

Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

по курсу «Программирование»

на тему «Разработка программы сложной структуры
методом нисходящего программирования»

809-01.14
К

Выполнил

студент группы 22BOA2

Губин Б.В.

Принял

Митрохин М. А.

Пенза 2023

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет Вычислительной техники

Кафедра «Вычислительная техника»

«УТВЕРЖДАЮ»

Зав. кафедрой ВТ

Митрохин М. А.

« »

2023 г.

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Губину Борису Владимировичу

Группа 22ВОА2

Тема проекта: База данных/Составление односвязного списка с использованием сортировки. Разработка программы сложной структуры методом нисходящего программирования.

Исходные данные (технические требования) на проектирование

Наименование программы: База данных «Контактная книжка».

База данных включает следующие поля:

Имя, фамилия, телефонный номер, электронный адрес.

Задание. Программа должна выполнять следующие функции:

- Добавление/удаление записей, их редактирование.
- Поиск и сортировка записей.
- Сохранение базы данных в файл и удаление этого файла.
- Загрузка ранее сохраненной базы данных из файла.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных — списков, структур, 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 — Windows.
8. Система программирования — C/C++ на базе среды Microsoft Visual Studio 2022.
9. Язык программирования — Си.

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Структура модулей программы, схема алгоритма программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

1. В соответствии с графиком выполнения курсового проекта.

2. _____

3. _____

Дата выдачи задания « 1 » _____ октября 202

Дата защиты проекта « _ » _____ 202

Руководитель _____ « 1 » октября 2023

Задание получил _____ « 1 » октября 2023

Студент Губин Борис Владимирович

Содержание

Введение	
1. Постановка задачи.....	4
2. Выбор решения.....	5
• Определение необходимых модулей.....	6
• Определение структуры файла базы данных.....	7
3. Описание разработки программы.....	8
4. Отладка и тестирование.....	9
5. Описание программы.....	10
• Разработка модуля main.c.....	11
• Разработка меню.....	12
• Разработка функции на языке Assembler.....	13
6. Руководство пользователя.....	14
7. Список используемых источников.....	15
8. Приложение А. Листинги программы.....	16
9. Приложение Б. Снимки экрана.....	17

Введение

Базы данных — это совокупность структур, предназначенных для хранения больших объемов информации, и программных модулей, осуществляющих управление данными, выборку, сортировку и другие подобные действия над записями. Создавая базу данных, пользователь стремится упорядочить информацию по различным признакам и быстро извлекать данные, при необходимости делая выборку с произвольным сочетанием признаков.

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

Одна таблица уже по сути является базой данных. Однако в реальной жизни структуры баз данных, а соответственно и способы их создания, намного обширнее и сложнее.

С появлением и расширением компьютерных технологий появились противоречия между ограниченными возможностями человека по восприятию и переработке информации и существующими массивами хранящейся и передаваемой информации. Для решения подобных проблем применяются автоматизированные базы данных. Они стали неотъемлемой частью практически всех компьютерных систем — от глобальной отрасли до отдельного предприятия. Для персональных компьютеров было разработано множество различных программ, предназначенных для решения различных потребностей человека. Данная программа является информационно-поисковой системой, позволяющей любому пользователю обратиться к данным о студентах и произвести необходимые действия над записями — от сортировки по алфавиту до поиска по группе.

1 Постановка задачи и обязательные требования

Необходимо разработать программу «База данных/Односвязный список с использованием сортировки», которая должна отвечать следующим требованиям:

1. Многомодульность. Программа должна быть поделена на логические модули. Это упростит дальнейшее совершенствование программы, а также позволит локализовать ошибки.

2. Использование сложных типов данных: массивов, структур, объектов и т.д. Для разработки данной программы необходимо подробно изучить принципы работы с массивами.

3. Режим работы видеосистемы: текстовый/графический. Необходимо определиться с типом управления, видом интерфейса и изучить приёмы реализации графики

4. Устройство ввода информации: клавиатура/мышь. Необходимо определиться с механикой взаимодействия игрока с игрой и её интерфейсом.

5. Пользовательский интерфейс должен быть построен на основе меню или панели инструментов. Необходимо реализовать пользовательское меню, с заставкой.

6. Операционная система: MS Windows. При разработке использовать среду MS Windows с системой программирования MS Visual Studio.

7. Язык программирования: Си, а также некоторые функции из языка C++ для работы с графикой.

2.1 Выбор решения

- Определение необходимых модулей программы

Разработанная программа состоит из трех модулей:

- List.h
- main.c

Заголовочный файл list.h необходим для объявления глобальных переменных и подключения стандартных библиотек языка C. Данная библиотека была создана специально для данной курсовой работы.

Модуль main.c – основной модуль данной курсовой работы. Модуль содержит все требуемые функции программы: добавление записей в БД, удаление записей из БД, редактирование записей БД, поиск и сортировка записей БД, фильтрация записей БД по определенному критерию, создание новой БД, сохранение текущей БД, загрузка сохраненных БД.

Все модули с расширением .c используют заголовочные файлы (main.c еще глобальную переменную) из библиотеки list.h. Тем самым выполняется требование модульности данной курсовой работы. Управление приложением осуществляется с помощью клавиатуры. Одновременная работа двух программ с одной базой данных не допускается. Данные из БД отображаются в программе в виде таблицы из трёх столбцов: Фамилия, Имя, Номер телефона. Навигация по таблице организована через события клавиатуры, а именно через клавиши – цифры от 1 до 5 (с использованием клавиши 0 включительно).

2.2 Определение структуры файла базы данных

Для структурирования данных внутри файла была выбрана следующая структура данных:

```
typedef struct Node {  
char name[50];    //имя пользователя  
char surname[50]; //фамилия пользователя  
char phoneNumber[20]; // телефонный номер пользователя  
char email[100];  // электронный адрес пользователя
```

Поля структуры «Имя», «Фамилия», «Номер телефона», «Электронный адрес» имеют символьный формат

База данных реализована в виде массива структур. Количество записей в файле неограниченно.

3 Описание разработки программы

В написании данной программы использован достаточно распространенный язык программирования C. При разработке языка был принят компромисс между низким уровнем языка ассемблера и высоким уровнем других языков. Язык C даёт программисту возможность выражать свои алгоритмы и данные наиболее подходящими для этого средствами. Язык обеспечивает возможности структурирования данных, позволяет разрабатывать большие структурированные программы, предоставляя возможность отдельной разработки подпрограмм.

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

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

Основная программа выполняется функцией `main()`. Функция отвечает за инициализацию консоли, вывод меню на экран, передачу управления в функции обработки событий меню. После ввода некоторого значения с клавиатуры функция обработки событий меню получает идентификатор выбранного пункта меню, в соответствии с которым организуется передача управления в нужную функцию. Блок-схема функции `main()` и функции обработки событий меню представлена на рисунке 1.



Рисунок 1– схема программы, функция обработки событий меню

4 Отладка и тестирование

Средой разработки была выбрана программа Visual Studio 2019. Программа содержит в себе все необходимые средства для успешной отладки и тестирования модулей программы, а также всей программы в целом. Данная среда разработки выпущена относительно недавно, что позволяет проводить необходимые операции над разработанной программой наиболее производительно и эффективно. Для выполнения отладки применялись средства Visual Studio, такие как: точки останова, анализ содержимого переменных, выполнение кода по шагам (по строкам), анализ памяти.

Тестирование проводилось в рабочем порядке, на всех этапах разработки программы и написания кода. Тестирование позволило выявить множество проблем и недочетов по части внешнего вида таблицы, соответствия пунктов меню функциям и пр. Для эффективной работы программы были созданы проверки, не позволяющие проводить обработку некоторых частей программы в определенных условиях.

5.1 Описание программы

- Разработка модуля main.c

Модуль main.c является основным модулем программы. Вспомогательные модули (list.c, ui.c, ui.h) содержат в себе все функции, используемые в main.c. В модуле происходит полный процесс работы с БД, начиная от ее создания и вывода на экран и заканчивая сохранением готовой БД. При запуске программы на экран выводится меню, содержащее шесть функций, готовых выполняться по нажатию соответствующей клавиши (1,2,3,4,5,0). После нажатия одной из шести клавиш управление переходит в функции, реализованные в основном модуле (файле) main.c, меню соответственно исчезает, происходит обработка выбранного пункта.

Клавиша, вызывающая событие	Действие пользователя	Действие программы
-----------------------------	-----------------------	--------------------

Описание всех состояний программы приведено в таблице 1:

Таблица 1. Описание состояний программы

1	Выбран пункт «Add Contact»	Пользователю предлагается заполнить поля «Имя», «Фамилия», «Номер», «Электронный адрес».
2	Выбран пункт «Output a sorted list»	Пользователю предлагается вывести отсортированный по критерию список.
3	Выбран пункт «Output an unsorted list»	Осуществляется вывод не отсортированного списка.
4	Выбран пункт «Download the list from a file»	Осуществляется загрузка контактов из файла.
5	Выбран пункт «Save the list to a file»	Осуществляется сохранение контактов в файл.
6	Выбран пункт «Delete a contact by last name»	Пользователю предлагается удалить контакт по фамилии.
7	Выбран пункт «Exit»	Осуществляется выход из программы.

5.2 Разработка меню

Меню реализовано в файле main.c. Состоит из следующих пунктов:

- «Добавление контактов»
- «Удаление контактов»
- «Вывод контактов»
- «Сохранение контактов в файл»
- «Загрузка контактов из файла»

Главной задачей функций файла main.c является обеспечение возможности выбора пользователю одного из пунктов меню и возврата программе идентификатора выбранного пункта, в соответствии с которым происходит передача управления в нужную часть программы.

Основное меню реагирует на нажатие клавиш: цифры от 0 до 7.

Отладка программы

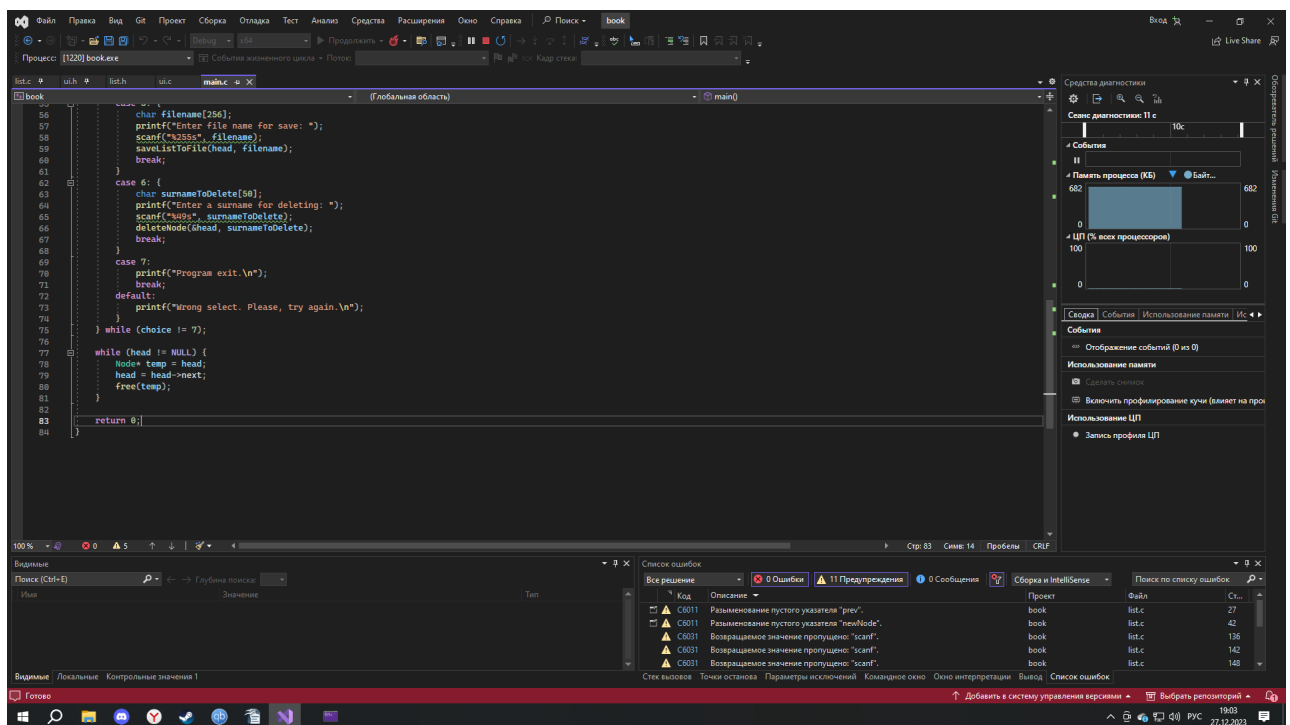


Рисунок 2 — отладка программы

6 Описание структуры программы

Файл «main.c» — главный файл. В нём объявляются основные переменные и массивы программы. Функция «main» – главная функция, в которой вызываются функции, которые используются для работы программы. Структура программы показана на рисунке 2.



Рисунок 3 – структура программы

Исходный файл «*main.c*» - главный файл программы. Включает в себя

заголовочный файл «*list.c*» а так же исходный файл «*list.h*» который содержит в себе функцию «*Node* head = NULL;*», с помощью которой пользователь вводит в поля данные — Имя/Фамилию/Номер/Электронный адрес пользователя. Размерность данных регулируются с помощью функций-

```
((char*)malloc(sizeof(char*)));». Заголовочный файл котором осуществляется
поиск контактов по номеру для его удаления с помощью цикла if (temp !=
NULL && strcmp(temp->surname, surnameToDelete) == 0) {
*head = temp->next;
free(temp);
return;
```

, в противном случае нужный контакт будет не найден и пользователю высветится сообщение об ошибке. *void insertSorted(Node** head, Node* newNode, int (*compare)(const Node*, const Node*))* {

```
if (*head == NULL || compare(newNode, *head) <= 0) {
    newNode->next = *head;
    *head = newNode;
```

Заголовочный файл «*ui.c*» и исходный файл «*ui.h*» служат для регулирования полей, в которых пользователь заносит данные о контактах-

// Шапка таблицы

```
printf("====*====*\n");
printf("| № | Name | Surname | Phone Number | E-mail | \n");
printf("====*====*\n");
```

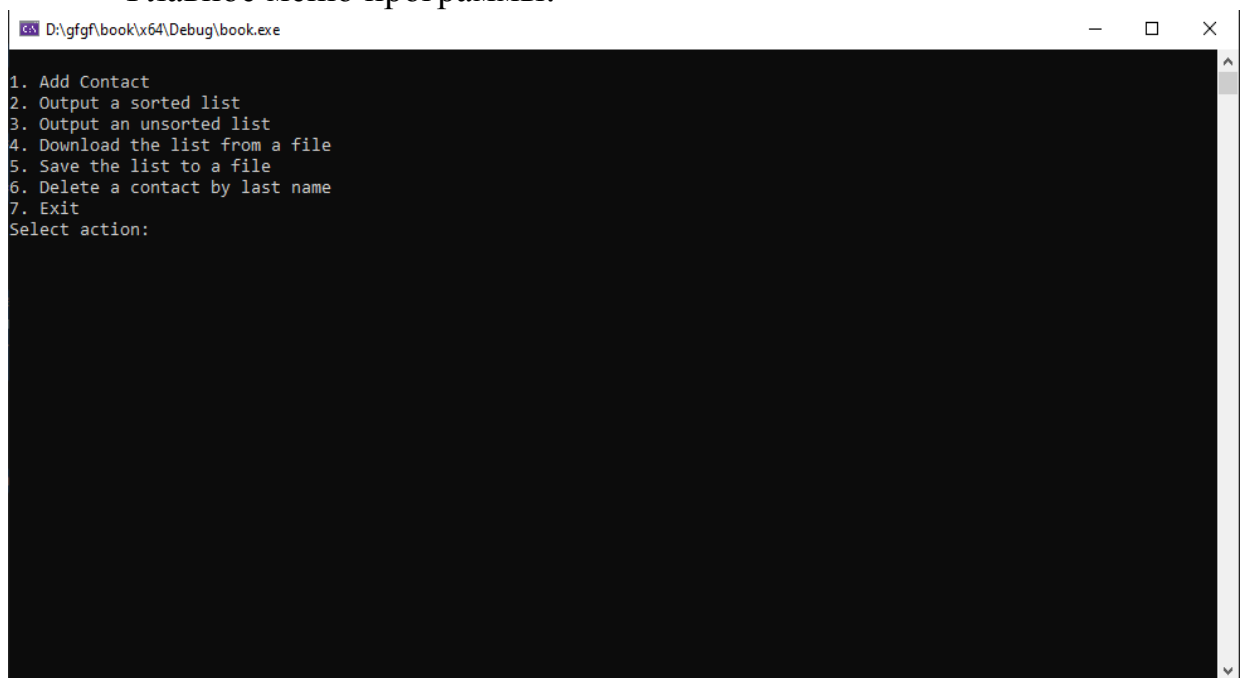
Заголовочный файл «*ui.h*» и исходный файл «*ui.c*» служат для поиска, открытия и хранения файла. В случае ошибки пользователю высветится сообщение:-

```
fprintf(stderr, "Can not open file for reading\n");
printf("Memory allocation error.\n");
fprintf(stderr, "Can not open file for record\n");
printf("Wrong select. Please, try again.\n");
```

7 Руководство пользователя

Программа предназначена для хранения и занесения контактов в БД. Программа имеет интуитивно понятный интерфейс и содержит в себе такие функции, как «Добавление контактов», «Удаление контактов», «Вывод контактов», «Сохранение контактов в файл», «Загрузка контактов из файла», «Выход из программы».

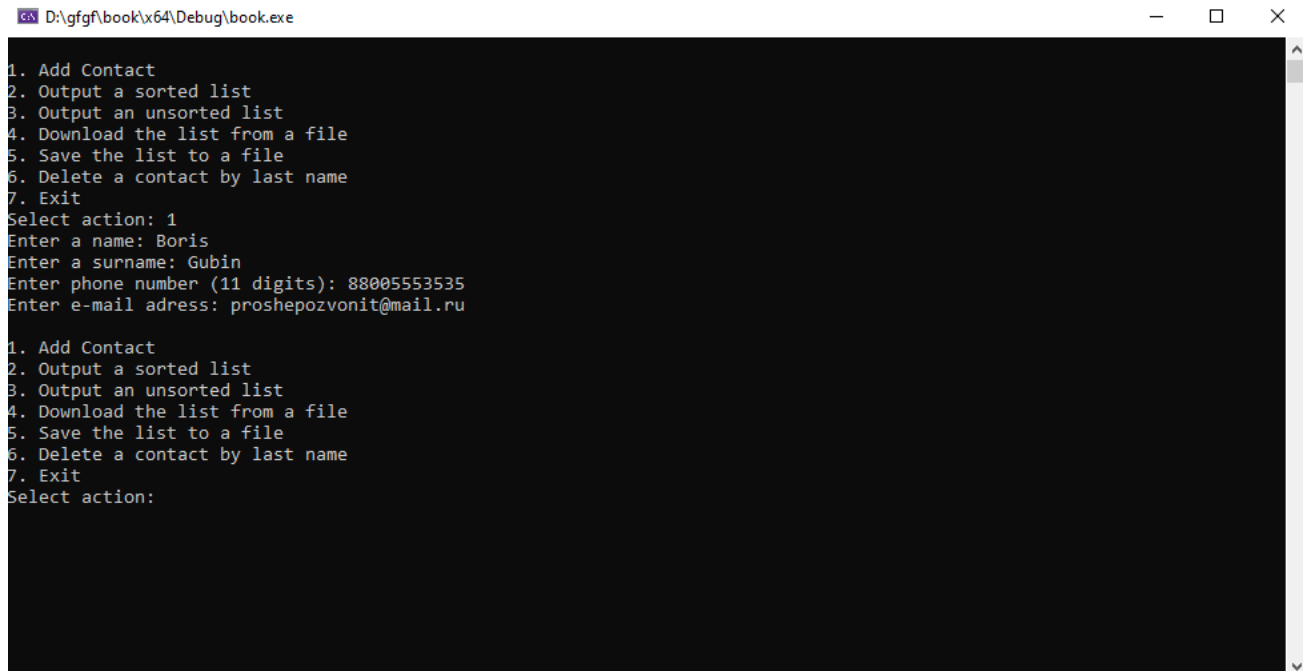
Главное меню программы:



8.

Рисунок 4 – главное меню программы

Для добавления контакта пользователю необходимо нажать ввести цифру 1 – «добавление контактов»:



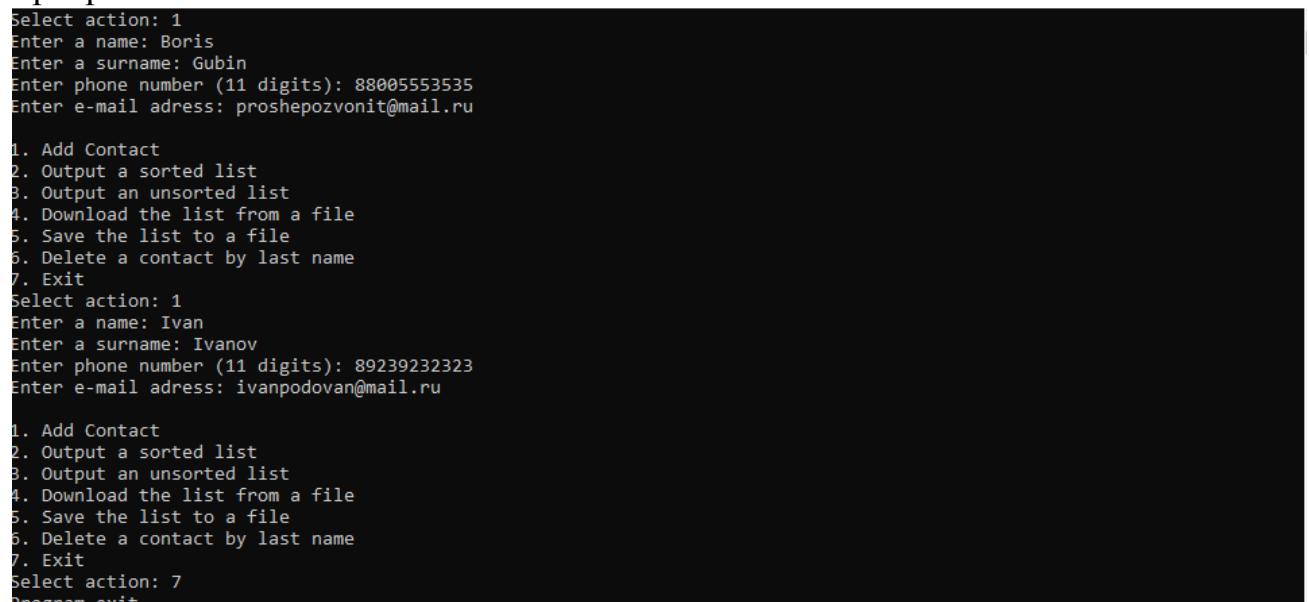
```
D:\gfgf\book\x64\Debug\book.exe

1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 1
Enter a name: Boris
Enter a surname: Gubin
Enter phone number (11 digits): 88005553535
Enter e-mail address: proshepozvonit@mail.ru

1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action:
```

Рисунок 5 – добавление контакта

Для добавления еще одного контакта пользователю снова нужно ввести цифру 1, а затем нажать цифру 0 для возвращения в главное меню программы:



```
D:\gfgf\book\x64\Debug\book.exe

1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 1
Enter a name: Boris
Enter a surname: Gubin
Enter phone number (11 digits): 88005553535
Enter e-mail address: proshepozvonit@mail.ru

1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 1
Enter a name: Ivan
Enter a surname: Ivanov
Enter phone number (11 digits): 89239232323
Enter e-mail address: ivanpodovan@mail.ru

1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 7
program exit
```

Рисунок 6 – создание второго контакта и выход в главное меню программы

Для вывода контактов пользователь должен ввести цифру 3 -

« Вывод контактов»:

```
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 3
*-----*-----*-----*-----*-----*-----*
| № | Name | Surname | Phone Number | E-mail |
|-----*-----*-----*-----*-----*-----*
| 1 | Ivanov | Ivan | 89283435456 | ivanpodovan@gmail.com |
|-----*-----*-----*-----*-----*-----*
| 2 | Boris | Gubin | 88005553535 | proshepozvonit@mail.ru |
|-----*-----*-----*-----*-----*-----*
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
```

Рисунок 7– вывод контактов

Для удаления из списков контактов, пользователь должен ввести цифру 2, а затем ввести порядковый номер контакта, чей намерен удалить:

```
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 6
Enter a surname for deleting: Ivanov
```

Рисунок 8– меню удаления контакта

Для проверки пользователь должен ввести цифру 3 и проверить, удалился ли нужный контакт:

```
Select action: 3
*-----*-----*-----*-----*-----*-----*
| № | Name | Surname | Phone Number | E-mail |
|-----*-----*-----*-----*-----*-----*
| 1 | Boris | Gubin | 88005553535 | proshepozvonit@mail.ru |
|-----*-----*-----*-----*-----*-----*
```

Рисунок 9 – проверка функции «удаление контакта»

Для сохранения контактов в файл пользователю необходимо ввести цифру 5 - «Сохранение контактов в файл»:

```
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 5
Enter file name for save: open.txt
```

Рисунок 10 – сохранение контактов в файл

Для загрузки контактов из файла пользователю нужно ввести цифру 4 - «Загрузка контактов из файла»:

```
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 4
Enter file name for download: open.txt
```

Рисунок 11 – загрузка контактов

Для выхода из программы пользователю нужно ввести цифру 7 - «Выход из программы»:

```
1. Add Contact
2. Output a sorted list
3. Output an unsorted list
4. Download the list from a file
5. Save the list to a file
6. Delete a contact by last name
7. Exit
Select action: 7
Program exit.

D:\gfgf\book\x64\Debug\book.exe (процесс 11192) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "А
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 12– выход из программы

Заключение

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

В данной курсовой работе была написана программа, используемая для хранения информации о контактах, «база данных контактов». Программа представляет ограниченный функционал, содержит базовые средства для выполнения операций в базе данных.

Программу в будущем можно улучшать, добавлять новые функции в области поиска и фильтрации, сортировки записей в базе данных; расширять меню новыми функциями, оптимизировать и персонализировать программу под конкретного пользователя и т.д.

Список используемых источников

1. Керниган Б., Ричи Д. Язык С. 2020 г. (дата обращения 15.10.23)
2. Земсков Ю.В. Программирование на языке С/С++. 2012 г.(дата обращения 15.10.23)
3. MSDN (дата обращения 15.10.23)
4. Mort N. Programming in C. 2020 г. (дата обращения 15.10.23)
5. Давыдов В.Г. Программирование и основы алгоритмизации. 2003 г. (дата обращения 15.10.23)

Приложение А. Листинг программы

Приложение А.1 Файл «main.c»

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "ui.h"
#include "list.h"
#include <locale.h>
int main() {
    Node* head = NULL;
    int choice;

    do {
        printf("\n1. Add Contact\n2. Output a sorted list\n3. Output an unsorted list\n4. Download the
list from a file\n5. Save the list to a file\n6. Delete a contact by last name\n7. Exit\n");
        printf("Select action: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addContact(&head);
                break;
            case 2: {
                int sortChoice;
                printf("\nSorting criteria:\n1. By name\n2. By surname\n3. By phone number\n4. By e-mail\
n");
                printf("Select sorting criteria: ");
                scanf("%d", &sortChoice);
                switch (sortChoice) {
                    case 1:
                        sortList(&head, compareByName);
                        break;
                    case 2:
                        sortList(&head, compareBySurname);
                        break;
                    case 3:
                        sortList(&head, compareByPhoneNumber);
                        break;
                    case 4:
                        sortList(&head, compareByEmail);
                        break;
                    default:
                        printf("Unccorrtect option!\n");
                }
                printList(head);
                break;
            }
            case 3:
                printList(head);
                break;
            case 4: {
                char filename[256];
                printf("Enter file name for download: ");
                scanf("%255s", filename);
                loadListFromFile(&head, filename);
                break;
            }
            case 5: {
                char filename[256];
                printf("Enter file name for save: ");
                scanf("%255s", filename);
```

```

        saveListToFile(head, filename);
        break;
    }
    case 6: {
        char surnameToDelete[50];
        printf("Enter a surname for deleting: ");
        scanf("%49s", surnameToDelete);
        deleteNode(&head, surnameToDelete);
        break;
    }
    case 7:
        printf("Program exit.\n");
        break;
    default:
        printf("Wrong select. Please, try again.\n");
    }
} while (choice != 7);

while (head != NULL) {
    Node* temp = head;
    head = head->next;
    free(temp);
}

return 0;
}

```

Приложение A.2 Файл «list.c»

```

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

void deleteNode(Node** head, const char* surnameToDelete) {
    Node* temp = *head, *prev = NULL;

    if (temp != NULL && strcmp(temp->surname, surnameToDelete) == 0) {
        *head = temp->next;
        free(temp);
        return;
    }

    while (temp != NULL && strcmp(temp->surname, surnameToDelete) != 0) {
        prev = temp;

```

```

    temp = temp->next;
}

if (temp == NULL) return;

prev->next = temp->next;
free(temp);
}

void loadListFromFile(Node** head, const char* filename) {
    FILE* file = fopen(filename, "r");
    if (!file) {
        fprintf(stderr, "Can not open file for reading\n");
        return;
    }

    char buffer[256];
    while (fgets(buffer, sizeof(buffer), file)) {
        Node* newNode = (Node*)malloc(sizeof(Node));
        if (sscanf(buffer, "%49[^\n],%49[^\n],%19[^\n],%99[^\n]",
            newNode->name, newNode->surname, newNode->phoneNumber, newNode->email) == 4) {
            newNode->next = *head;
            *head = newNode;
        }
    }

    fclose(file);
}

int compareByName(const Node* a, const Node* b) {
    return strcmp(a->name, b->name);
}

int compareBySurname(const Node* a, const Node* b) {
    return strcmp(a->surname, b->surname);
}

int compareByPhoneNumber(const Node* a, const Node* b) {
    return strcmp(a->phoneNumber, b->phoneNumber);
}

int compareByEmail(const Node* a, const Node* b) {
    return strcmp(a->email, b->email);
}

void insertSorted(Node** head, Node* newNode, int (*compare)(const Node*, const Node*)) {
    if (*head == NULL || compare(newNode, *head) <= 0) {
        newNode->next = *head;
        *head = newNode;
    }
    else {
        Node* current = *head;
        while (current->next != NULL && compare(newNode, current->next) > 0) {
            current = current->next;
        }
        newNode->next = current->next;
        current->next = newNode;
    }
}

void sortList(Node** head, int (*compare)(const Node*, const Node*)) {
    if (*head == NULL || (*head)->next == NULL) {
        return;
    }
    Node* sorted = NULL;
    Node* current = *head;
    Node* next = NULL;

    while (current != NULL) {
        next = current->next;

```

```

        insertSorted(&sorted, current, compare);
        current = next;
    }
    *head = sorted;
}

int contains_digit(const char* str) {
    while (*str) {
        if (isdigit((unsigned char)*str)) {
            return 1;
        }
        str++;
    }
    return 0;
}

int is_valid_phone_number(const char* str) {
    return strlen(str) == 11 && strspn(str, "0123456789") == 11;
}

int contains_one_at(const char* str) {
    int at_count = 0;
    while (*str) {
        if (*str == '@') {
            at_count++;
        }
        str++;
    }
    return at_count == 1;
}

void capitalize(char* str) {
    if (str && *str) {
        *str = toupper((unsigned char)*str);
    }
}

Node* createNode() {
    Node* newNode = (Node*)malloc(sizeof(Node));
    if (newNode == NULL) {
        printf("Memory allocation error.\n");
        exit(1);
    }

    do {
        printf("Enter a name: ");
        scanf("%49s", newNode->name);
        capitalize(newNode->name);
    } while (!contains_digit(newNode->name));

    do {
        printf("Enter a surname: ");
        scanf("%49s", newNode->surname);
        capitalize(newNode->surname);
    } while (!contains_digit(newNode->surname));

    do {
        printf("Enter phone number (11 digits): ");
        scanf("%19s", newNode->phoneNumber);
    } while (!is_valid_phone_number(newNode->phoneNumber));

    do {
        printf("Enter e-mail adress: ");
        scanf("%99s", newNode->email);
    } while (!contains_one_at(newNode->email));

    newNode->next = NULL;
    return newNode;
}

void addContact(Node** head) {

```

```

    Node* newNode = createNode();
    newNode->next = *head;
    *head = newNode;
}

void saveListToFile(Node* head, const char* filename) {
    FILE* file = fopen(filename, "w");
    if (file == NULL) {
        fprintf(stderr, "Can not open file for record\n");
        return;
    }

    Node* current = head;
    while (current != NULL) {
        // Записываем данные контакта в файл
        fprintf(file, "%s,%s,%s,%s\n",
            current->name,
            current->surname,
            current->phoneNumber,
            current->email);
        current = current->next;
    }

    fclose(file); // Закрываем файл
}

void addNode(Node** head) {
    Node* newNode = createNode();
    newNode->next = *head;
    *head = newNode;
}

```

Приложение A.3 Файл «list.h»

```

#ifndef LIST_H
#define LIST_H

typedef struct Node {
    char name[50];
    char surname[50];
    char phoneNumber[20];
    char email[100];
    struct Node* next;
} Node;

Node* createNode();
void addNode(Node** head);
void deleteNode(Node** head, const char* surnameToDelete);
void loadListFromFile(Node** head, const char* filename);
void saveListToFile(Node* head, const char* filename);
void sortList(Node** head, int (*compare)(const Node*, const Node*));
int compareByName(const Node* a, const Node* b);
int compareBySurname(const Node* a, const Node* b);
int compareByPhoneNumber(const Node* a, const Node* b);
int compareByEmail(const Node* a, const Node* b);
void addContact(Node** head);
#endif #pragma once

```

Приложение A.4 Файл «ui.h»

```

#ifndef UI_H
#define UI_H

#include "list.h"

```

```
void printList(Node* head);

#endif
```

Приложение A.5 Файл «ui.c»

```
#include "ui.h"
#include <locale.h>
#include <stdio.h>
void printList(Node* head) {
    setlocale(LC_ALL, "Russian");
    int index = 0;

    // Шапка таблицы

    printf("*****\n");
    printf("| № | Name | Surname | Phone Number | E-mail | \n");

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

    // Вывод данных
    Node* current = head;
    while (current != NULL) {
        printf("| %3d | %-12s | %-13s | %-20s | %-24s | \n",
            ++index, current->name, current->surname, current->phoneNumber, current->email);
        printf("*****\n");
        current = current->next;
    }

    // Если контактов нет
    if (index == 0) {
        fprintf(stderr, "No contacts\n");
    }
}
```

Приложение Б.

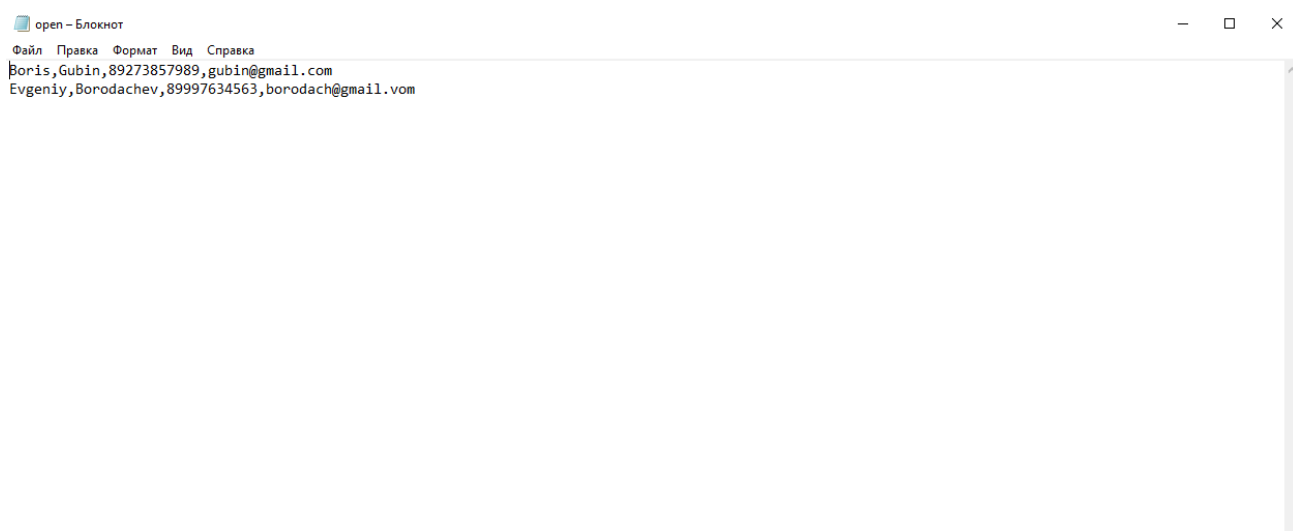


Рисунок 13. Содержимое файла базы данных