

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Программирование»

на тему «База данных по столицам стран»

Выполнил:

студент группы 21ВВ3:

Пронин Илья

Принял:

Голотенков Н.О.

Патунин Д.В.

10.06.22
кор

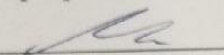
Пенза 2022

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра "Вычислительной техники"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ



М.А. Митрохин

« ____ » ____ 20__ г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Трошину Илье Геннадьевичу Группа 21ВВЗ

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

Исходные данные (технические требования) на проектирование

*Каширование программы: база данных по статистике страниц.
Функции программы: добавление записей, удаление записей,
редактирование записей, поиск и сортировка записей,
фильтрация записей по определ. признакам, 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, массивов, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню или 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, схема программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком выполнения курсового проекта.

Дата выдачи задания " 25 " феврале

Дата защиты проекта " 10 " 06.22

Руководитель Гатунин Д.В.

Задание получил " 25 " феврале 2022г.

Студент Грокин Илье Геннадьевич

Содержание

1. Введение	5
2. Основная часть	7
2.1. Постановка задачи	7
2.2. Выбор решения	8
2.2.1.Определение необходимых модулей программы	8
2.2.2.Определение структуры базы данных	9
2.3. Описание разработки программы	10
2.4. Отладка и тестирование	14
2.5. Описание программы	15
2.5.1.Разработка приложения	15
2.5.2.Разработка меню	16
2.5.3. Разработка Функции на языке Ассемблер	16
2.6. Руководство пользователя	17
3. Заключение	20
4. Список используемых источников	22
5. Приложение 1. Листинги программы	23
5.1. Приложение 1.1.	23
5.2. Приложение 1.2.	23
5.3. Приложение 1.3.	23
5.4. Приложение 1.4.	31
6. Приложение 2.	33
6.1. Приложение 2.1.	33

Введение

При создании средних по размеру приложений используется *структурное программирование*, идея которого заключается в том, что структура программы должна отражать структуру решаемой задачи, чтобы алгоритм решения был ясно виден из исходного текста. Для этого надо иметь средства для создания программы не только с помощью трех простых операторов, но и с помощью средств, более точно отражающих конкретную структуру алгоритма. С этой целью в программирование введено понятие *подпрограммы* — набора операторов, выполняющих нужное действие и не зависящих от других частей исходного кода. Программа разбивается на множество мелких подпрограмм (занимающих до 50 операторов — критический порог для быстрого понимания цели подпрограммы), каждая из которых выполняет одно из действий, предусмотренных исходным заданием. Комбинируя эти подпрограммы, удастся формировать итоговый алгоритм уже не из простых операторов, а из законченных блоков кода, имеющих определенную смысловую нагрузку, причем обращаться к таким блокам можно по названиям. Получается, что подпрограммы — это новые операторы или операции языка, определяемые программистом.

База данных - организованная в соответствии с определёнными правилами и поддерживаемая в памяти компьютера совокупность данных, характеризующая актуальное состояние некоторой предметной области и используемая для удовлетворения информационных потребностей пользователей.

База данных представляет собой упорядоченную информацию, объединенную в единое целое (например: список телефонных номеров, товаров, деталей, покупателей и т.д.).

Сам термин база данных появился в начале 1960-х годов, и был введен в употребление на симпозиумах, организованных

компанией SDC в 1964 и 1965 годах, хотя понимался сначала в довольно узком смысле, в контексте систем искусственного интеллекта. В широкое употребление в современном понимании термин вошёл лишь в 1970-е годы.

С понятием базы данных тесно связано понятие системы управления базами данных (СУБД). СУБД – это комплекс программных средств, предназначенных для создания структуры новой базы данных, наполнения ее содержимым, редактирования содержимого и создания различных пользовательских представлений информации (отбор, сортировка, фильтрация и т.д.).

Создавая базу данных, пользователь стремится упорядочить информацию по каким-либо признакам и быстро извлекать выборку с произвольным сочетанием признаков. А сделать это можно, только если данные структурированы.

Элементарная база данных представляется в виде двумерной таблицы.

В процессе выполнения курсовой работы была разработана программа на языке Си, позволяющая выполнять работу (поиск, сортировка, фильтрация, добавление и удаление строк и т.д.) с базой данных «Столицы мира».

Основная часть

1.1 Постановка задачи

Необходимо разработать программу «База данных – Столицы мира». Информация в базе данных структурирована по полям: столица, страна, площадь, численность. Для реализации данной программы необходимо определить структуру базы данных, размер, размер каждой записи базы данных.

Программа должна предоставить возможности: добавление новой записи, удаление записи; поиск, редактирование, сортировку и фильтрацию записей; сохранение базы данных в файл, загрузку базы данных из файла.

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

Необходимо использовать сложные типы данных – массивы, структуры, файлы. Необходимо определиться с элементами управления. Необходимо изучить приемы и механизмы псевдографического интерфейса. Это необходимо для понимания, как и почему на консоли появляются псевдографические символы линии и т.д.

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

Пользовательский интерфейс должен быть построен на основе меню. Необходимо изучить приемы программирования меню, выбрать подходящий вид меню. Это необходимо для создания интуитивно понятного интерфейса.

Многомодульность программы. Необходимо разделить программу на логические модули. Это повысит локализацию ошибок при отладке и тестировании программы.

1.2 Выбор решения

Определение необходимых модулей программы

Разработанная программа состоит из следующих модулей:

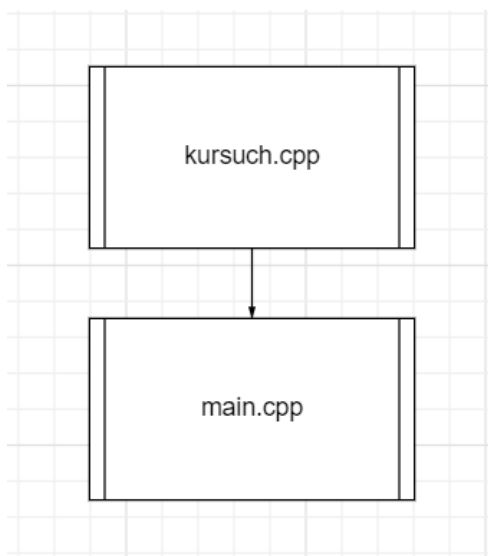


Рисунок 1 - Структура модулей программы

Файлы: «main.cpp», «bibl.h», «opening.cpp», «header.h», - необходимы для включения функций обработки данных. Данные файлы содержат функции создания, удаления, чтения, поиска, редактирования, сортировки, фильтрации, вывода на экран базы данных и функцию сохранения результата обработки базы данных.

Приложение «kursuch.exe» представляет собой совокупность данных модулей. Приложение использует данные, содержащиеся в файле «task1». Все вышеперечисленное удовлетворяет требованию модульности программы данной курсовой работы. Управление приложением осуществляется с помощью клавиатуры. При запуске программы выводится консоль для дальнейшего исполнения программы.

Определение структуры базы данных

Для структурирования данных внутри файла была выбрана следующая структура данных:

```
typedef struct tagITEM
{
    char Capital[20], Country[20];
    int Square, Population;
} ITEM;
```

Поля структуры: Столица, страна - имеют текстовый формат. Площадь, население – имеют целочисленный формат. Под столицу, страну выделено 20 символов. Выбранные структуры просты и соответствуют всем требованиям программы.

Ниже представлены схемы данных и ресурсов.

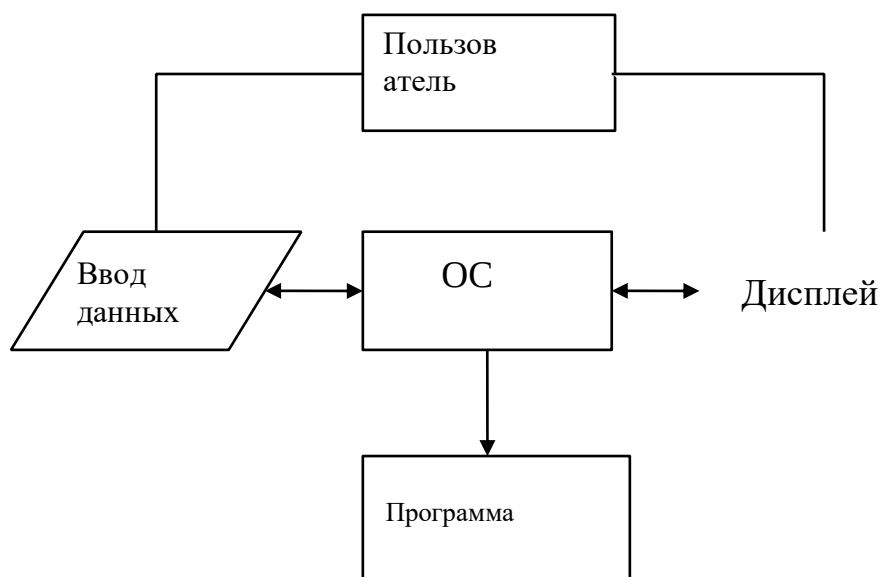


Рисунок 2- Схема данных

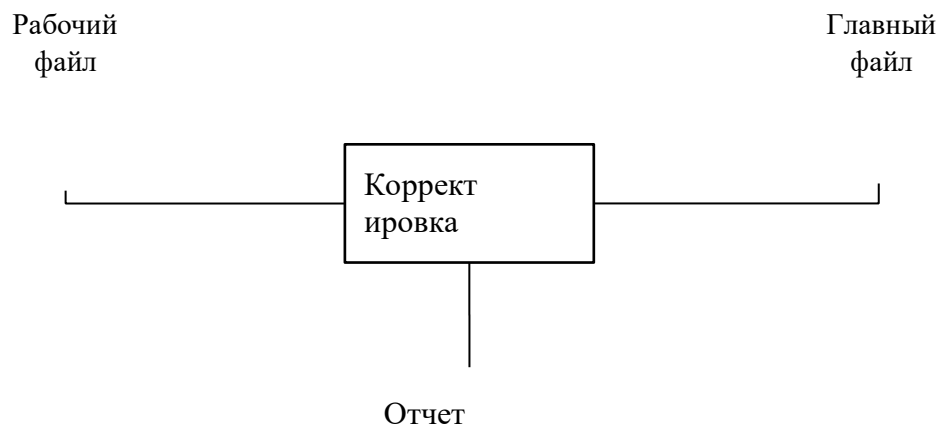


Рисунок 3. Схема ресурсов

2.2. Описание разработки программы

Для написания данной программы был использован язык программирования Си.

В качестве среды программирования был выбран программный продукт VisualStudio 2010.

Интерфейс информационной системы представлен в виде меню, т.к. он является наиболее удобным. Исполнение навигации программы по пунктам меню будет осуществляться с помощью клавиатуры.

Разработка программы началась с реализации набора функций, необходимых для работы с файлом базы данных. Используются:

void Save () - сохранение базы данных в файл:

void Sort() – упорядочивание массива по столицам;

void add () - добавление записи;

void Del () - удаление записи;

void Edit () - редактирование записи;

void Print () – вывод базы данных;

void Save () загрузка базы данных в файл;

void Filter () - фильтрация;

void Menu () – вывод меню;

void Search () – поиск по столице;

Основная программа выполняется функцией main(). В функции main реализован цикл обработки событий. Цикл обработки событий реагирует на события, вызванные клавиатурой, выполняет дешифрацию события и вызывает функции для обработки того или иного события. При активизации меню, управление переходит в функции реализации меню.

Был разработан алгоритм функции main. Блок-схема функции представлена на рисунке. Функция отвечает за инициализацию консоли и за передачу управления в функции обработки меню. Цикл обработки событий главного окна обеспечивает активизацию меню и выход из него. После выхода из функции обработки меню функция обработки событий главного окна получает идентификатор выбранного пункта меню. В зависимости от идентификатора функция вызывает функцию обработки выбранного пункта меню.

На рисунке 4 представлена схема обработки событий.

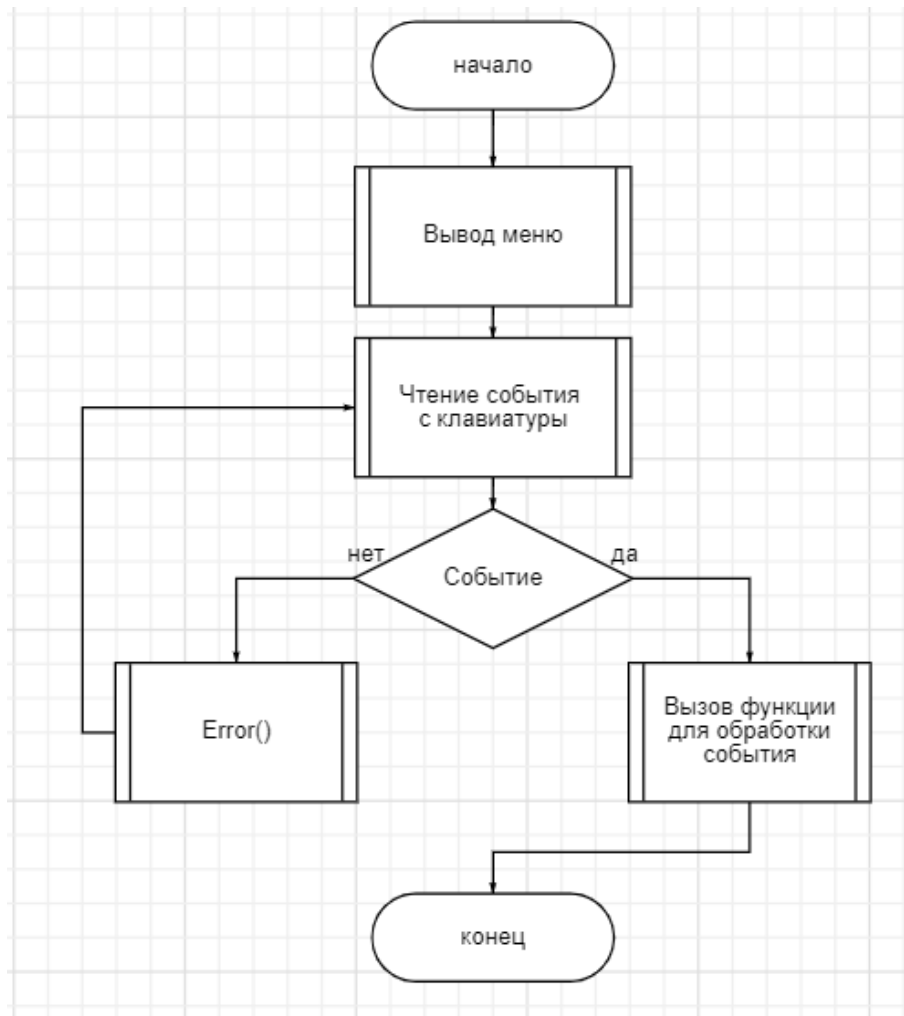
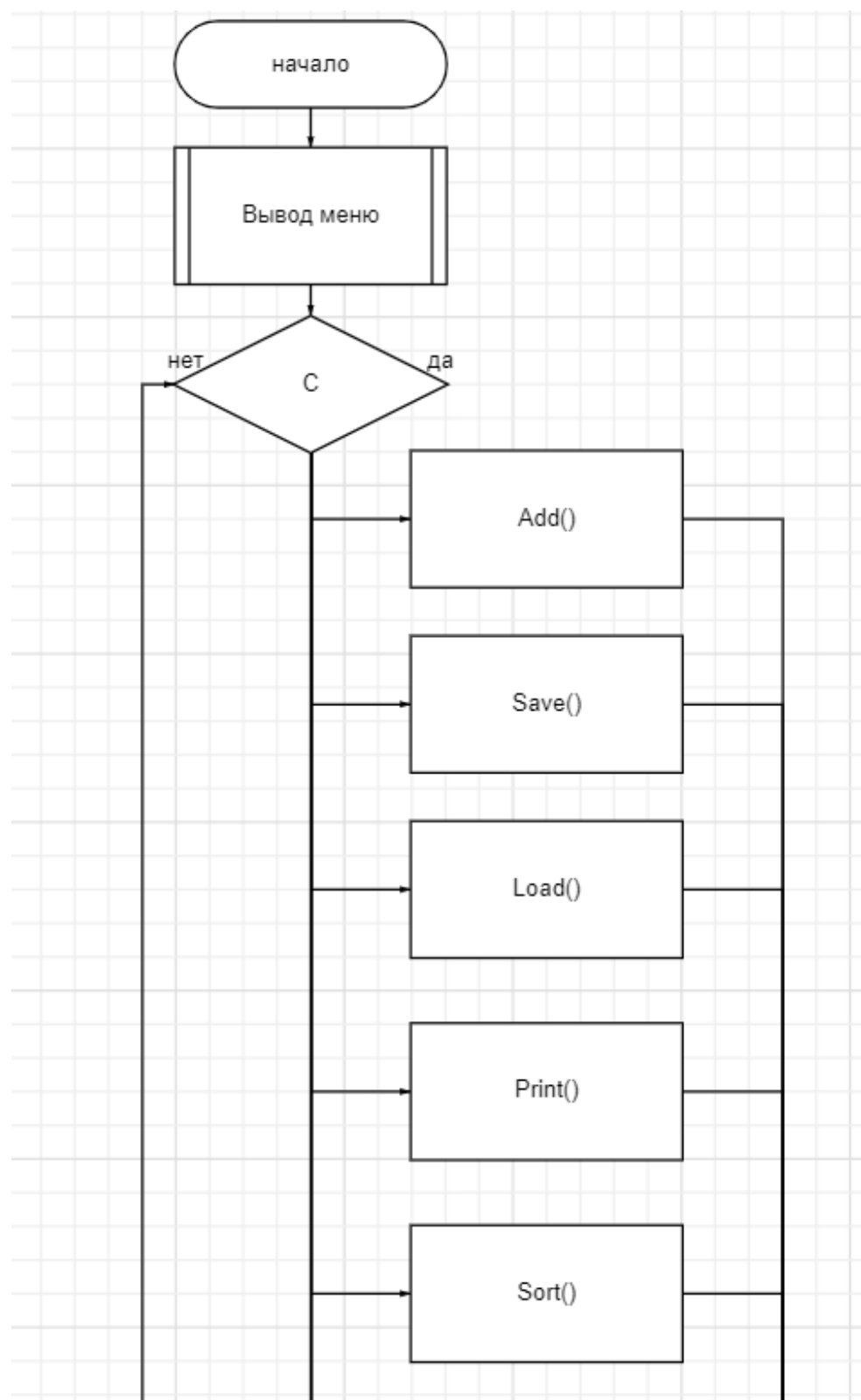


Рисунок 4- Схема обработки событий

Как только главная функция обнаружила событие от клавиатуры, вызывается функция обработки события. В зависимости от активного элемента выполняются соответствующие действия. Если события принадлежат меню, то выполняется перемещение выделенного пункта меню. Если выбрано меню, вызываются функции реализации меню. В функциях реализации меню находится свой цикл проверки событий клавиатуры. На выходе из функций реализации меню определяется, какой из пунктов меню был выбран.

Алгоритм представлен на рисунке 5.



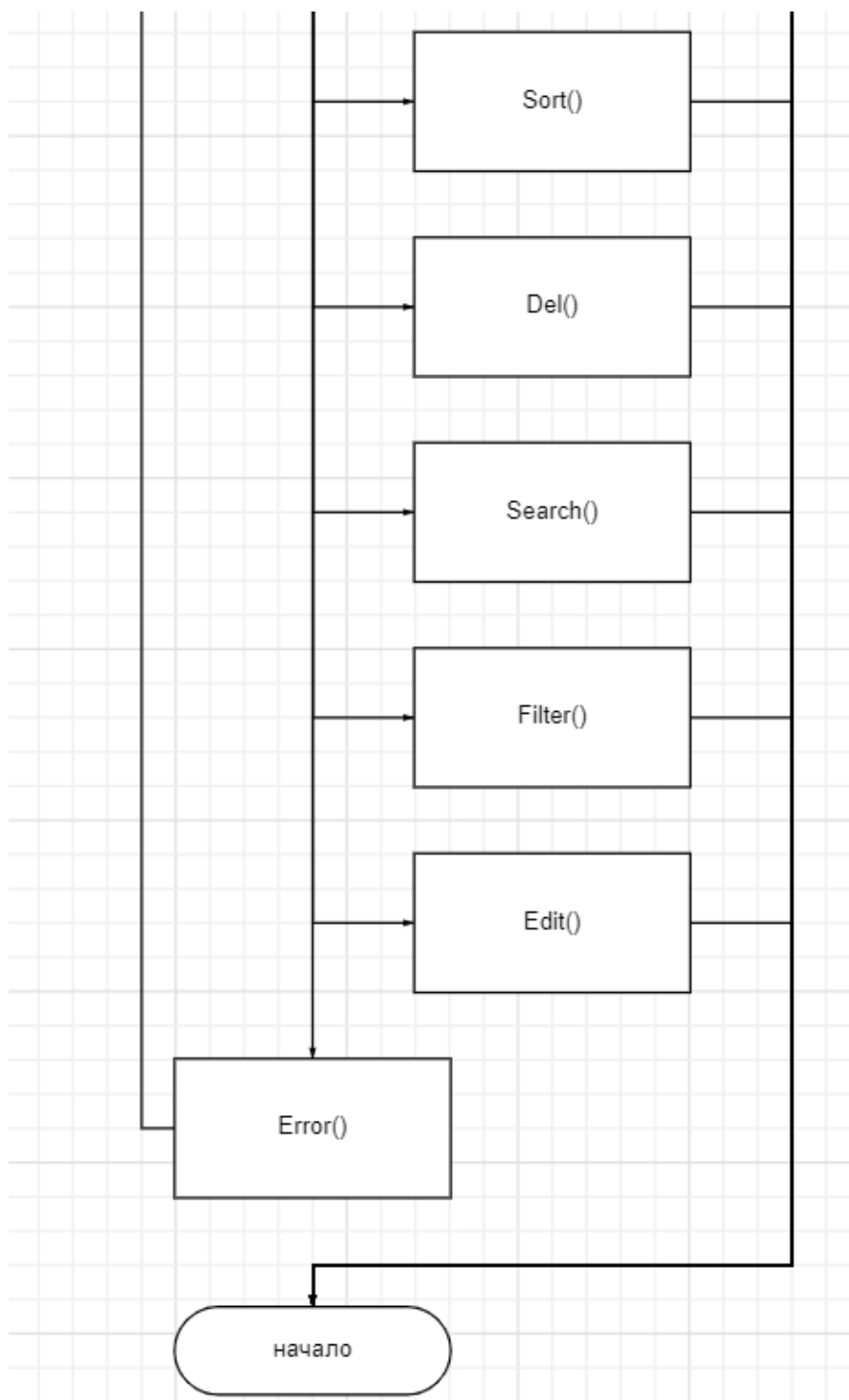


Рисунок 5- Алгоритм работы меню

2.3. Отладка и тестирование

В качестве среды разработки была выбрана программа VisualStudio2019. Программа предоставляет все средства необходимые при разработке и отладке разрабатываемых модулей и программ. Для отладки использовались такие инструменты как точки остановок,

выполнение кода по шагам, анализ содержимого локальных и глобальных переменных, анализ содержимого памяти.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с перерисовкой содержимого таблицы, с перерисовкой меню и многое другое. Также были добавлены проверки, связанные с недопустимостью обработки некоторых пунктов меню при определенных условиях.

2.4. Описание программы

Разработка приложения kursuch.exe

Приложение «kursuch.exe» является основным модулем программы. При запуске происходит активация меню, затем, выбрав соответствующую опцию, программа переходит к выполнению соответствующих функций. Выход из меню осуществляется после выбора одного из пунктов меню.

Если пользователь выбрал добавление новой записи, ему предлагается заполнить все поля таблицы по очереди. Запись добавляется в конец таблицы, счетчик записей увеличивается на единицу.

При выборе пунктов 2 или 3, происходит сохранение базы в файл или загрузка базы из файла соответственно.

При выборе пункта 4 на экран выведутся все записи. При выборе 5 пункта происходит сортировка столиц по алфавиту. Если пользователь выбрал 6 пункт, то пользователь должен ввести номер записи для удаления. При выборе пункта 7, пользователю предоставляется ввод столицы. После этого на экран выведутся записи, удовлетворяющие условию поиска. При выборе 8 пункта пользователю предлагается отфильтровать столицы по площади (предлагается найти все столицы, у которых площадь равна или больше введенной). После этого на экран выведутся записи, удовлетворяющие условию. При выборе 9 пункта

пользователю предоставляется список всех записей и ожидается ввод номера записи для редактирования. Если пользователь выбрал 9 пункт, то пользователю предлагается ввести все поля, после чего выводится отредактированная запись. При выборе пункта 0, пользователь выходит из программы.

Листинг программы приведен в приложении 1.

На рисунке 6 изображена иерархическая структура программы.

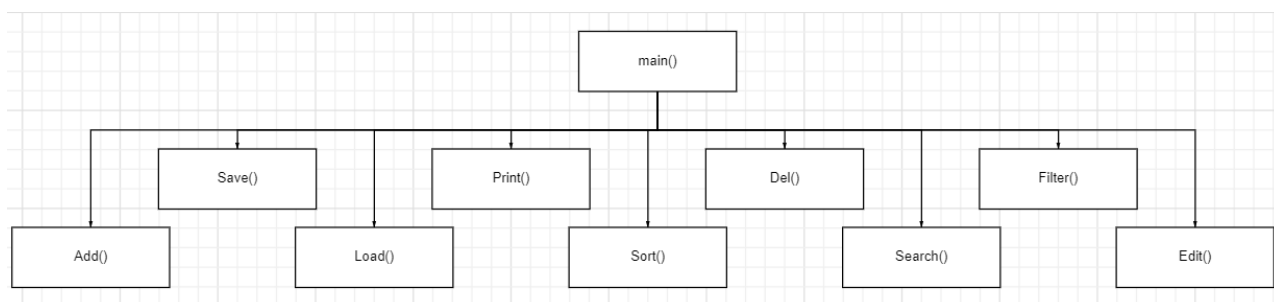


Рисунок 6- Схема иерархической структуры программы

Разработка меню

Меню реализовано в файле «`main.cpp`» и состоит из следующих пунктов:

Создание новой базы данных, Сохранение базы данных в файл, Загрузка базы данных из файла, Вывод базы данных на экран, Редактирование базы, Удаление записей, Сортировка записей, Поиск записей, Фильтрация записей, Выход.

Меню реагирует на нажатие клавиш: цифр от 0 до 9. При этом на выбранном пункте меню управление передается на соответствующую функцию.

Разработка функции на языке Ассемблер

Как известно, функции, реализованные на языке ассемблера, работают быстрее и занимают меньше места. В качестве разрабатываемой функции была выбрана функция, которая выводит ошибку, если мы ввели число, которого нет на панели меню.

Описание функции выполнено на языке C, а тело функции представляет

собой ассемблерную вставку, компилятор, входящий в состав среды разработки VS2010, позволяет это делать. Для этого перед ассемблерной вставкой необходимо написать ключевое слово “_asm”, а сам код поместить в фигурные скобки.

Для реализации функций вывода были использованы следующие ассемблерные инструкции:

- инструкция push выполняет «вталкивание» используемых регистров в стек
- инструкция mov выполняет копирование содержимого источника в приемник
- инструкция add суммирование содержимого источника и приемника
- инструкция lea перенос адрес источника в приемника
- инструкция call вызывает необходимую функцию

Код функции представлен в приложении.

Руководство пользователя

Программа «kursuch.exe» предназначена для хранения информации о различных столицах мира. Программа имеет интуитивно понятный интерфейс и поддерживает такие операции как добавление записи, удаление, редактирование, сортировку, фильтрацию, поиск по записям, загрузку базы данных из файла и сохранение в файл.

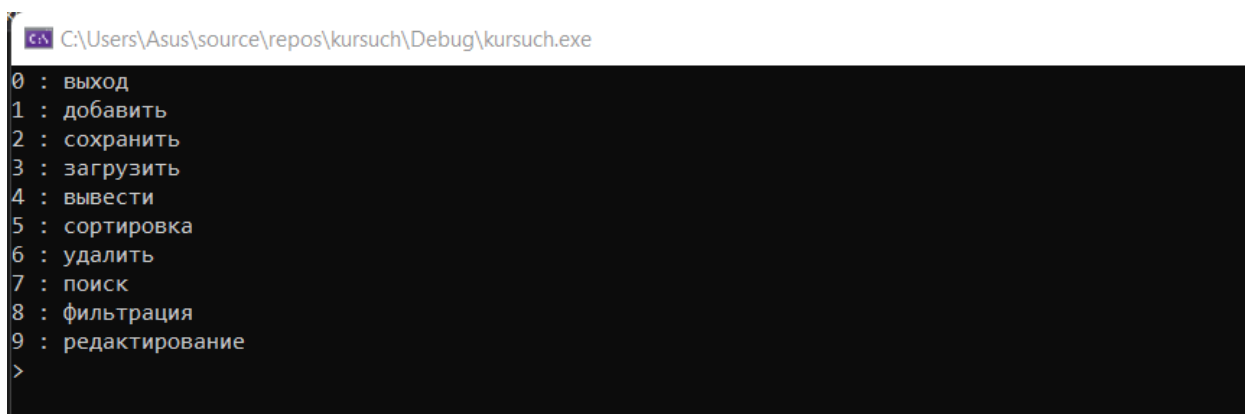


Рисунок 7- Вывод меню

Для загрузки базы данных из файла необходимо выбрать пункт меню «загрузка». При успешном выполнении операции будет выведено соответствующее сообщение.

```
>2
Сохранение в файл 'task1.txt' выполнено успешно0 : выход
```

Рисунок 8- Загрузка базы данных из файла

Для вывода всей базы данных на экран необходимо выбрать пункт меню «Вывод».

```
>4
N Столица Страна Площадь Население
1. москва россия 17100000 144000000
2. вашингтон америка 9000000 333000000
3. оттава канада 9985000 38000000
4. берлин германия 357588 83000000
5. лондон великобритания 243809 66000000
6. пекин китай 9597000 1402000000
7. киев украина 603000 44000000
8. нью-дели индия 3000000 1423000000
0 : выход
```

Рисунок 9- Вывод базы данных

Для добавления записи в базу данных необходимо выбрать пункт «добавление». После нужно ввести необходимые данные, после ввода каждой строки нажимать клавишу Enter. Запись будет добавлена в базу данных, и на экран будет выведено меню.

```
>1
Введите столицу > москва
Введите страну > россия
Введите площадь > 17100000
Введите население > 144000000
0
```

Рисунок 10- Добавление записи

Для удаления записи из базы данных следует выбрать пункт «удаление». После ввода номера записи она будет удалена из базы данных, и будет выведено сообщение «Удалено».

```

>6
N Столица Страна Площадь Население
1. москва россия 17100000 144000000
2. вашингтон америка 9000000 333000000
3. оттава канада 9985000 38000000
4. берлин германия 357588 83000000
5. лондон великобритания 243809 66000000
6. пекин китай 9597000 1402000000
7. киев украина 603000 44000000
8. нью-дели индия 3000000 1423000000
Введите номер удаляемого элемента > 7
0 : выход
1 : добавить
2 : сохранить
3 : загрузить
4 : вывести
5 : сортировка
6 : удалить
7 : поиск
8 : фильтрация
9 : редактирование
>4
N Столица Страна Площадь Население
1. москва россия 17100000 144000000
2. вашингтон америка 9000000 333000000
3. оттава канада 9985000 38000000
4. берлин германия 357588 83000000
5. лондон великобритания 243809 66000000
6. пекин китай 9597000 1402000000
7. нью-дели индия 3000000 1423000000
0 : выход

```

Рисунок 11- Удаление записи

Для редактирования записи необходимо выбрать пункт «редактирование». Для поиска записи, которую необходимо отредактировать нужно ввести столицу. Затем необходимо ввести новые данные.

```

>9
пекин
6. пекин китай 9597000 1402000000
Введите столицу > пекин
Введите страну > китай
Введите площадь > 9597000
Введите население > 130000000
6. пекин китай 9597000 130000000
0 : выход

```

Рисунок 12- Редактирование записи

Для поиска нужной записи базы данных нужно выбрать пункт «ПОИСК».

```

>7
оттава
3. оттава канада 9985000 38000000

```

Рисунок 13- Поиск записи

Для сортировки нужно выбрать пункт «сортировка». После чего записи в базе данных будут отсортированы по столицам.

```
>5
0 : выход
1 : добавить
2 : сохранить
3 : загрузить
4 : вывести
5 : сортировка
6 : удалить
7 : поиск
8 : фильтрация
9 : редактирование
>4
```

N	Столица	Страна	Площадь	Население
1.	берлин	германия	357588	83000000
2.	вашингтон	америка	9000000	333000000
3.	лондон	великобритания	243809	66000000
4.	москва	россия	17100000	144000000
5.	оттава	канада	9985000	38000000
6.	пекин	китай	9597000	130000000

Рисунок 14- Сортировка записей

Для фильтрации записей необходимо выбрать пункт «фильтрация» (предлагается найти все столицы, у которых площадь равна или больше введённой). После этого на экран выведутся записи, удовлетворяющие условию.

```
>8
Ввести площадь >360000
```

2.	вашингтон	америка	9000000	333000000
4.	москва	россия	17100000	144000000
5.	оттава	канада	9985000	38000000
6.	пекин	китай	9597000	130000000

Рисунок 15- Фильтрация базы данных

Заключение

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

программирования VisualStudio2010. Получены навыки отладки и тестирования программ.

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

Список используемых источников

1. Герберт Шилдт. Полный справочник по С.
2. Н. Культин. Основы программирования в Microsoft Visual C++ 2010, 2010г
3. Калашников О. – Ассемблер – это просто. Учимся программировать, 2011 г.
4. Скэнлон Л.Дж. Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера. 1991г.

Приложение 1. Листинги программы

Приложение 1.1

Файл bibl.h

```
#pragma once
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <locale.h>
#include <Windows.h>
#include <stdbool.h>
#pragma warning(disable: 4996)

#include "header.h"
```

Приложение 1.2

Файл Header.h

```
#pragma once

void Print(void);
void Add(void);
void Del(void);
void Save(void);
void Load(void);
void Sort(void);
void Search(void);
void Filter(void);
void Edit(void);
void Error(int a);
```

Приложение 1.3

Файл kursuch.cpp

```
#include "bibl.h"
```

```

/* Структура для хранения информации про одного человека */
typedef struct tagITEM
{
    char Capital[20], Country[20];
    int Square, Population;
} ITEM;

char* RUS(char* text)
{
    char* buf = (char*)malloc(strlen(text) + 1);
    OemToCharA(text, buf);
    return buf;
}

ITEM Men[20];

/* Количество занятых элементов в массиве */
int Number;

/* Функция для вывода всех элементов */

void Print(void)
{
    int i;

    printf(" N    Столица                Страна                Площадь\n\n");
    printf("Население\n");
    for (i = 0; i < Number; i++)
        printf("%2i.    %-20s    %-20s    %-20i    %3i\n", i + 1,
            Men[i].Capital,
                Men[i].Country, Men[i].Square, Men[i].Population);
}

/* Функция для добавления элемента */
void Add(void)
{

```

```

    if (Number == 19)
    {
        fprintf(stderr, "Невозможно добавить еще один элемент\n");
        return;
    }

    printf("Введите столицу > ");
    scanf("%s", &Men[Number].Capital);
    printf("Введите страну > ");
    scanf("%s", &Men[Number].Country);
    printf("Введите площадь > ");
    scanf("%i", &Men[Number].Square);
    printf("Введите население > ");
    scanf("%i", &Men[Number].Population);

    Number++;
}

/* Функция для удаления элемента */
void Del(void)
{
    int i;

    Print();
    printf("Введите номер удаляемого элемента > ");
    scanf("%i", &i);
    if (i < 1 || i > Number)
    {
        fprintf(stderr, "Элемент с номером %i не существует\n",
i);
        return;
    }

    for (i--; i < Number - 1; i++)
        Men[i] = Men[i + 1];
}

```

```

        Number--;
        printf("\nЭлемент удален");
    }
    /* Функция для сохранения массива в файле */
    void Save(void)
    {
        FILE* F;
        int i;

        if ((F = fopen("task1.dat", "wt")) == NULL)
        {
            fprintf(stderr, "Невозможно открыть для записи файл 'task1.dta'\n");
            return;
        }

        fprintf(F, "%i\n", Number);
        for (i = 0; i < Number; i++)
            fprintf(F, "%s\n%s\n%i\n%i\n", Men[i].Capital,
                    Men[i].Country, Men[i].Square, Men[i].Population);

        printf("Сохранение в файл 'task1.txt' выполнено успешно");
        fclose(F);
    }

    /* Функция для чтения массива из файла */
    void Load(void)
    {
        FILE* F;
        int i;

        if ((F = fopen("task1.dat", "rt")) == NULL)
        {

```

```

        fprintf(stderr, "Невозможно открыть для чтения файл
'task1.dta'\n");
        return;
    }

    fscanf(F, "%i", &Number);
    for (i = 0; i < Number; i++)
        fscanf(F, "%s%s%i%i", Men[i].Capital,
                Men[i].Country, Men[i].Square, Men[i].Population);

    fclose(F);
} /* End of 'Load' function */

/* Функция для упорядочивания массива по столицам */
void Sort(void)
{
    int i, j;
    ITEM Temp;

    for (j = Number - 1; j > 0; j--)
        for (i = 0; i < j; i++)
            if (strcmp(Men[i].Capital, Men[i + 1].Capital) > 0)
            {
                Temp = Men[i];
                Men[i] = Men[i + 1];
                Men[i + 1] = Temp;
            }
}

void Search(void)
{
    FILE* F;
    int i;
    if (Number == 0) {
        printf("Массив столиц пуст");
        return;
    }

```

```

    }

    char text[40] = "";
    scanf("%s", text);
    bool isFound = false;

    for (i = 0; i < Number; i++) {
        if (isFound) break;
        if (strcmp(Men[i].Capital, text) == 0) {
            printf("%2i.  %-20s  %-20s  %-20i  %3i\n", i + 1,
Men[i].Capital,
                                Men[i].Country,
                                Men[i].Square,
Men[i].Population);
            isFound = true;
        }
    }
    if (!isFound) {
        printf("Столица с таким названием не найдена");
    }

}

void Filter(void)
{
    int i, k;
    k = 0;
    if (Number == 0) {
        printf("Массив столиц пуст");
        return;
    }
    printf("Введите площадь >");
    bool isfound = false;
    scanf("%i", &k);
    for (i = 0; i < Number; i++) {
        if (Men[i].Square >= k) {

```

```

        isfound = true;
        printf("%2i.  %-20s  %-20s  %-20i  %3i\n", i + 1,
Men[i].Capital,
                Men[i].Country,
                Men[i].Square,
Men[i].Population);
    }
}
if (!isfound) {
    printf("Столицы с площадью больше введённой не найдена");
}
}

void Edit(void)
{
    int i;
    if (Number == 0) {
        printf("Массив столиц пуст");
        return;
    }

    char text[40] = "";
    scanf("%s", text);
    bool isFound = false;

    for (i = 0; i < Number; i++) {
        if (isFound) break;
        if (strcmp(Men[i].Capital, text) == 0) {

            printf("%2i.  %-20s  %-20s  %-20i  %3i\n", i + 1,
Men[i].Capital,
                    Men[i].Country,
                    Men[i].Square,
Men[i].Population);

            printf("Введите столицу > ");
            scanf("%s", &Men[i].Capital);

```

```

        printf("Введите страну > ");
        scanf("%s", &Men[i].Country);
        printf("Введите площадь > ");
        scanf("%i", &Men[i].Square);
        printf("Введите население > ");
        scanf("%i", &Men[i].Population);

        printf("%2i.  %-20s  %-20s  %-20i  %3i\n", i + 1,
Men[i].Capital,
        Men[i].Country,
Men[i].Population);
        Men[i].Square,
        isFound = true;
    }
}
if (!isFound) {
    printf("Столица с таким названием не найдена");
}
}
void Error(int a) {

    char error_[] = "\nДанной команды %с нет\n";
    char error_continue[] = "попробуйте снова!\n\n";

    __asm {
        mov edi, a;
        push edi;
        lea eax, error_;
        push eax;
        call printf;
        add esp, 8
        lea ebx, error_continue;
        push ebx;
        call printf;
        add esp, 4;
    }
}

```

```
}  
}
```

Приложение 1.4.

Файл main.cpp

```
#include "bibl.h"  
  
int Menu(void)  
{  
    int c = 0;  
  
    while ((c < '0' || c > '9') && c != 27)  
    {  
        printf("0 : выход\n"  
               "1 : добавить\n"  
               "2 : сохранить\n"  
               "3 : загрузить\n"  
               "4 : вывести\n"  
               "5 : сортировка\n"  
               "6 : удалить\n"  
               "7 : поиск\n"  
               "8 : фильтрация\n"  
               "9 : редактирование\n"  
               ">");  
        c = getch();  
        printf("%c\n", c);  
        if (c < '0' || c > '9') Error(c);  
    }  
    return c;  
}  
  
void main(int)  
{  
    SetConsoleCP(1251);  
    SetConsoleOutputCP(1251);
```

```

int Selection;
int Number;
Number = 0;
while ((Selection = Menu()) != '0' && Selection != 27)
    switch (Selection)
    {
    case '1':
        Add();
        break;
    case '2':
        Save();
        break;
    case '3':
        Load();
        break;
    case '4':
        Print();
        break;
    case '5':
        Sort();
        break;
    case '6':
        Del();
        break;
    case '7':
        Search();
        break;
    case '8':
        Filter();
        break;
    case '9':
        Edit();
        break;
    }
}

```

Приложение.

Результаты работы программы, снимки экрана, содержимое файлов.

Приложение 1

Москва	Россия	17100000	144000000
Вашингтон	Америка	9500000	333000000
Оттава	Канада	9985000	38000000
Берлин	Германия	357588	83000000
Лондон	Великобритания	243809	66000000
Пекин	Китай	9597000	1402000000
Киев	Украина	603000	44000000
Нью-дели	Индия	3000000	1423000000