
incesi
incedi
jmp _loop

        _not_equal :
movsxeax, al
movsxebx, bl
subeax, ebx
mov result, eax
jmp _end

        _equal :
xoreax, eax
mov result, eax

        _end :
}
return result;
}

```

```

intcompare_by_name(const void* a, const void* b) {
    Contact* ca = *(Contact**)a;
    Contact* cb = *(Contact**)b;
    return strcmp(ca->name, cb->name);
}

```

```

intcompare_by_id_desc(const void* a, const void* b) {
    Contact* ca = *(Contact**)a;
    Contact* cb = *(Contact**)b;

```

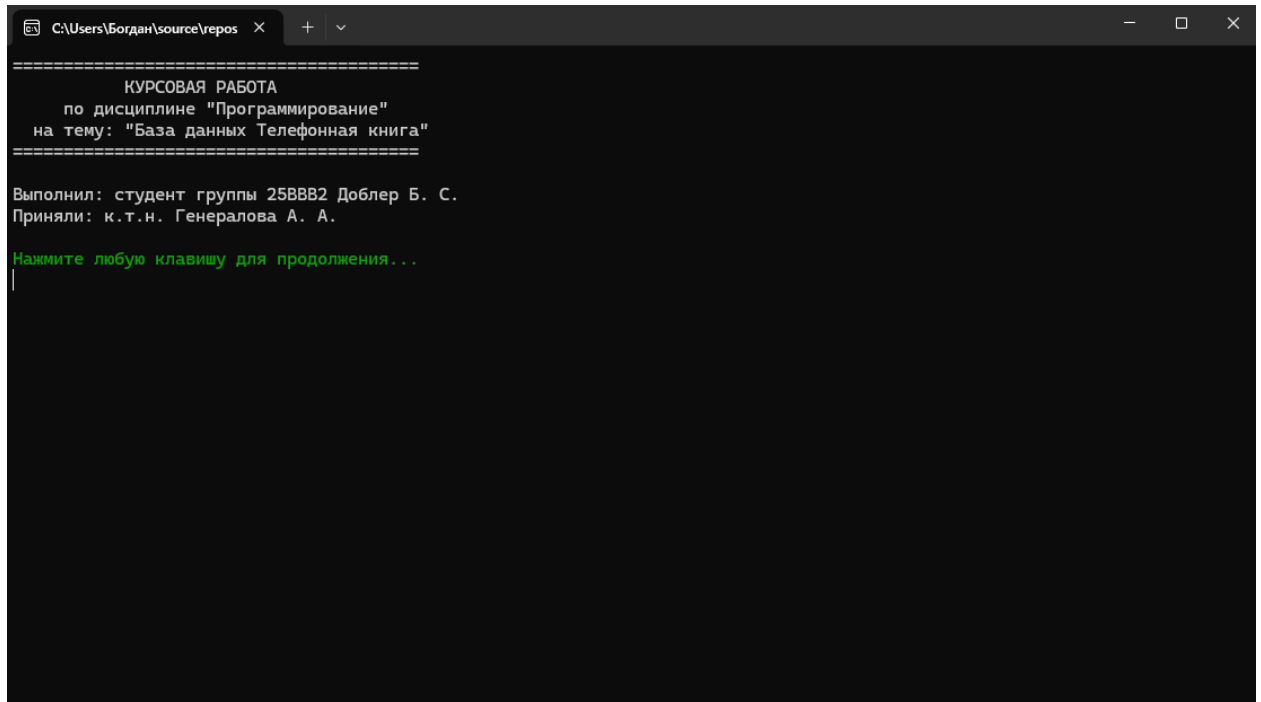
```
return cb->id - ca->id;  
}
```

Файл `utils.h`

```
#ifndef UTILS_H  
#define UTILS_H  
  
void crop_string(char* str);  
int asm_strcmp(const char* s1, const char* s2);  
int compare_by_name(const void* a, const void* b);  
int compare_by_id_desc(const void* a, const void* b);  
  
#endif
```

Приложение В Результаты работы программы

В.1 – Информационная заставка (рисунок 26)



```
C:\Users\Богдан\source\repos
=====
      КУРСОВАЯ РАБОТА
по дисциплине "Программирование"
на тему: "База данных Телефонная книга"
=====
Выполнил: студент группы 25BVB2 Доблер Б. С.
Приняли: к.т.н. Генералова А. А.
Нажмите любую клавишу для продолжения...
|
```

Рисунок 26 – Информационная заставка

Приложение С Результат работы программы

Файл test.txt

-ST@RT-

1

Богдан

89674454339

bogdan2007@gmail.com

25BBB2

1

-CONTINUE-

2

Екатерина

8777546365

ekatz22@gmail/com

21ИА2

0

-CONTINUE-

3

Федор

87453614255

fedor0909@email.com

23ВИС1

0

-CONTINUE-