Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Курсова робота з дисципліни

**“**Основи об’єктно-орієнтованого програмування**”**

Тема: Програмна система під назвою «Записная книжка»

Виконавець: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Г. Мінаков

ст. гр. ПІ-13-1

Керівник: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Л.Д. Самофалов

доцент

Комісія:

проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.М. Бондарєв

зав. каф. ПI, проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ З.В. Дудар

Робота захищена з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_

Харків 2014

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 25 с., 12 рис.

Мета роботи – створення програмної системи під назвою «Записная книжка» для роботи з інформаційною базою даних людей.

Метод розробки – попереднє проектування та наступна реалізація програмної системи за допомогою мови програмування C# з використанням NET Framework 4.5. Розробка виконувалась у середі програмування Microsoft Visual Studio 2012.

В результаті розробки отримана система під назвою «Записна книжка» для роботи з інформаційною базою даних людей, роль якої виконують текстові файли. Система передбачає можливість перегляду та зміни інформації стосовно людей: анкетні дані, адреси, телефони, місце роботи або навчання, посада знайомих, колег і родичів, характер знайомства, ділові якості.

КУРСОВА РОБОТА, ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, WINDOWS FORMS, MICROSOFT VISUAL STUDIO 2012, NET FRAMEWORK.

СОДЕРЖАНИЕ

ВСТУПЛЕНИЕ

1 СПЕЦИФИКАЦИЯ ПРОГРАММЫ

1.1 Основные требования и функции программы

1.2 Пользовательский интерфейс

2 ПРОЕКТНАЯ СПЕЦИФИКАЦИЯ

2.1 Объектная модель программы

2.2 Реализация функций программы

3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

ВЫВОДЫ

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

ВСТУПЛЕНИЕ

В данной пояснительной записке рассматривается описание программы «Записная книжка». Целью данного курсового проекта является автоматизация обработки анкетних данных на основе объектно-ориентированного подхода.

Задачи выполнения работы:

* исследование предметной области с целью выявления основных сущностей и их атрибутов;
* проектирование иерархии классов и интерфейсов на основе выделенных сущностей;
* применение принципа инкапсуляции к классам;
* временный отказ от БД в пользу использования текстовых документов.

При объектно-ориентированном подходе программа представляет собой описание объектов, их свойств (или атрибутов), совокупностей (или классов), отношений между ними, способов их взаимодействия и операций над объектами (или методов). Объектно-ориентированный подход требует глубокого понимания основных принципов, или, иначе, концепций, на которых он базируется. К числу основополагающих понятий ООП обычно относят абстракцию данных, наследование, инкапсуляцию и полиморфизм.

Объектно-ориентированное программирование в настоящее время является абсолютным лидером в области прикладного программирования.

В качестве основного инструмента разработки применяется Microsoft Visual Studio 2013. Язык программирования C#.

1 СПЕЦИФИКАЦИЯ ПРОГРАММЫ

1.1 Основные требования и функции программы

В данной курсовой работе следует разработать систему, которая смогла бы работать с анкетными данными, изменять и упорядочивать их, а также формировать поздравления с днем рождения(по текущей дате).

В результате разработки будет получена программа, в которой будет предвидена возможность просмотра сведений о людях, например, таких: анкетные данные, адреса, телефоны, место работы или учебы, должность знакомых, коллег и родственников, характер знакомства, деловые качества.

Для разработки выбрана Microsoft Visual Studio 2012 Ultimate как среда разработки, которая наиболее точно соответствует современным требованиям конструирования программного обеспечения и обладает наиболее удобным интерфейсом. В качестве базы данных следует использовать представление данных в виде текстового файла с перечнем анкетных данных.

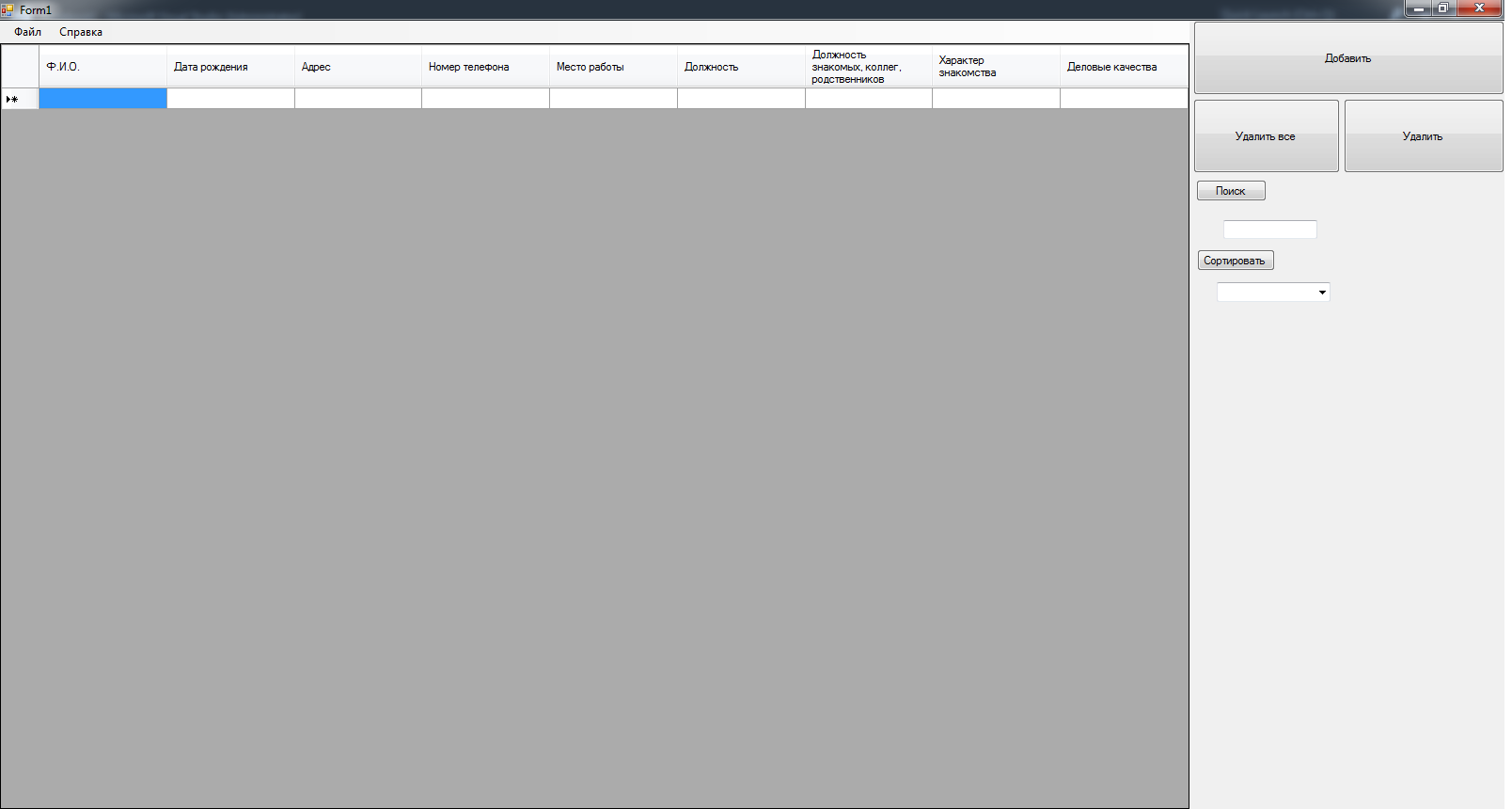
Из главного окна должна быть предусмотрена возможность быстрого поиска по произвольному шаблону.

Пользователь должен иметь возможность добавления информации о человеке, а также удаления ненужной информации из текстового файла.

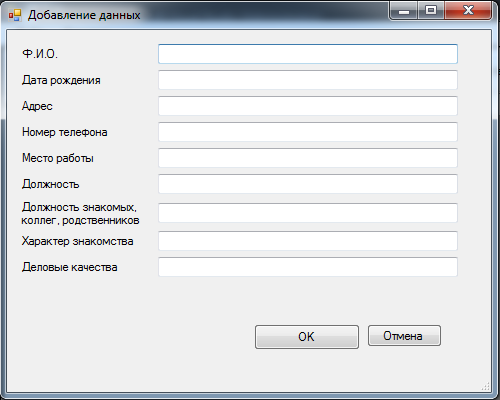
1.2 Пользовательский интерфейс

Весь пользовательский интерфейс разделен на 4 формы.

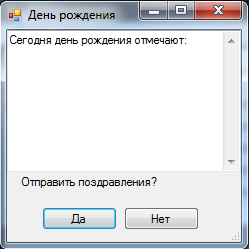
Рассмотрим главную форму, на которой пользователь сможет выполнять основные функции программы. Главная форма содержит 5 кнопок и панель управления.



Первая кнопка – «Добавить» вызывает форму добавления информации о человеке.



Также программа может формировать поздравления с днем рождения(по текущей дате).



Под кнопкой «Добавить» находятся две кнопки – «Удалить все», которая полностью очищает таблицу и «Удалить», которая удаляет выбранную строку.

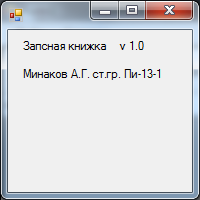
Если пользователь ввел информацию в текстовое поле поиска и нажал кнопку «Поиск», то в таблице будут выделены те люди, которые совпадают с требованиями поиска.

Также пользователь может сортировать данные. Для этого необходимо выбрать в списке необходимый пункт и нажать на кнопку «Сортировать».

Перейдем теперь к панели управления. Там располагается 2 кнопки: «Файл» и «Справка».

Рассмотрим кнопку «Файл». Здесь мы можем выбрать один из трех пунктов: загрузить файл (кнопка «Открыть»), сохранить данные (кнопка «Сохранить») или же закрыть программу (кнопка «Закрыть»).

Вторая кнопка в панели управления – «Справка». Она вызывает диалоговое окно, в котором указано название программы и автор.



2 ПРОЕКТНАЯ СПЕЦИФИКАЦИЯ

2.1 Объектная модель программы

В ООП главным элементом является класс, включающий множество объектов с одинаковыми свойствами, операциями и отношениями. В данной программе реализованы классы:

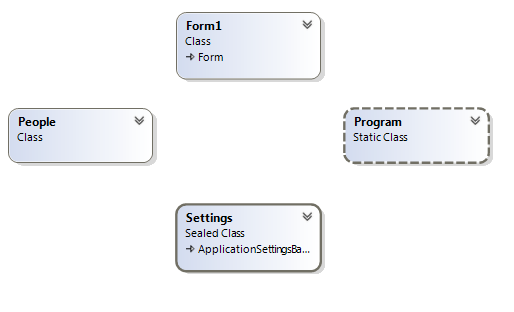


Рисунок 2.1 – Объектная спецификация программы

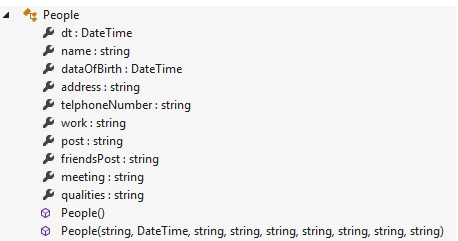


Рисунок 2.2 – Поля, свойства и конструкторы в классе People

Класс People содержит сведения о человеке: анкетные данные, адреса, телефоны, место работы или учебы, должность знакомых, коллег и родственников, характер знакомства, деловые качества.

public class People

{

public DateTime dt { get; set; }

public string name { get; set; }

public DateTime dataOfBirth { get; set; }

public string address { get; set; }

public string telphoneNumber { get; set; }

public string work { get; set; }

public string post { get; set; }

public string friendsPost { get; set; }

public string meeting { get; set; }

public string qualities { get; set; }

public People() { }

public People(string \_name, DateTime \_dataOfBirth, string \_address, string \_telephoneNumber, string \_work, string \_post,

string \_friendsPost, string \_meeting, string \_qualities)

{

this.dt = DateTime.Now;

this.name = \_name;

this.dataOfBirth = \_dataOfBirth;

this.address = \_address;

this.telphoneNumber = \_telephoneNumber;

this.work = \_work;

this.post = \_post;

this.friendsPost = \_friendsPost;

this.meeting = \_meeting;

this.qualities = \_qualities;

}

}

Отображение и обработка данных происходит в таблице DataGridView.

При нажатии на кнопку «Добавить» появляется форма добавления человека. В которой при нажатии кнопки «OK» вызывается следующая функция:

private void OK\_btn\_Click(object sender, EventArgs e)

{

bool validation = true;

DateTime dataOfBirth = default(DateTime);

//Валидация даты рождения

try

{

dataOfBirth = DateTime.Parse(DataOfBirth\_textBox.Text);

}

catch (Exception ex)

{

validation = false;

MessageBox.Show("Неправильный формат даты рождения

(ДД.ММ.ГГГГ)!!!");

}

var error = ErrorBox\_label;

string name = Name\_textBox.Text;

string address = Address\_textBox.Text;

string telphoneNumber = TelephoneNumber\_textBox.Text;

string work = Work\_textBox.Text;

string post = Post\_textBox.Text;

string friendsPost = FriendsPost\_textBox.Text;

string meeting = Meeting\_textBox.Text;

string qualities = Qualities\_textBox.Text;

//Валидация введенных данных

if (string.IsNullOrEmpty(name))

{

error.Text = "Не заполнено поле имени ! \n";

validation = false;

}

else if (string.IsNullOrEmpty(dataOfBirth.ToString()))

{

error.Text = "Не заполнено поле даты рождения! \n";

validation = false; }

else if (string.IsNullOrEmpty(address))

{

error.Text = "Не заполнено поле адреса! \n";

validation = false;

}

else if (string.IsNullOrEmpty(telphoneNumber))

{

error.Text = "Не заполнено поле телефонного номера! \n";

validation = false;

}

else if (string.IsNullOrEmpty(work))

{

error.Text = "Не заполнено поле места работы/учебы! \n";

validation = false;

}

else if (string.IsNullOrEmpty(post))

{

error.Text = "Не заполнено поле должность! \n";

validation = false;

}

else if (string.IsNullOrEmpty(friendsPost))

{

error.Text = "Не заполнено поле должность знакомых! \n";

validation = false;

}

else if (string.IsNullOrEmpty(meeting)){

error.Text = "Не заполнено поле характер знакомства! \n";

validation = false;

}

else if (string.IsNullOrEmpty(qualities))

{

error.Text = "Не заполнено поле деловые качества! \n";

validation = false;

}

//Запись введенных данных

if (validation)

{

this.DialogResult = DialogResult.OK;

var people = new People(name, dataOfBirth, address,

telphoneNumber, work, post, friendsPost,

meeting, qualities);

mainForm.AddValuesToDatagrid(people);

//Очистка полей

foreach (var item in this.Controls)

{

if (item is TextBox)

{

var x = item as TextBox;

x.Text = "";

}

}

}

}

Удаление записей происходит по нажатию кнопок «Удалить все» и «Удалить» которые вызывают функции «button2\_Click» и «ClearSelect\_button\_Click» соответственно:

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

{

dataGridView1.Rows.Clear();//Полная очистка полей в dataGridView

}

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

{

try

{

//Удаление выбранной строки

if (dataGridView1.SelectedCells != null)

{

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

dataGridView1.Rows.RemoveAt(ind);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

Поиск по произвольному шаблону выполняется следующим образом: после нажатия кнопки «Поиск», активируется функция «Find\_button\_Click»:

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

{

//Поиск данных в dataGridView

for (int i = 0; i < dataGridView1.RowCount; i++)

{

dataGridView1.Rows[i].Selected = false;

for (int j = 0; j < dataGridView1.ColumnCount; j++)

if (dataGridView1.Rows[i].Cells[j].Value != null)

if(dataGridView1.Rows[i].Cells[j].

Value.ToString().Contains(Find\_textBox.Text))

{

dataGridView1.Rows[i].Selected = true;

break;

}

}

}

Сортировка выполняется следующим образом: необходимо выбрать критерий и после нажатия кнопки «Сортировать», активируется функция «Sort\_button\_Click»:

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

{

//Сортировка данных в dataGridView

switch (Sort\_comboBox.Text)

{

case "Ф.И.О.":

dataGridView1.Sort(dataGridView1.Columns[0],

ListSortDirection.Ascending);

break;

case "Дата рождения":

dataGridView1.Sort(dataGridView1.Columns[1],

ListSortDirection.Ascending);

break;

case "Адрес":

dataGridView1.Sort(dataGridView1.Columns[2],

ListSortDirection.Ascending);

break;

case "Номер телефона":

dataGridView1.Sort(dataGridView1.Columns[3],

ListSortDirection.Ascending);

break;

case "Место работы":

dataGridView1.Sort(dataGridView1.Columns[4],

ListSortDirection.Ascending);

break;

case "Должность":

dataGridView1.Sort(dataGridView1.Columns[5],

ListSortDirection.Ascending);

break;

case "Должность знакомых":

dataGridView1.Sort(dataGridView1.Columns[6],

ListSortDirection.Ascending);

break;

case "Характер знакомства":

dataGridView1.Sort(dataGridView1.Columns[7],

ListSortDirection.Ascending);

break;

case "Деловые качества":

dataGridView1.Sort(dataGridView1.Columns[8],

ListSortDirection.Ascending);

break;

case "Дата последней корректировки":

dataGridView1.Sort(dataGridView1.Columns[9],

ListSortDirection.Ascending);

break;

}

}

Другие методы, используемые при работе приложения:

а) считывание данных и их запись в таблицу:

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

{

Stream mystr = null;

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((mystr = openFileDialog1.OpenFile()) != null)

{

//Открытие потока чтения

StreamReader myread = new StreamReader(mystr);

string[] str;

int num = 0;

try

{

//Запись данных в dataGridView

string[] str1 = myread.ReadToEnd().Split('\n');

num = str1.Count();

dataGridView1.RowCount = num;

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

{

str = str1[i].Split('^');

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

string data = str[j].Replace("[abv]", "^");

dataGridView1.Rows[i].Cells[j].Value = data;

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

myread.Close();//Закрытие потока чтения

CheckDateBirthday();

}

}

}

}

б) запись данных в файл:

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

{

Stream myStream;

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((myStream = saveFileDialog1.OpenFile()) != null)

{

//Открытие потока записи

StreamWriter myWritet = new StreamWriter(myStream);

try

{

//запись данных из dataGridView в

for (int i =0; i < dataGridView1.RowCount-1; i++)

{

for(int j=0; j<dataGridView1.ColumnCount; j++)

{

string data =dataGridView1.Rows[i].Cells[j].

Value.ToString().Replace("^", "[abv]");

myWritet.Write(data + '^');

}

myWritet.WriteLine();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

myWritet.Close();//Закрытие потока записи

}

myStream.Close();

}

}

}

Добавление данных в dataGridView реализовано в функции AddValuesToDatagrid

public void AddValuesToDatagrid(People people)

{

int index = dataGridView1.Rows.Count - 1;

dataGridView1.Rows.Add();//Добавление новой строки в dataGridView

dataGridView1[0, index].Value = people.name;

dataGridView1[1, index].Value = people.dataOfBirth;

dataGridView1[2, index].Value = people.address;

dataGridView1[3, index].Value = people.telphoneNumber;

dataGridView1[4, index].Value = people.work;

dataGridView1[5, index].Value = people.post;

dataGridView1[6, index].Value = people.friendsPost;

dataGridView1[7, index].Value = people.meeting;

dataGridView1[8, index].Value = people.qualities;

dataGridView1[9, index].Value = people.dt;

dataGridView1.Update(); //Обновление данных в dataGridView

}

Также программа способна формировать поздравление с днем рождения(по текущей дате). Это реализовано в функции CheckDateBirthday

public void CheckDateBirthday()

{

DateTime dt = DateTime.Today;

bool check = false;

DateTime dateBirth;

StringBuilder resString = new StringBuilder();

int count = dataGridView1.Rows.Count;

int countBirth = 0;

string[] nameBirth = new string[10];

resString.Append("Сегодня день рождения отмечают: \r\n");

try

{

//Поиск людей у которых сегодня день рождения

for (int i = 0; i < count - 1; i++)

{

string data =

dataGridView1.Rows[i].Cells[1].Value.ToString();

dateBirth = DateTime.Parse(data);

if (dateBirth.Month == dt.Month &&

dateBirth.Day == dt.Day)

{

resString.Append(dataGridView1.Rows[i].

Cells[0].Value.ToString() + "\r\n");

check = true;

//Добавление имен именинников

nameBirth[countBirth] =

dataGridView1.Rows[i].Cells[0].Value.ToString();

countBirth++;

}

}

if (check)

{

//Обработка выбора пользователя

BirthForm birthForm = new BirthForm(resString);

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

{

SendBirthMes(countBirth, nameBirth);

MessageBox.Show("Поздравления отправлены!");

}

else

{

MessageBox.Show("Поздравления отправлены не

были!");

}

}

}

catch (Exception ex){

MessageBox.Show(ex.Message);

}

}

Запись поздравлений в файлы происходит в функции SendBirthMes

public void SendBirthMes(int countBirth, string[] nameBirth)

{

for (int j = 0; j < countBirth; j++)

{

string text = "Дорогой " + nameBirth[j] + ", позравляю вас с

Днём рождения!!!";

string file = @"C:\1\" + nameBirth[j] + ".txt";

//Запись поздравления в файл

System.IO.File.WriteAllText(file, text);

}

}

3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Чтобы начать работать с программой, переместите из папки PrisonerList на свой компьютер само приложение PrisonerList.exe и папку Data. Запустите PrisonerList.exe. У вас на компьютере должен быть уставлен Microsoft .NET Framework 4.5. Если Framework не установлен или устарел, установите или обновите .NET Framework.

После запуска приложения пользователю откроется главное окно программы (рис 3.1).

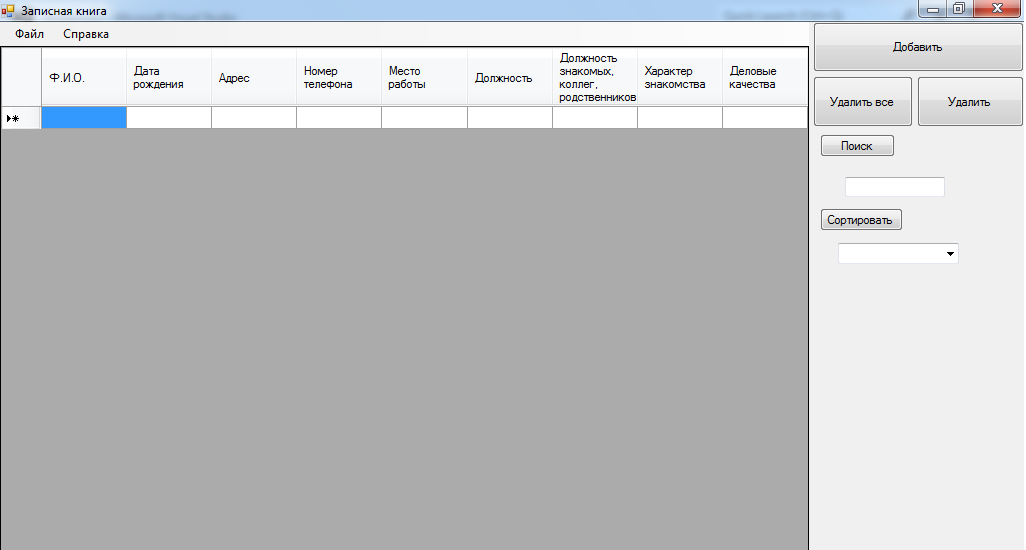


Рисунок 3.1 – Главное окно программы

Пользователь программы может воспользоваться автоматически загруженными базами или загрузить базу, выбрав меню Файл → Загрузить.

При нажатии кнопки «Добавить» (рис.3.1) пользователь может добавить нового человека в базу. Тогда перед ним откроется окно ввода информации (рис. 3.2). При нажатии на кнопку «ОК» в базу добавляется новый элемент.

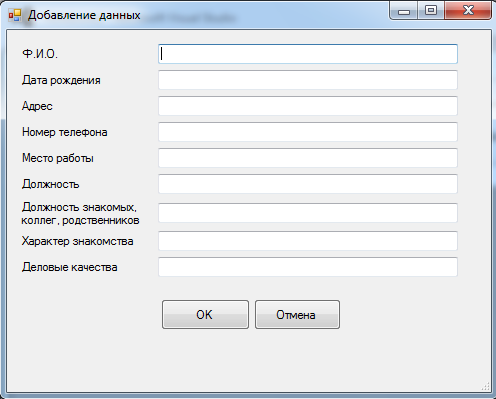


Рисунок 3.2 – Окно добавления человека в базу

Удаление данных происходит при нажатии на кнопку «Удалить» или «Удалить все». Также пользователю нужно подтвердить удаление объекта (рис 3.3)

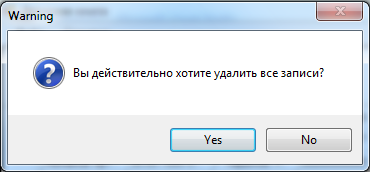


Рисунок 3.3 – Подтверждение удаления объекта

Поиск по произвольному шаблону осуществляется путем введения информации (буквы, цифры) в строку поиска.

При закрытии окна программы, будет выведено окно подтверждения закрытия(рис 3.4).

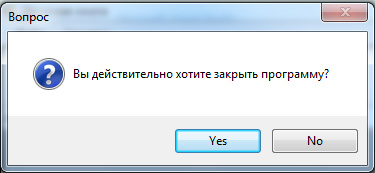


Рисунок 3.4 – Подтверждение закрытия программы

Сохранение текущей активной базы осуществляется выбором меню Файл→ Сохранить.

Информацию о программе, ее авторе можно просмотреть, выбрав меню Справка→ Справка (рис 3.5)

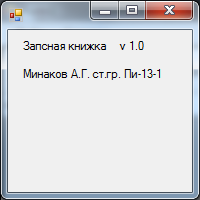


Рисунок 3.5 – Окно «Справка»

ВЫВОДЫ

В результате разработки курсового проекта было создано программное средство под названием «Записная книжка», предназначенное для облегчения и систематизации работы с базами анкетных данных людей, которое позволяет пользователю содержать информацию о заключенных в электронном виде. В проекте реализованы такие базовые задачи, как добавление и удаление информации о людях, поиск данных, сохранение текущих данных либо же загрузка уже существующих баз данных. Также программа способна формировать поздравление с днем рождения (по текущей дате).

В проекте представлен объектно-ориентированный подход в проектировании программного обеспечения информационного плана, дающий возможность на ранних этапах разработки учесть все нюансы будущей программы, необходимый набор функций, состав и структуру баз данных, что в дальнейшем исключает необходимость переработки уже написанных компонентов программы.

Программа реализована полноценно с удобным и интуитивно понятным интерфейсом. Благодаря понятной архитектуре и несложному управлению работать программой может и неподготовленный пользователь, для него доступно руководство по использованию программы.

Программа может быть дополнена новыми функциями, такими как: создание отчета о составе записной книжки, , расчёта потенциальной опасности заключенного и т.д. Возможно улучшение пользовательского интерфейса, осуществление работы приложения со стандартами баз данных.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дейтел Х., Дейтел П. Как программировать на С++. Третье издание.-М.:ЗАО „Издательство БИНОМ”, 2001г.-1152с.
2. Эндрю Троельсен. C# и платформа .NET. Библиотека программиста. – СПб.: Питер, 2005. – 796 с.
3. Бондарев В.М., Объектно-ориентированное программирование на С#. Учебное пособие. – Харьков: СМИТ, 2009г. – 224 с.
4. Бондарев В.М., Программирование на С++. Учебное пособие. – Харьков: СМИТ, 2004г. – 294 с.
5. Герберт Шилдт - C# 4.0. Полное руководство