Оглавление

[1.ЗАДАНИЕ 1](#_Toc414514362)

[1.1.Тема курсового проекта 1](#_Toc414514363)

[1.2.Постановка задачи: 1](#_Toc414514364)

[1.3.Перечень основных вопросов, подлежащих изучению и разработке: 1](#_Toc414514365)

[1.4.Срок сдачи курсового проекта 1](#_Toc414514366)

[2.ВВЕДЕНИЕ 2](#_Toc414514367)

[2.1.Основные задачи состоят в следующем: 2](#_Toc414514368)

[2.2.Функциональное назначение программы 2](#_Toc414514369)

[3.ОСНОВНАЯ ЧАСТЬ 3](#_Toc414514370)

[3.1.Требование к программному средству 3](#_Toc414514371)

[3.2.Требования к структуре программного средства 3](#_Toc414514372)

[3.3.Требования к надежности 3](#_Toc414514373)

[3.4.Условия эксплуатации 3](#_Toc414514374)

[3.5.Требования к составу и параметрам технических средств 3](#_Toc414514375)

[3.6.Требования к информационной и программной совместимости 4](#_Toc414514376)

[3.6.1.Требование к информационным структурам и методам решения 4](#_Toc414514377)

[3.6.2.Требования к исходным кодам и языкам программирования 4](#_Toc414514378)

[3.6.3.Требования к программным средствам, используемым программой](#_Toc414514379)

[3.6.4.Специальные требования 4](#_Toc414514380)

[3.7.Требования к программной документации 4](#_Toc414514381)

[3.8.Требования к эргономике и технической эстетике 4](#_Toc414514382)

[3.9.Декомпозиция поставленной задачи 6](#_Toc414514383)

[3.10.Общая архитектура программного средства 6](#_Toc414514384)

[3.11.Реализация функционального назначения программного средства 7](#_Toc414514385)

[3.11.1.Создание файла базы данных 7](#_Toc414514386)

[3.11.2.Работа с базой данных 7](#_Toc414514387)

[3.15.Сообщение системному программисту 9](#_Toc414514388)

[3.16.Сообщение пользователю 9](#_Toc414514389)

[3.17.Сообщения программисту 13](#_Toc414514390)

[4.ГРАФИЧЕСКАЯ ЧАСТЬ 15](#_Toc414514391)

[4.1. Функциональная структура программы 15](#_Toc414514392)

[4.2.Результаты работы программы 16](#_Toc414514393)

[5.ЗАКЛЮЧЕНИЕ 17](#_Toc414514394)

[6.СПИСОК ЛИТЕРАТУРЫ 18](#_Toc414514395)

[ПРИЛОЖЕНИЕ А 19](#_Toc414514396)

[ПРИЛОЖЕНИЕ Б 20](#_Toc414514397)

[ЛИСТ ЗАМЕЧАНИЙ ПО ПОЯСНИТЕЛЬНОЙ ЗАПИСКЕ КУРСОВОГО ПРОЕКТА 50](#_Toc414514398)

# 1.ЗАДАНИЕ

## 1.1.Тема курсового проекта

Целью данной курсовой работы являлось создание автоматизированной базы данных: «Личная телефонная книжка».

## 1.2.Постановка задачи:

1. Создание файла «Личной телефонной книжки» для каждого пользователя.
2. Защита личных данных от других пользователей.
3. Основные функции базы данных: добавление, редактирование, удаление, поиск, сортировка, просмотр данных.
4. Возможность переноса данных с одного ПК на другой и просмотр данных.

## 1.3.Перечень основных вопросов, подлежащих изучению и разработке:

1. Актуальность и значимость темы.
2. Алгоритм программы.
3. Выбор языка программирования.
4. Анализ специальной литературы.

## 1.4.Срок сдачи курсового проекта

Срок сдачи курсового проекта – 19 марта 2015 года.

# 2.ВВЕДЕНИЕ

В век электронных технологий, интернета, мобильных телефонов и прочего мы заводим много новых знакомых и нам нужно где-то записывать всю информацию, нужную для общения с конкретными лицами. В письменном виде возникает много неудобств в этом плане.

Автоматизация позволяет хранить, структурировать и систематизировать большие объемы данных.

Преимуществом компьютерной обработки перед ручной работой является скорость обработки данных, удобное представление данных, автоматическое заполнение некоторых данных, богатые возможности по обработке данных, например, легко можно найти любого человека по заданным критериям, что упрощает поиск конкретных людей и информацию о них.

Целью работы является разработка автоматизированной базы данных: «Личная телефонная книжка».

## 2.1.Основные задачи состоят в следующем:

1. Исследовать содержание программы.
2. Выбрать язык программирования.
3. Разработать алгоритм программы.
4. Продумать удобный интерфейс для пользователя.
5. Кодирование программы на выбранном языке.
6. Тестирование и отладка программы.
7. Сбор результатов.
8. Оформление отчёта.

## 2.2.Функциональное назначение программы

Программа предназначена для личного пользования.

Функциональным назначением программы является хранение личных записей пользователя, их редактирование и поиск нужной информации.

# 3.ОСНОВНАЯ ЧАСТЬ

## 3.1.Требование к программному средству

Функциональным требованием программы является представление информации о знакомых и занесение их в базу. Редактирование записей. Подробная информация о каждой записи. Поиск в базе по определенным критериям. Контроль вводимых данных. Простота интерфейса, вывод выходных данных в виде таблиц.

## 3.2.Требования к структуре программного средства

Для более удобной работы в программе в дальнейшем нужно будет доработать систему поиска по записям и добавить больше полей для записи информации (расширить базу данных) по требованию заказчика.

## 3.3.Требования к надежности

Т.к. база данных программы основывается на библиотеках Microsoft Access при создании базы данных есть возможность установить на нее личный пароль, который шифруется и сохраняется в файле базы данных, и при попытке его открыть через программу или же через Microsoft Access будет выводится сообщение о том, что база данных требует пароль для доступа и попросит его ввести.

## 3.4.Условия эксплуатации

Для корректной работы в программе пользователю потребуется минимальное умение пользоваться ПК и знать русский язык.

## 3.5.Требования к составу и параметрам технических средств

В состав технических средств должен входить IВМ-совместимый персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

* 1. Процессор Pentium-2.0Hz, не менее.
  2. Оперативную память объемом, 1Гигабайт, не менее.
  3. Операционную систему Windows XP и выше.
  4. Microsoft .NET Framework 4

## 3.6.Требования к информационной и программной совместимости

### 3.6.1.Требование к информационным структурам и методам решения

Информационная структура файлов должна включать в себя текст, содержащий разметку, предусмотренную спецификацией формата Microsoft Access.

### 3.6.2.Требования к исходным кодам и языкам программирования

Исходные коды программы должны быть реализованы на подходящем языке программирования с целью лучшей оптимизации. В качестве интегрированной среды разработки должна быть использована среда Visual Studio 2010 и выше.

### 3.6.3.Требования к программным средствам, используемым программой

Системные программные средства, используемые программой, должны быть представлены в лицензионной локализованной версии операционной системы Windows.

### 3.6.4.Специальные требования

Программа должна обеспечивать взаимодействие с пользователем (оператором) посредством графического пользовательского интерфейса, разработанного согласно рекомендациям компании-производителя операционной системы.

## 3.7.Требования к программной документации

Документацию необходимо оформлять по ГОСТ 7.32-2001 «Отчет о научно-исследовательской работе. Структура и правила оформления» и ГОСТ 19.106-78 «Единая система программной документации. Требования к программным документам, выполненным печатным образом».

Состав программной документации должен включать в себя:

1. Описание алгоритма.
2. программу и методики испытаний.
3. руководство системного программиста.
4. руководство оператора.

## 3.8.Требования к эргономике и технической эстетике

Система должна иметь человеко-машинный интерфейс, удовлетворяющий следующим требованиям:

1. Взаимодействие системы и пользователя должно осуществляться на русском языке, за исключением системных сообщений, не подлежащих русификации.
2. При работе с интерфейсом пользователь должен быть ориентирован на работу с клавиатурой и манипулятором графической информации «мышь».
3. Должно быть реализовано отображение на экране только тех возможностей, которые доступны конкретному пользователю в соответствии с его функциональной ролью в системе.
4. Представление управляющих элементов, экранных форм и их информационных элементов (окон, панелей и т.п.) должно быть унифицировано. Экранные формы должны полностью находиться в видимой площади экрана монитора с диагональю 17’ при разрешении экрана 1280 х 1024 и выше.

## 3.9.Декомпозиция поставленной задачи

Поставленная задача была разбита на следующие подзадачи:

1. Анализ требуемых функций программы.
2. Анализ пунктов меню.
3. Анализ методов получения и вывода входной, и выходной информации.
4. Анализ языков программирования.
5. Исследование возможностей языков.
6. Выбор языка.
7. Разработка алгоритма регистрации клиента.
8. Разработка алгоритма регистрации квартиры клиента.
9. Разработка алгоритма регистрации заявок клиентов.
10. Разработка алгоритма подбора квартир по заявкам от клиентов.
11. Анализ количества требуемых форм.
12. Разработка дизайна.
13. Программирование прототипа программы.
14. Добавление заранее разработанного интерфейса.
15. Анализ методов тестирования программы.
16. Тестирование программы.
17. Сбор результатов.
18. Оформление отчёта.

## 3.10.Общая архитектура программного средства

Т.к. программа использует библиотеки из Microsoft .NET Framework 4 Общая архитектураПСвыглядит следующим образом:

Таблица - Структура программного средства

|  |  |
| --- | --- |
| PrivateNoteBook.exe | Главный исполняемый файл |

Передача информации от одного модуля к другому происходит путем получения поля из одного класса в другой и для всех работ с базой данной используется класс «WorkWithDB.cs».

## 3.11.Реализация функционального назначения программного средства

### 3.11.1.Создание файла базы данных

Таблица - Имя файла, пароль

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип поля | Описание |
| Login | string | Имя файла |
| Password | string | Пароль |

Данная функция находится в «WorkWithDB.cs»

CreateFile

Для выполнения этой функции требуется указать имя файла (Логин) и по желанию пароль. В результате выполнения функции создастся новый файл базы данных с форматом «.mdb».

### 3.11.2.Работа с базой данных

Таблица - Структура базы данных

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип поля | Описание |
| Id | ID | Идентификатор в базе |
| Фамилия | string | Имя |
| Имя | string | Фамилия |
| Отчество | string | Отчество |
| Дата\_Рождения | DATE | Электронная почта |
| Телефон | string | Телефон |
| Адрес | string | Адрес |
| Email | string | Электронный адрес |
| Кем\_Является | string | Кем является человек |
| Доп\_Информация | string | Доп.Информация |
| Взял\_В\_Займы | double | Сколько человек взял в займы |
| Дай\_В\_Займы | double | Сколько человек дал вам в займы |
| Долг | double | Общий долг человека |
| Фото | Byte array | Массив байтов картинки (фото) |

Данные функции находится в «WorkWithDB.cs»

LoadFile –Чтение из файла и вывод в DataGridView

ReadFromFile – Чтение из файла и вывод в конкретные поля

AddNewLine – Добавление записи

deleterow – Удаление записи

ChangeLine – Изменение записи

## 3.15.Сообщение системному программисту

Инструкция по установке:

1. Запустить установочный файл и выбрать язык.
2. Откроется окно приветствия нажать кнопку далее
3. Выбрать место установки программы
4. Задать имя отображения в панели пуск или выбрать опцию не создавать ярлык в меню панели пуск и нажать далее
5. Включить или выключить создание ярлыка на рабочем столе и нажать далее, и установить.
6. Во время установки программа запустит установку Microsoft .NET Framework 4 (Необходимые компоненты для корректной работы программы). После установки программы установщик выдаст что установка прошла успешно и можно выбрать опцию запустить программу или не запускать, после этого нажать кнопку завершить.

Базы данных создаются в папке DataBase в папке с программой.

## 3.16.Сообщение пользователю

Для создания нового файла базы данных запустите программу и на форме «Входа» введите логин (имя файла) и пароль (по желанию) и нажмите кнопку создать.

Если же файл уже создан, то введите логин, пароль (если таковой имеется) и нажмите кнопку войти.

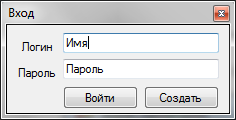


Рисунок 3.16.1 Вход

Далее для создания новой записи нажмите кнопку «Добавить запись»

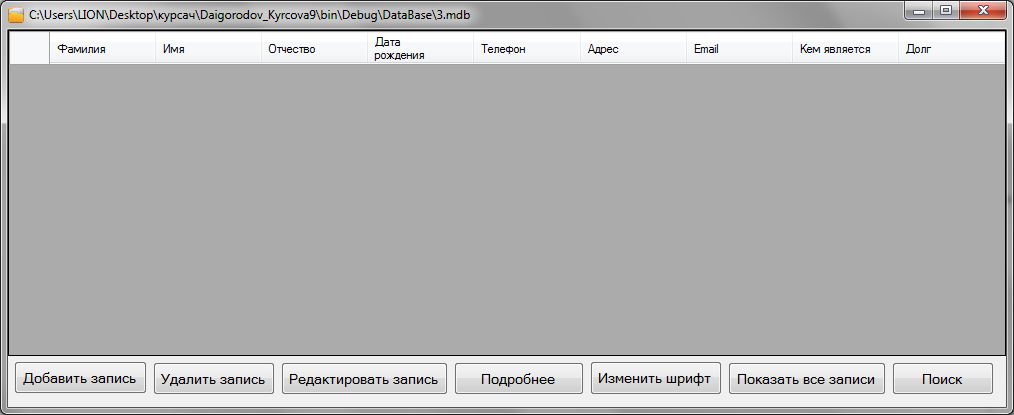


Рисунок 3.16.2 - Основное окно

и заполните форму добавления записи.

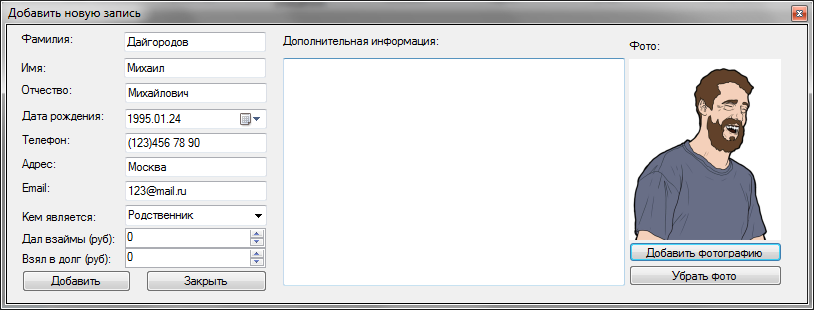


Рисунок 3.16.3 - Форма добавление записи

Нажмите на кнопку «Добавить фотографию» если нужно добавить картинку или фото к записи, для удаление картинки нажмите «Убрать фото».

Для сохранения записи в файле нажмите «Добавить».

Для редактирования, удаления, подробного просмотра записи выберите запись мышкой и нажмите соответствующую кнопку.

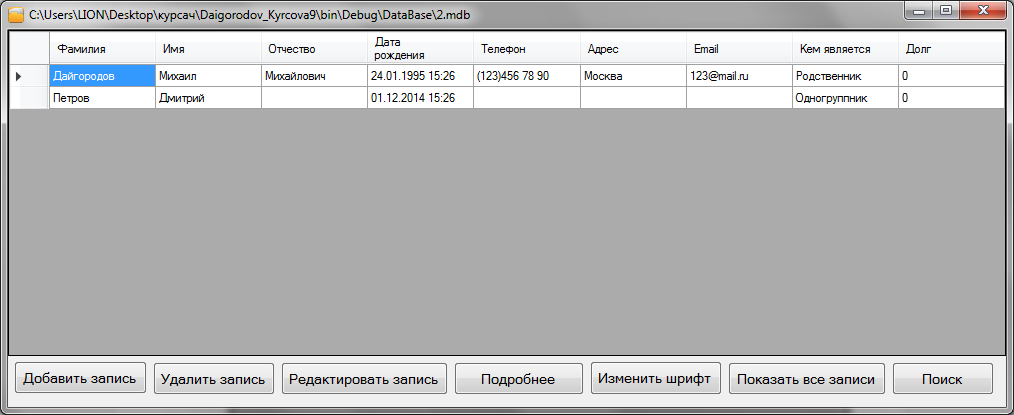


Рисунок 3.16.4 - Основное окно 2

При нажатии на кнопку «Изменить шрифт» появится окно, позволяющее выбрать шрифт, его размер и цвет.

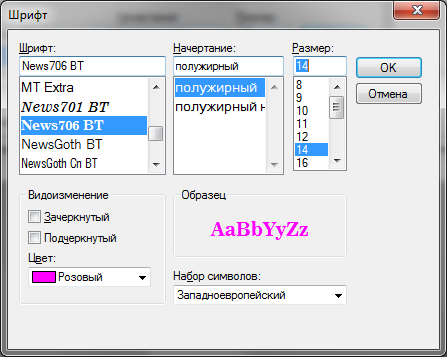


Рисунок 3.16.5 - Выбор шрифта

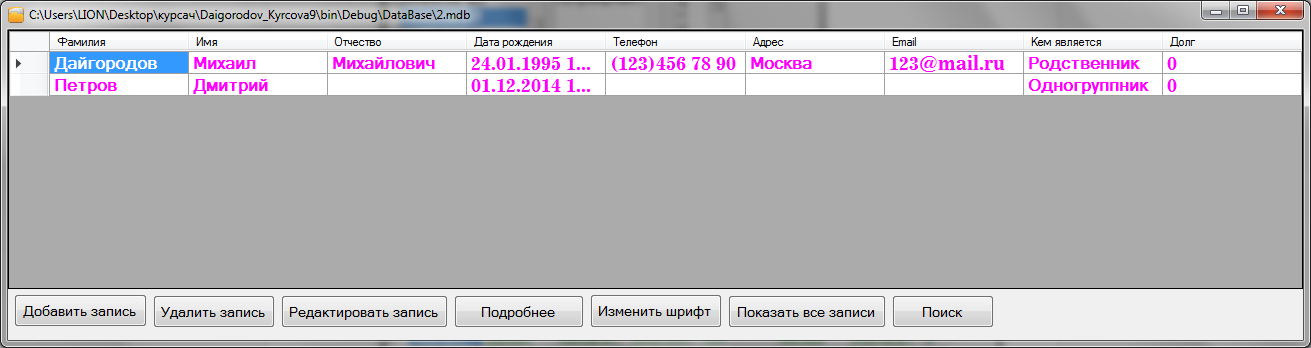


Рисунок 3.16.6 - Основное окно 3

При нажатии на кнопку «Поиск» появится окно с «фильтрами» для поиска записей с определенными критериями по базе данных. Заполните нужные поля и поставьте галочку напротив поля, чтобы включить фильтрование базы по данному критерию.

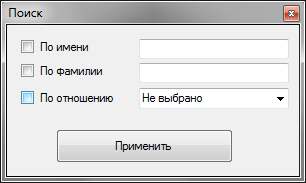


Рисунок 3.16.7 - Поиск

Нажмите на кнопку «Показать все записи» чтобы выключить все фильтры.

## 3.17.Сообщения программисту

Для компиляции проекта необходимо следующее минимальное программное обеспечение:

* операционная среда - MS Windows XP/ Vista/ 7/ 8.
* среда разработки Visual Studio 2012 или выше.

Программа написана на языке C#, в рамках компилятора Visual Studio.

Программа использует базу данных Microsoft Access.

Все основные функции работы с базой данных описаны в классе «WorkWithDB.cs». Создание таблицы в файле осуществляется путем SQL команд. Для изменения, добавления, удаления полей в таблице следует использовать функцию CreateTable используя в качестве параметров SQL команду создания таблицы, используя синтаксис SQL.

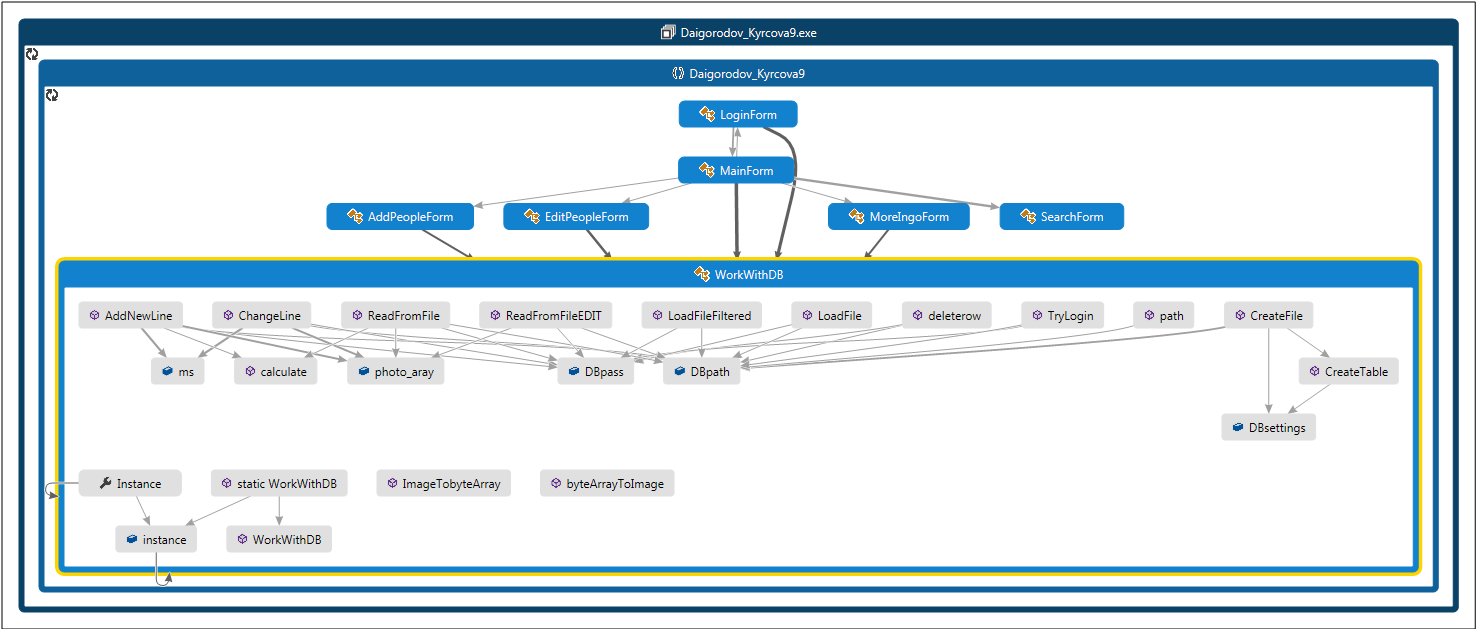
Пример:

CreateTable("CREATE TABLE People(ID Identity(1,1), " + "Фамилия String, " + "Имя String, " + "Отчество String, " + "Дата\_Рождения Date, " + "Телефон String, " + "Адресс String, " + "Email String, " + "Кем\_Является String, " + "Доп\_Информация String, " + "Взял\_В\_Займы Double, " + "Дал\_В\_Займы Double, " + "Долг Double, " + "Фото IMAGE, " + "Primary Key (ID));");

Добавлять данную функцию следует в метод CreateFile, который отвечает за создание нового файла.

# 4.ГРАФИЧЕСКАЯ ЧАСТЬ

## 4.1. Функциональная структура программы



## 4.2.Результаты работы программы

Основное окно программы со всеми записями и функциями:

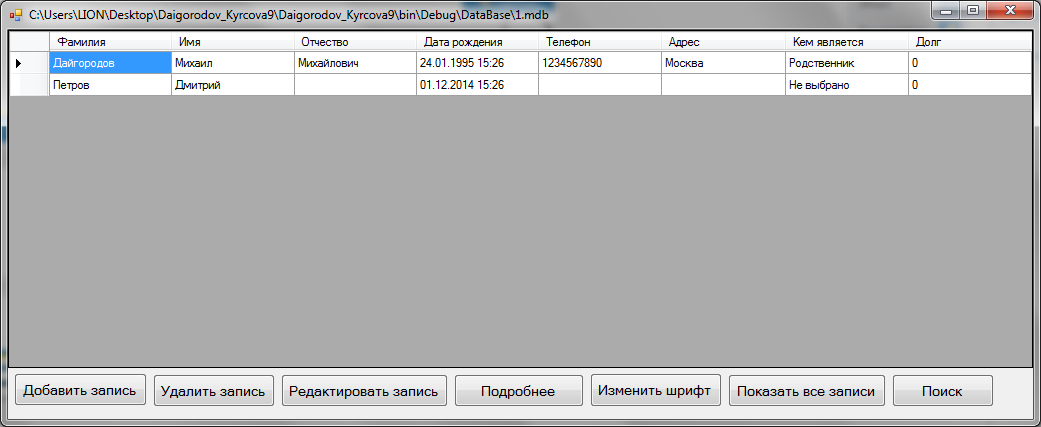


Рисунок 4.2.1 Окно программы

# 5.ЗАКЛЮЧЕНИЕ

В данной курсовой работе была достигнута частичная автоматизация личной телефонной книжки.

Целью проекта является значительное упрощение создания, хранения, переноса, поиска и редактирования личных записей, удобства работы с ними и просмотра. Отпадает необходимость использования множества бумаг для хранения всех нужных нам данных.

Были изучены возможности языка SQL в Microsoft Access и его интеграция в c#. Благодаря множеству запросов в SQL мы можем очень легко структурировать нашу базу банных под любые нужны, а также производит поиск и редактирование любых записей.

Данный проект можно использовать в домашних условиях, а также есть возможно легко модернизировать его под нужны определенный круг людей, также при необходимости проект можно улучшить во много раз и разрешить пользователю вводить свои личные SQL запросы, а не только те, которые хранятся в коде проект. Так же при углубленном изучении SQL и мобильных операционных систем есть возможность перенести данный проект на мобильные устройства и создать синхронизацию личных данных между различными устройствами.

# 6.СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ19.106-78 Единая система программной документации ТРЕБОВАНИЯ К ПРОГРАММНЫМ ДОКУМЕНТАМ, ВЫПОЛНЕННЫМ ПЕЧАТНЫМ СПОСОБОМ
2. Павловская Т.А. C#. Программирование на языке высокого уровня: учебник для вузов. – СПб.: Питер, 2009. – 432 с.
3. Зиборов В.В. Visual C# 2012 на примерах. - СПб.: БХВ-Петербург, 2013.- 480с
4. Зубкова Т.М. Технология разработки программного обеспечения: Учебное пособие. - Оренбург: ГОУ ОГУ, 2004. – 101 с.
5. [krimel.karelia.ru](http://www.krimel.karelia.ru/file.cgi?id=325) — рекомендации по оформлению рефератов, контрольных, курсовых и дипломных работ (в формате DOC);
6. [vsegost.com](http://vsegost.com/Catalog/27/2737.shtml) — межгосударственный стандарт 7.32—2001 на структуру и оформление отчета о научно-исследовательской работе.

# ПРИЛОЖЕНИЕ А

На диске с курсовым проектом находятся:

1. Установщик программы Setup.exe, который также включает в себя установку Microsoft .NET Framework 4.
2. Документацию в электронном виде.
3. Проект программы в формате «.sln» для Visual Studio 2012 и выше.
4. Код программы.
5. Отдельным файлом титульный лист.
6. Графическую часть проекта «Функциональная структура программы» в формате «.dgml», «.modelproj» в папке ModelingProject1, а также в формате «.xps» и «.bmp» в корневой папке диска.
7. Презентация

# ПРИЛОЖЕНИЕ Б

class Program

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Daigorodov\_Kyrcova9

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new LoginForm());

}

}

}

class WorkWithDB

using System;

using System.ComponentModel;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using ADOX;

using System.IO;

using System.Drawing;

using System.Drawing.Imaging;

namespace Daigorodov\_Kyrcova9

{

class WorkWithDB

{

string DBpath;

string DBsettings;

string DBpass;

MemoryStream ms;

byte[] photo\_aray;

static readonly WorkWithDB instance = new WorkWithDB();

public static WorkWithDB Instance

{

get

{

return instance;

}

}

public string path()

{

return DBpath;

}

//Создание нового файла

public bool CreateFile(string filename, string password)

{

if (!Directory.Exists(AppDomain.CurrentDomain.BaseDirectory + "DataBase\\"))

{

Directory.CreateDirectory(AppDomain.CurrentDomain.BaseDirectory + "DataBase\\");

}

ADOX.Catalog cat = new Catalog();

DBpath = AppDomain.CurrentDomain.BaseDirectory + "DataBase\\" + filename + ".mdb";

if (File.Exists(DBpath))

{

MessageBox.Show("Файл с таким именем уже существует!");

return false;

}

else

{

DBsettings = "provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + password + ";";

cat.Create(DBsettings);

cat = null;

CreateTable("CREATE TABLE People(ID Identity(1,1), " + "Фамилия String, " + "Имя String, " + "Отчество String, " + "Дата\_Рождения Date, " + "Телефон String, " + "Адресс String, " + "Email String, " + "Кем\_Является String, " + "Доп\_Информация String, " + "Взял\_В\_Займы Double, " + "Дал\_В\_Займы Double, " + "Долг Double, " + "Фото IMAGE, " + "Primary Key (ID));");

return true;

}

}

//Создание нового файла

//Создание таблицы в файле

public void CreateTable(string Que)

{

string conn = "provider=Microsoft.Jet.OLEDB.4.0;" + "data source=" + DBsettings;

OleDbConnection connect = new OleDbConnection(conn);

connect.Open();

using (OleDbCommand command = new OleDbCommand(Que, connect))

{

try

{

command.ExecuteNonQuery();

}

catch (OleDbException ex)

{

MessageBox.Show("Произошла ошибка при создании таблицы\n" + ex.Message);

}

}

}

//Создание таблицы в файле

//Попытка чтения файла с указанным именем и паролем

public void TryLogin(string filename, string password)

{

string connString = "provider=Microsoft.Jet.OleDb.4.0;Data Source=" + AppDomain.CurrentDomain.BaseDirectory + "DataBase\\" + filename + ".mdb" + ";Jet OLEDB:Database Password=" + password + ";";

string query = "SELECT \* FROM People";

OleDbDataAdapter dAdapter = new OleDbDataAdapter(query, connString);

DataTable dTable = new DataTable();

dAdapter.Fill(dTable);

DBpath = AppDomain.CurrentDomain.BaseDirectory + "DataBase\\" + filename + ".mdb";

DBpass = password;

}

//Попытка чтения файла с указанным именем и паролем

//Чтение файла и вывод в DataGridView

public void LoadFile(DataGridView dataGridView1)

{

//create the connection string

string connString = "provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";";

//create the database query

string query = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People";

//create an OleDbDataAdapter to execute the query

OleDbDataAdapter dAdapter = new OleDbDataAdapter(query, connString);

//create a command builder

OleDbCommandBuilder cBuilder = new OleDbCommandBuilder(dAdapter);

//create a DataTable to hold the query results

DataTable dTable = new DataTable();

//fill the DataTable

dAdapter.Fill(dTable);

//the DataGridView

//BindingSource to sync DataTable and DataGridView

BindingSource bSource = new BindingSource();

//set the BindingSource DataSource

bSource.DataSource = dTable;

//set the DataGridView DataSource

dataGridView1.DataSource = bSource;

}

//Чтение файла и вывод в DataGridView

//Чтение файла и вывод в DataGridView с включенным фильтром данных

public void LoadFileFiltered(DataGridView dataGridView1,bool namecheck,string name, bool lastnamecheck,string lastname,bool whocheck,string who)

{

//create the connection string

string connString = "provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";";

//create the database query

string query = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является FROM People";

//Команды поиска

string name1 = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE Имя LIKE'" + name + "'";

string last = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE Фамилия LIKE'" + lastname + "'";

string who1 = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE Кем\_Является ='" + who + "'";

string namelast = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE (Имя LIKE'" + name + "' AND Фамилия LIKE'" + lastname + "')";

string namewho = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE (Имя LIKE'" + name + "' AND Кем\_Является ='" + who + "')";

string lastwho = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE (Фамилия LIKE'" + lastname + "' AND Кем\_Является ='" + who + "')";

string namelastwho = "SELECT ID,Фамилия,Имя,Отчество,Дата\_Рождения,Телефон,Адресс,Email,Кем\_Является,Долг FROM People WHERE (Имя LIKE'" + name + "' AND Фамилия LIKE'" + lastname + "' AND Кем\_Является ='" + who + "')";

if (namecheck) query = name1;

if (lastnamecheck) query = last;

if (whocheck) query = who1;

if (namecheck && lastnamecheck) query = namelast;

if (namecheck && whocheck) query = namewho;

if (lastnamecheck && whocheck) query = lastwho;

if (namecheck && lastnamecheck && whocheck) query = namelastwho;

//create an OleDbDataAdapter to execute the query

OleDbDataAdapter dAdapter = new OleDbDataAdapter(query, connString);

//create a command builder

OleDbCommandBuilder cBuilder = new OleDbCommandBuilder(dAdapter);

//create a DataTable to hold the query results

DataTable dTable = new DataTable();

//fill the DataTable

dAdapter.Fill(dTable);

//the DataGridView

//BindingSource to sync DataTable and DataGridView

BindingSource bSource = new BindingSource();

//set the BindingSource DataSource

bSource.DataSource = dTable;

//set the DataGridView DataSource

dataGridView1.DataSource = bSource;

}

//Чтение файла и вывод в DataGridView с включенным фильтром данных

//Преобразование массива байтов в картинку

public static Image byteArrayToImage(byte[] fileBytes)

{

using (MemoryStream fileStream = new MemoryStream(fileBytes))

{

Image temp;

try

{

temp = Image.FromStream(fileStream);

}

catch (ArgumentException) {

return null;

}

return temp;

}

}

//Преобразование массива байтов в картинку

//Преобразование картинки в массив байтов

public static byte[] ImageTobyteArray(Image image)

{

ImageConverter converter = new ImageConverter();

return (byte[])converter.ConvertTo(image, typeof(byte[]));

}

//Преобразование картинки в массив байтов

//Добавление новой записи

public void AddNewLine(TextBox textname, TextBox textname2, TextBox textname3, DateTimePicker date, TextBox phone, TextBox addinfo, ComboBox whois, PictureBox photo, TextBox adress,TextBox email, NumericUpDown dal, NumericUpDown vz9l)

{

OleDbConnection con = new OleDbConnection("provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";");

OleDbCommand cmd = new OleDbCommand();

if (photo.Image != null)

{

ms = new MemoryStream();

photo.Image.Save(ms, ImageFormat.Jpeg);

photo\_aray = new byte[ms.Length];

ms.Position = 0;

ms.Read(photo\_aray, 0, photo\_aray.Length);

}

cmd.Connection = con;

cmd.CommandText = "Insert into [People] (Имя, Фамилия, Отчество, Дата\_Рождения, Телефон, Адресс, Email, Кем\_Является, Доп\_Информация, Взял\_В\_Займы, Дал\_В\_Займы, Долг, Фото) VALUES (?,?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, @Фото)";

cmd.Parameters.AddWithValue("@Имя", textname.Text);

cmd.Parameters.AddWithValue("@Фамилия", textname2.Text);

cmd.Parameters.AddWithValue("@Отчество", textname3.Text);

cmd.Parameters.AddWithValue("@Дата\_Рождения,", date.Value);

cmd.Parameters.AddWithValue("@Телефон,", phone.Text);

cmd.Parameters.AddWithValue("@Адресс", adress.Text);

cmd.Parameters.AddWithValue("@Email", email.Text);

cmd.Parameters.AddWithValue("@Кем\_Является", whois.SelectedItem.ToString());

cmd.Parameters.AddWithValue("@Доп\_Информация", addinfo.Text);

cmd.Parameters.AddWithValue("@Взял\_В\_Займы", Convert.ToDouble(vz9l.Value));

cmd.Parameters.AddWithValue("@Дал\_В\_Займы", Convert.ToDouble(dal.Value));

cmd.Parameters.AddWithValue("@Долг", calculate(Convert.ToDouble(vz9l.Value),Convert.ToDouble(dal.Value)));

if (photo.Image != null)

{

cmd.Parameters.AddWithValue("@Фото", photo\_aray);

}

else

{

cmd.Parameters.AddWithValue("@Фото", DBNull.Value);

}

con.Open();

cmd.ExecuteNonQuery();

con.Close();

}

//Добавление новой записи

//Чтение из файла с выводом в конкретные поля

public void ReadFromFileEDIT(int ID,TextBox textname, TextBox textname2, TextBox textname3, DateTimePicker date, TextBox phone, TextBox addinfo, ComboBox whois, PictureBox photo, TextBox adress, TextBox email, NumericUpDown dal, NumericUpDown vz9l)

{

OleDbConnection con = new OleDbConnection("provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";");

OleDbCommand cmd = new OleDbCommand();

cmd.Connection = con;

cmd.CommandText = "Select ID,Имя, Фамилия, Отчество, Дата\_Рождения, Телефон, Адресс, Email, Кем\_Является, Доп\_Информация, Взял\_В\_Займы, Дал\_В\_Займы, Фото from People";

con.Open();

OleDbDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

int currentid = Convert.ToInt32(dr["ID"]);

if (currentid == ID)

{

textname.Text = dr["Имя"].ToString();

textname2.Text = dr["Фамилия"].ToString();

textname3.Text = dr["Отчество"].ToString();

date.Value = Convert.ToDateTime(dr["Дата\_Рождения"]);

phone.Text = dr["Телефон"].ToString();

adress.Text = dr["Адресс"].ToString();

email.Text = dr["Email"].ToString();

whois.SelectedIndex = whois.FindStringExact(dr["Кем\_Является"].ToString());

addinfo.Text = dr["Доп\_Информация"].ToString();

vz9l.Value = Convert.ToDecimal(dr["Взял\_В\_Займы"]);

dal.Value =Convert.ToDecimal(dr["Дал\_В\_Займы"]);

photo.Image = null;

if (dr["Фото"] != DBNull.Value)

{

photo\_aray = (byte[])dr["Фото"];

MemoryStream ms = new MemoryStream(photo\_aray);

photo.Image = Image.FromStream(ms);

}

}

}

con.Close();

}

//Чтение из файла с выводом в конкретные поля

//Чтение из файла с выводом в конкретные поля TODO

public void ReadFromFile(int ID, TextBox textname, TextBox textname2, TextBox textname3, TextBox date, TextBox phone, TextBox addinfo, TextBox whois, PictureBox photo, TextBox adress,TextBox email, TextBox dolg)

{

OleDbConnection con = new OleDbConnection("provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";");

OleDbCommand cmd = new OleDbCommand();

cmd.Connection = con;

cmd.CommandText = "Select ID,Имя, Фамилия, Отчество, Дата\_Рождения, Телефон, Адресс, Email, Кем\_Является, Доп\_Информация, Взял\_В\_Займы, Дал\_В\_Займы, Долг, Фото from People";

con.Open();

OleDbDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

int currentid = Convert.ToInt32(dr["ID"]);

if (currentid == ID)

{

textname.Text = dr["Имя"].ToString();

textname2.Text = dr["Фамилия"].ToString();

textname3.Text = dr["Отчество"].ToString();

date.Text = dr["Дата\_Рождения"].ToString();

phone.Text = dr["Телефон"].ToString();

adress.Text = dr["Адресс"].ToString();

email.Text = dr["Email"].ToString();

whois.Text = dr["Кем\_Является"].ToString();

addinfo.Text = dr["Доп\_Информация"].ToString();

dolg.Text = dr["Долг"].ToString(); //calculate(Convert.ToDouble(dr["Взял\_В\_Займы"]),Convert.ToDouble(dr["Дал\_В\_Займы"])).ToString();

photo.Image = null;

if (dr["Фото"] != DBNull.Value)

{

photo\_aray = (byte[])dr["Фото"];

MemoryStream ms = new MemoryStream(photo\_aray);

photo.Image = Image.FromStream(ms);

}

}

}

con.Close();

}

//Чтение из файла с выводом в конкретные поля TODO отредактировать и запихнуть все в 1 функцию

//Простая формула подсчета

private double calculate(double Vz9l, double dal)

{

return Vz9l - dal;

}

//Простая формула подсчета

//Функция изменения записи

public void ChangeLine(int ID,TextBox textname, TextBox textname2, TextBox textname3, DateTimePicker date, TextBox phone, TextBox addinfo, ComboBox whois, PictureBox photo, TextBox adress, TextBox email, NumericUpDown dal, NumericUpDown vz9l)

{

OleDbConnection con = new OleDbConnection("provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";");

OleDbCommand cmd = new OleDbCommand();

if (photo.Image != null)

{

ms = new MemoryStream();

photo.Image.Save(ms, ImageFormat.Jpeg);

photo\_aray = new byte[ms.Length];

ms.Position = 0;

ms.Read(photo\_aray, 0, photo\_aray.Length);

}

cmd.Connection = con;

cmd.CommandText = "UPDATE [People] SET Имя=? , Фамилия=? , Отчество=? , Дата\_Рождения=? , Телефон=? , Адресс=? , Email=? , Кем\_Является=? , Доп\_Информация=? , Взял\_В\_Займы=? , Дал\_В\_Займы=? , Долг=?, Фото=? WHERE ID=?;";

cmd.Parameters.AddWithValue("@Имя", textname.Text);

cmd.Parameters.AddWithValue("@Фамилия", textname2.Text);

cmd.Parameters.AddWithValue("@Отчество", textname3.Text);

cmd.Parameters.AddWithValue("@Дата\_Рождения,", date.Value);

cmd.Parameters.AddWithValue("@Телефон,", phone.Text);

cmd.Parameters.AddWithValue("@Адресс", adress.Text);

cmd.Parameters.AddWithValue("@Email", email.Text);

cmd.Parameters.AddWithValue("@Кем\_Является", whois.SelectedItem.ToString());

cmd.Parameters.AddWithValue("@Доп\_Информация", addinfo.Text);

cmd.Parameters.AddWithValue("@Взял\_В\_Займы", Convert.ToDouble(vz9l.Value));

cmd.Parameters.AddWithValue("@Дал\_В\_Займы", Convert.ToDouble(dal.Value));

cmd.Parameters.AddWithValue("@Долг", calculate(Convert.ToDouble(vz9l.Value),Convert.ToDouble(dal.Value)).ToString());

if (photo.Image != null)

{

cmd.Parameters.AddWithValue("@Фото", photo\_aray);

}

else

{

cmd.Parameters.AddWithValue("@Фото", DBNull.Value);

}

cmd.Parameters.AddWithValue("@ID", ID);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

}

//Функция изменения записи

//Функция удаления записи

public void deleterow(int ID)

{

OleDbConnection con = new OleDbConnection("provider=Microsoft.Jet.OleDb.4.0;Data Source=" + DBpath + ";Jet OLEDB:Database Password=" + DBpass + ";");

OleDbCommand cmd = new OleDbCommand();

cmd.Connection = con;

cmd.CommandText = "DELETE FROM People WHERE ID =?";

con.Open();

cmd.Parameters.AddWithValue("@ID", ID);

cmd.ExecuteNonQuery();

con.Close();

}

//Функция удаления записи

}

}

partial class LoginForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using ADOX;

using System.IO;

namespace Daigorodov\_Kyrcova9

{

public partial class LoginForm : Form

{

public LoginForm()

{

InitializeComponent();

}

protected override void OnFormClosing(FormClosingEventArgs e)

{

Application.Exit();

}

private void Enter\_button\_Click(object sender, EventArgs e)

{

try

{

WorkWithDB.Instance.TryLogin(Login\_textBox.Text, Pass\_textBox.Text);

MainForm mainform = new MainForm();

mainform.Show();

this.Hide();

}

catch

{

MessageBox.Show("Неверный логин или пароль!");

}

}

private void Create\_button\_Click(object sender, EventArgs e)

{

if (Login\_textBox.Text == "")

{

MessageBox.Show("Введите логин (имя файла)");

}

else

{

if (Pass\_textBox.Text == "")

{

DialogResult dialogResult = MessageBox.Show("Вы уверены, что хотите создать новый файл БЕЗ пароля?", "Вы не ввели пароль!", MessageBoxButtons.YesNo);

if (dialogResult == DialogResult.Yes)

{

if (WorkWithDB.Instance.CreateFile(Login\_textBox.Text, Pass\_textBox.Text))

{

WorkWithDB.Instance.TryLogin(Login\_textBox.Text, Pass\_textBox.Text);

MainForm mainform = new MainForm();

mainform.Show();

this.Hide();

}

}

}

else

{

if (WorkWithDB.Instance.CreateFile(Login\_textBox.Text, Pass\_textBox.Text))

{

WorkWithDB.Instance.TryLogin(Login\_textBox.Text, Pass\_textBox.Text);

MainForm mainform = new MainForm();

mainform.Show();

this.Hide();

}

}

}

}

}

}

public partial class MainForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Daigorodov\_Kyrcova9

{

public partial class MainForm : Form

{

int currentid;

public MainForm()

{

InitializeComponent();

this.Text = WorkWithDB.Instance.path();

buttonUTF8WordConvert.Visible = false;

}

protected override void OnFormClosing(FormClosingEventArgs e)

{

LoginForm login = new LoginForm();

login.Show();

}

private void MainForm\_Load(object sender, EventArgs e)

{

WorkWithDB.Instance.LoadFile(dataGridView1);

dataGridView1.Columns[0].Visible = false;

dataGridView1.Columns[4].HeaderText = "Дата рождения";

dataGridView1.Columns[6].HeaderText = "Адрес";

dataGridView1.Columns[8].HeaderText = "Кем является";

}

private void Add\_button\_Click(object sender, EventArgs e)

{

AddPeopleForm AddPeople = new AddPeopleForm();

AddPeople.ShowDialog(this);

WorkWithDB.Instance.LoadFile(dataGridView1);

}

private void Delete\_button\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0 || dataGridView1.SelectedCells.Count > 0)

{

int selectedrowindex = dataGridView1.SelectedCells[0].RowIndex;

DataGridViewRow selectedRow = dataGridView1.Rows[selectedrowindex];

if (selectedRow.Cells["ID"].Value == null)

{

MessageBox.Show("Вы выбрали пустую строку!");

}

else

{

currentid = Convert.ToInt32(selectedRow.Cells["ID"].Value);

DialogResult dialogResult = MessageBox.Show("Вы уверены, что хотите удалить запись?", "Удаление записи", MessageBoxButtons.YesNo);

if (dialogResult == DialogResult.Yes)

{

WorkWithDB.Instance.deleterow(currentid);

WorkWithDB.Instance.LoadFile(dataGridView1);

}

}

}

else MessageBox.Show("Вы не выбрали строку!");

}

private void Edit\_button\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0 || dataGridView1.SelectedCells.Count > 0)

{

int selectedrowindex = dataGridView1.SelectedCells[0].RowIndex;

DataGridViewRow selectedRow = dataGridView1.Rows[selectedrowindex];

if (selectedRow.Cells["ID"].Value == null)

{

MessageBox.Show("Вы выбрали пустую строку!");

}

else

{

currentid = Convert.ToInt32(selectedRow.Cells["ID"].Value);

EditPeopleForm EditPeople = new EditPeopleForm(currentid);

EditPeople.ShowDialog(this);

WorkWithDB.Instance.LoadFile(dataGridView1);

}

}

else MessageBox.Show("Вы не выбрали строку!");

}

private void Search\_button\_Click(object sender, EventArgs e)

{

SearchForm Search = new SearchForm();

Search.ShowDialog(this);

if (Search.DialogResult == DialogResult.OK) WorkWithDB.Instance.LoadFileFiltered(dataGridView1, SearchForm.namecheck, SearchForm.name, SearchForm.lastnamecheck, SearchForm.lastname,SearchForm.whocheck,SearchForm.who);

}

private void MoreInfo\_button\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0 || dataGridView1.SelectedCells.Count > 0)

{

int selectedrowindex = dataGridView1.SelectedCells[0].RowIndex;

DataGridViewRow selectedRow = dataGridView1.Rows[selectedrowindex];

if (selectedRow.Cells["ID"].Value == null)

{

MessageBox.Show("Вы выбрали пустую строку!");

}

else

{

currentid = Convert.ToInt32(selectedRow.Cells["ID"].Value);

MoreIngoForm moreinfo = new MoreIngoForm(currentid);

moreinfo.ShowDialog(this);

}

}

else MessageBox.Show("Вы не выбрали строку!");

}

private void button1\_Click(object sender, EventArgs e)

{

WorkWithDB.Instance.LoadFile(dataGridView1);

}

private void button2\_Click(object sender, EventArgs e)

{

fontDialog1.ShowColor = true;

fontDialog1.ShowDialog();

foreach (DataGridViewColumn c in dataGridView1.Columns)

{

c.DefaultCellStyle.Font = fontDialog1.Font;

c.DefaultCellStyle.ForeColor = fontDialog1.Color;

}

}

//TODO Доделать вывод в WORD

private void ToCsV(DataGridView dGV, string filename)

{

string stOutput = "";

// Export titles:

string sHeaders = "";

for (int j = 0; j < dGV.Columns.Count; j++)

sHeaders = sHeaders.ToString() + Convert.ToString(dGV.Columns[j].HeaderText) + "\t";

stOutput += sHeaders + "\r\n";

// Export data.

for (int i = 0; i < dGV.RowCount - 1; i++)

{

string stLine = "";

for (int j = 0; j < dGV.Rows[i].Cells.Count; j++)

stLine = stLine.ToString() + Convert.ToString(dGV.Rows[i].Cells[j].Value) + "\t";

stOutput += stLine + "\r\n";

}

Encoding utf16 = new UTF8Encoding(true, true);

byte[] output = utf16.GetBytes(stOutput);

FileStream fs = new FileStream(filename, FileMode.Create);

BinaryWriter bw = new BinaryWriter(fs);

bw.Write(output, 0, output.Length); //write the encoded file

bw.Flush();

bw.Close();

fs.Close();

}

private void button3\_Click(object sender, EventArgs e)

{

SaveFileDialog sfd = new SaveFileDialog();

sfd.Filter = "Word Documents (\*.doc)|\*.doc";

sfd.FileName = "export.doc";

if (sfd.ShowDialog() == DialogResult.OK)

{

ToCsV(dataGridView1, sfd.FileName); // Here dataGridview1 is your grid view name

}

}

}

}

partial class AddPeopleForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Daigorodov\_Kyrcova9

{

public partial class AddPeopleForm : Form

{

public AddPeopleForm()

{

InitializeComponent();

dateTimePickerBirthday.Format = DateTimePickerFormat.Custom;

dateTimePickerBirthday.CustomFormat = "yyyy.MM.dd";

}

private void AddPeopleForm\_Load(object sender, EventArgs e)

{

dateTimePickerBirthday.Value = dateTimePickerBirthday.Value;

comboBoxWho.SelectedIndex = comboBoxWho.FindStringExact("Не выбрано");

}

private void AddPhoto\_Click(object sender, EventArgs e)

{

OpenFileDialog openDialog = new OpenFileDialog();

openDialog.Multiselect = false;

openDialog.Filter = "Image Files (\*.bmp, \*.jpg)|\*.bmp;\*.jpg|All Files (\*.\*)|\*.\*";

if (openDialog.ShowDialog(this) == DialogResult.OK)

{

try

{

pictureBoxPhoto.Image = Image.FromFile(openDialog.FileName);

}

catch (Exception)

{

MessageBox.Show("Это не изображение");

}

}

}

private void buttonAddPeople\_Click(object sender, EventArgs e)

{

string sPattern1 = "@";

string sPattern2 = ".ru";

string sPattern3 = ".com";

string sPattern4 = ".net";

if (textBox\_Email.Text != "")

if (System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern1, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

&& (System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern2, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

|| System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern3, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

|| System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern4, System.Text.RegularExpressions.RegexOptions.IgnoreCase)))

{

WorkWithDB.Instance.AddNewLine(textBoxName, textBoxLastName, textBoxOtshestvo, dateTimePickerBirthday, textBoxPhone, textBoxInfo, comboBoxWho, pictureBoxPhoto, textBoxAdress, textBox\_Email, numericUpDownCreditGive, numericUpDownCreditTake);

this.Close();

}

else MessageBox.Show("Email введен не верно");

else

{

WorkWithDB.Instance.AddNewLine(textBoxName, textBoxLastName, textBoxOtshestvo, dateTimePickerBirthday, textBoxPhone, textBoxInfo, comboBoxWho, pictureBoxPhoto, textBoxAdress, textBox\_Email, numericUpDownCreditGive, numericUpDownCreditTake);

this.Close();

}

}

private void buttonCloseForm\_Click(object sender, EventArgs e)

{

this.Close();

}

private void buttonCleanPhoto\_Click(object sender, EventArgs e)

{

pictureBoxPhoto.Image = null;

}

}

}

partial class EditPeopleForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Daigorodov\_Kyrcova9

{

public partial class EditPeopleForm : Form

{

int CurrentId;

public EditPeopleForm(int id)

{

InitializeComponent();

CurrentId=id;

}

private void EditPeopleForm\_Load(object sender, EventArgs e)

{

dateTimePickerBirthday.Value = DateTime.Now;

WorkWithDB.Instance.ReadFromFileEDIT(CurrentId,textBoxName,textBoxLastName,textBoxOtshestvo,dateTimePickerBirthday,textBoxPhone,textBoxInfo,comboBoxWho,pictureBoxPhoto,textBoxAdress,textBox\_Email,numericUpDownCreditGive,numericUpDownCreditTake);

}

private void buttonAddPeople\_Click(object sender, EventArgs e)

{

string sPattern1 = "@";

string sPattern2 = ".ru";

string sPattern3 = ".com";

string sPattern4 = ".net";

if (textBox\_Email.Text != "")

if (System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern1, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

&& (System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern2, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

|| System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern3, System.Text.RegularExpressions.RegexOptions.IgnoreCase)

|| System.Text.RegularExpressions.Regex.IsMatch(textBox\_Email.Text, sPattern4, System.Text.RegularExpressions.RegexOptions.IgnoreCase)))

{

WorkWithDB.Instance.ChangeLine(CurrentId, textBoxName, textBoxLastName, textBoxOtshestvo, dateTimePickerBirthday, textBoxPhone, textBoxInfo, comboBoxWho, pictureBoxPhoto, textBoxAdress, textBox\_Email, numericUpDownCreditGive, numericUpDownCreditTake);

this.Close();

}

else MessageBox.Show("Email введен не верно");

else

{

WorkWithDB.Instance.ChangeLine(CurrentId, textBoxName, textBoxLastName, textBoxOtshestvo, dateTimePickerBirthday, textBoxPhone, textBoxInfo, comboBoxWho, pictureBoxPhoto, textBoxAdress, textBox\_Email, numericUpDownCreditGive, numericUpDownCreditTake);

this.Close();

}

}

private void AddPhoto\_Click(object sender, EventArgs e)

{

OpenFileDialog openDialog = new OpenFileDialog();

openDialog.Multiselect = false;

openDialog.Filter = "Image Files (\*.bmp, \*.jpg)|\*.bmp;\*.jpg|All Files (\*.\*)|\*.\*";

if (openDialog.ShowDialog(this) == DialogResult.OK)

{

try

{

pictureBoxPhoto.Image = Image.FromFile(openDialog.FileName);

}

catch (Exception)

{

MessageBox.Show("Это не изображение");

}

}

}

private void buttonCleanPhoto\_Click(object sender, EventArgs e)

{

pictureBoxPhoto.Image = null;

}

private void buttonCloseForm\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

partial class MoreIngoForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Daigorodov\_Kyrcova9

{

public partial class MoreIngoForm : Form

{

int currentid;

public MoreIngoForm(int ID)

{

InitializeComponent();

currentid = ID;

}

private void buttonCloseForm\_Click(object sender, EventArgs e)

{

this.Close();

}

private void MoreIngoForm\_Load(object sender, EventArgs e)

{

WorkWithDB.Instance.ReadFromFile(currentid, textBoxName, textBoxLastName, textBoxOtshestvo, textBoxDate, textBoxPhone,textBoxInfo, textBoxWho, pictureBoxPhoto,textBoxAdress,textbox\_Email,textBoxDolg);

}

}

}

partial class SearchForm : Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Daigorodov\_Kyrcova9

{

public partial class SearchForm : Form

{

public static bool namecheck;

public static bool lastnamecheck;

public static bool whocheck;

public static string name;

public static string lastname;

public static string who;

static readonly SearchForm instance = new SearchForm();

public static SearchForm Instance

{

get

{

return instance;

}

}

public SearchForm()

{

InitializeComponent();

}

private void buttonOk\_Click(object sender, EventArgs e)

{

name = textBoxName.Text;

lastname = textBoxLastName.Text;

namecheck = checkBoxName.Checked;

lastnamecheck = checkBoxLastName.Checked;

whocheck = checkBoxWho.Checked;

who = comboBoxWho.Text;

this.DialogResult = DialogResult.OK;

}

private void SearchForm\_Load(object sender, EventArgs e)

{

comboBoxWho.SelectedIndex = comboBoxWho.FindStringExact("Не выбрано");

}

}

}

# ЛИСТ ЗАМЕЧАНИЙ ПО ПОЯСНИТЕЛЬНОЙ ЗАПИСКЕ КУРСОВОГО ПРОЕКТА

|  |  |  |
| --- | --- | --- |
|  | Замечания консультантов | Подпись |
| 1. Замечания  по содержанию теоретической части |  |  |
| 1. Замечания  по содержанию практической части |  |  |
| 1. Замечания по оформлению |  |  |