Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни “ Об’єктно - орієнтоване програмування”

«Сбербанк»

Керівник , асистент каф ПІ Ляпота В.М.

Студент гр. ПІ-15-2 Батюченко В.А.

Комісія: к.т.н., професор\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дудар З.В.

к.т.н. професор\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В.М.

Харків 2016

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

Кафедра: ***Програмної інженерії***

Дисципліна: ***Об’єктно-орієнтоване програмування***

Спеціальність: ***Програмна інженерія***

Курс 1 .Група *ПІ-15-2* . Семестр *2* .

***ЗАВДАННЯ***

***на курсовий проект студента***

*Батюченка Віталія Андрійоаича*

1. Тема проекту: Сбербанк.
2. Термін здачі студентом закінченого проекту: *“28” - травня - 2016 р*.
3. Вихідні дані до проекту: *специфікація програми, методичні вказівки до виконання курсової роботи*.
4. Зміст розрахунково-пояснювальної записки: *вступ, програмна специфікація, проектна специфікація,* інструкція користувача (пояснює, як встановлювати програму та користуватися нею), висновки.
5. Перелік графічного матеріалу: екранні форми, ілюстрація взаємодії класів.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми | 1-03-2016 р. |
| 2 | Формулювання вимог до програми | 2-03-2016 – 12-03-2016 р. |
| 3 | Розробка підсистеми реєстрації. | 13-03-2016 – 24-03-2016 р. |
| 4 | Розробка підсистеми, зберігання та пошуку вкладник. | 25-03-2016 – 1-04-2016 р. |
| 5 | Розробка функцій видача грошей та поповнення рахунку. | 2-04-2016 – 9-04-2016 р. |
| 6 | Розробка функцій зберігання та завантаження даних | 10-04-2016 – 16-04-2016 р. |
| 7 | Тестування і доопрацювання розробленої програмної системи. | 17-04-2016 – 29-04-2016 р. |
| 8 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 30-04-2016 – 28-05-2016 р. |
| 9 | Захист | 28-05-2016 р. |

Студент Батюченко Віталій Андрійович

Керівник Ляпота Віталій Миколайович

« 1 »\_березня\_ 2016 р.

РЕФЕРАТ

Пояснительная записка к курсовой работе: 27 с., 27 рис., 1 додаток, 5 источника.

Цель курсового проекта – создание программы «Сбербанк», позволяющей работать со вкладчиками.

Метод разработки - среда разработки Microsoft Visual Studio 2014, Windows Forms, платформа .NET Framework 4.5[3], язык программирования C#.

В результате получена программа под названием «Сбербанк», которая позволяет осуществлять работу со вкладчиками.

Программа производит такие операции как: добавление вкладчика. Возможен поиск вкладчика по лицевому счёту или паспортными данными, выдача денег, пополнение счёта, начисление процентов каждый день и реализовано сохранение в txt-файл.

КУРСОВОЙ ПРОЕКТ, ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАМИРОВАНИЕ, ЯЗЫК ПРОГРАММИРОВАНИЯ C#, СБЕРБАНК.

СОДЕРЖАНИЕ

Введение 6

1 Спецификация программы 7

* 1. Постановка задачи 7
  2. Функции программы 7
  3. Описание пользовательского интерфейса 8

2 Проектная спецификация 11

2.1 Объектная модель программы 11

2.1.1 Класс depositore 11

2.1.2 Класс Work\_File 12

3 Руководство пользователя 13

3.1 Установка программы 13

3.2 Работа с программой 13

3.2.1 Добавление вкладчика 14

3.2.2 Поиск и операции над вкладчиком 14

3.2.3 Выход 17

Выводы 18

Перечень ссылок 19

Приложение А – Фрагменты кода программы 20

ВВЕДЕНИЕ

В этой работе рассмотрено приложение для сохранения большого перечня вкладчиков. Этот проект написан с помощью объектно-ориентированного программирования на языке C# с использованием учебной литературы.

В курсовом проекте создано приложение, с помощью которого, можем просматривать базу вкладчиков: их персональные данные, сумму вложения, текущую сумму денег, а также возможность сохранять данные вкладчиков в файл.

В наше время большинство людей пытается заменить работу с бумажными носителями заменить на работу с компьютером. То же самое происходит в банке. Удобнее хранить вкладчиков в компьютере, и искать их за 1 секунду по 1 щелчку мышки. Так же это еще 1 способ хранения данных, если уничтожится одна база данных то её можно восстановить с помощью другой. В работе со вкладчиком бывает играет человеческий фактор и человек может ошибиться при начислении процентов, но в случаи с компьютерной версией такое не может произойти.

Эта программа облегчит любому кассиру жизнь, экономя время для поиска того или иного вкладчика в журнале. Это приложение просто в использовании, в нем разберется любой пользователь. Интерфейс довольно прост для привлечения еще большей аудитории.

Использование объектно-ориентированного подхода при создании такого рода приложений упростит процесс разработки приложения и позволит в дальнейшем его расширять и совершенствовать, не прибегая к существенным изменениям в исходном коде.

1СПЕЦИФИКАЦИЯ ПРОГРАММЫ

* 1. Постановка задачи

В курсовой работе необходимо создать программу «Сбербанк».

В программе должна храниться информация о вкладчиках, а также их сумма вложения.

В справочнике будут доступны следующие функции: добавление вкладчика, сохранения вкладчиков в txt-файл по текущей дате . Возможен поиск по лицевому счёту или паспортным дынным. Можно снимать деньги и пополнять текущий баланс.

Также немало важная часть приложения – это удобный и простой интерфейс для пользователя. Если он будет не понятен для окружающих, то это может негативно отразиться на программе, либо пользователь будет долго разбираться с программой. Следовательно интерфейс должен быть понятен, что бы пользователь смог пользоваться всеми функциями этого приложения.

1.2 Функции программы

Функции программы:

1. Поиск вкладчиков (рис.1.1)..
2. Добавление вкладчиков (рис.1.10).
3. Операции с вкладчиками (рис.1.13).
4. Получение процентов по завершению дня (рис.1.16).
5. Сохранение вкладчиков в текстовый файл по завершению дня (рис.1.21).
6. Сохранение изменения связаные со вкладчиков в текстовый файл.

1.3 Описание пользовательского интерфейса

Когда запускаете программу то на дисплее откроется меню(1.1).В котором находятся две кнопки: добавление нового вкладчика(1.2) и операции со вкладчиком(1.3), а также текущая дата(1.4)

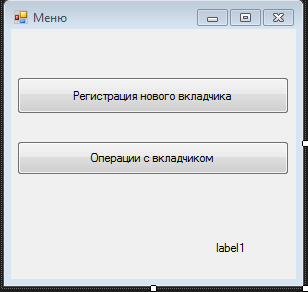


Рисунок 1.1. Вид главной формы.

Рисунок 1.2. Добавление кладчика. Рисунок 1.3.Операции с вкладчиком.



Рисунок 1.4.Текущая дата.

При нажатии на кнопку «Регистрация нового вкладчика» откроется регистрационое окно с полями ввода(1.5).При правильном заполнении и нажатии на кнопку «ОК» вы добавите нового вкладчика и выйдете в меню.Также при нажатии на кнопку «Назад» вы также выйдете в меню , но без добавления вкладчика.

При нажатии на кнопку «Операции с вкладчиком» отероется окно с выбором

Фильтра поиска, полем ввода данных, а также пустые данные(1.6). При правильном выборе фильтра и вводе дпнных вам выведится информация о пользователе и появится фильтр операций , поле ввода суммы и кнопка активации операции(1.7).После выполнения операции данные обновляются.

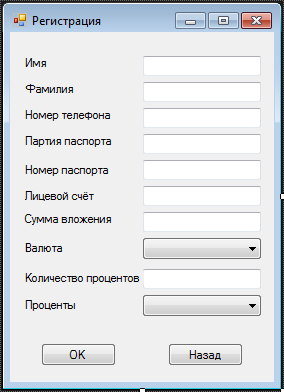


Рисунок 1.5.Регистрация вкладчика.

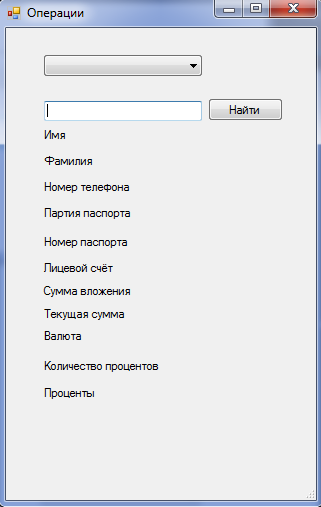


Рисунок 1.6. Вид открывшееся формы с поиском вкладчика.

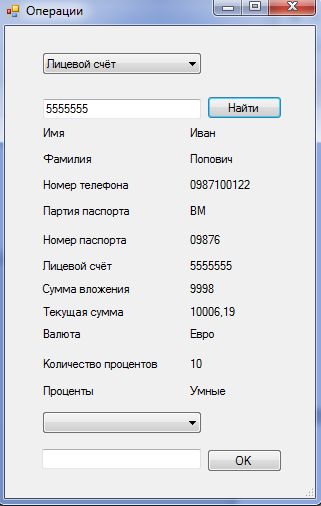


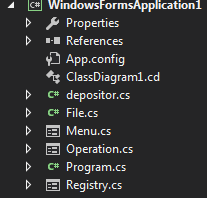
Рисунок 1.7.Форма операции целиком.

2 ПРОЕКТНАЯ СПЕЦИФИКАЦИЯ

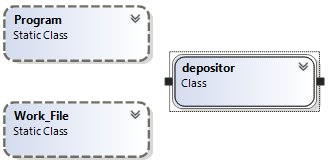
2.1 Объектная модель программы

В данной программе предоставлены классы, которые взаимодействуют друг с другом (рис.2.1).

В данной программе реализованы классы: depositor, Work\_File, Program, объектная модель[2] которых изображена на рисунке 2.2.



2.1. Архитектура проекта.



2.2. Объектная модель классов.

2.1.1 Класс depositor

Класс реализующий вкладчика. Присутствует конструктор, метод добавления и снятия денег(2.3).

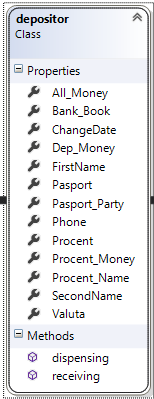


Рисунок 2.3. Класс depositor

2.1.2 Класс Work\_File

Статический класс объединяющий формы с вкладчиком, производит работу с файлами(2.4).Имеет методы добавление, удаление вкладчика, начисление процентов за день, инициализация следующего дня, занесение данных в файлы.

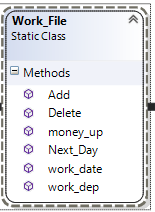


Рисунок 2.4. Класс Work\_File

3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

3.1 Установка программы

Для установки программы на компьютер необходимо выполнить следующие шаги в заданной последовательности.

Шаг 1 – Запускаем «Мой компьютер».

Шаг 2 – Заходим на носитель, с программным продуктом.

Шаг 3 – Распакуем представленный архив, в желаемую папку на вашем компьютере.

Шаг 4 – Поздравляю, можете запустить «.exe» файл и пользоваться данным приложением.

3.2 Работа с программой

Запускаем программу. Главное окно представлено на рисунке 3.1.

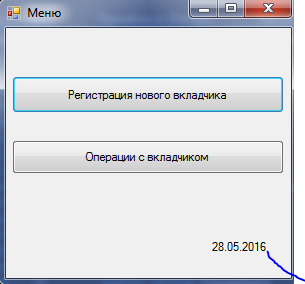
Текущая дата

Рисунок 3.1.Главное окно.

3.2.1 Добавление вкладчика

Для того чтобы добавить нового вкладчика нужно нажать на кнопку «Регистрация нового вкладчика». Затем заполнить поля, если не правильно заполните и нажмёте кнопку «ОК», то выдаст ошибку(3.2).При правильном заполнении полей и нажатии на кнопку «ОК», вы добавите вкладчика и вернётесь на главную форму.

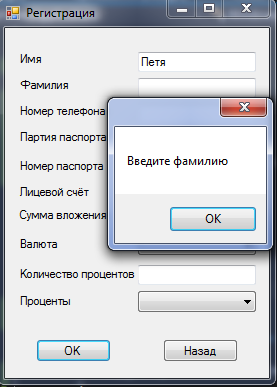


Рисунок 3.2.Ошибка при регистрации.

3.2.2 Поиск и операции над вкладчиком.

Чтобы пройти к операциям над вкладчиком нужно нажать на кнопку «Операции с вкладчиком».Затем выбрать фильтр поиска и заполнить поле, если вы неправильно выбрали фильтр или заполнили поле при нажатии на кнопку «Найти» выдаст ошибку(3.3).Если вы всё правильно сделали и нажали на кнопку «Найти», то вам выведится информация о вкладчике и предоставится возможно пополнить счёт или снять деньги(3.4).

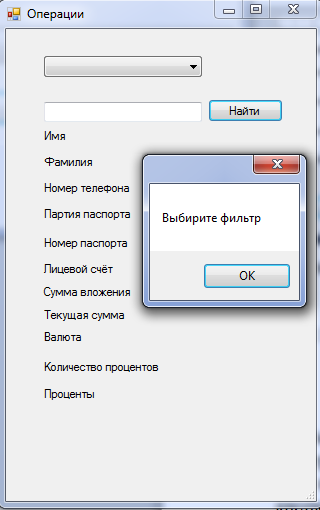


Рисунок 3.3. Ошибка при поиске вкладчика.

Затем чтобы пополнить счёт или снять деньги вам надо выбрать операцию и ввести сумму, если вы не правильно выбрали фильтр или некорректно вели сумму и нажали на кнопку «ОК», то выведет ошибку(3.5).Если вы всё правильно сделали и нажали на кнопку «ОК», то выполнятся изменнеия и в информации сразу это покажится.

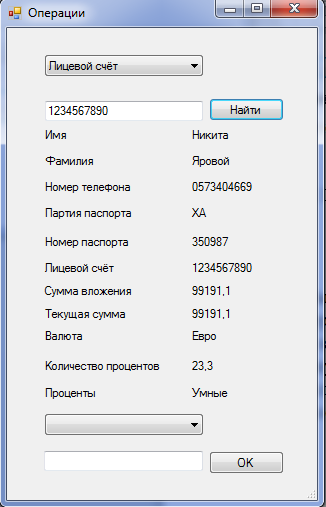


Рисунок 3.4.Успешный поиск вкладчика.

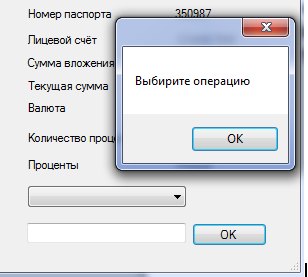


Рисунок 3.5. Ошибка при операции с вкладчиком.

3.2.3 Выход

Что бы завершить работу с программой необходимо в правом верхнем углу программы нажать крестик (рис.3.6).



Рисунок 3.6. Выход из программы.

ВЫВОДЫ

Получена программа под названием «Сбербанк», которая позволяет добавлять вкладчиков, искать вкладчиков, снимать деньги и пополнять счёт, а также сохранять вкладчиков txt-файл. При регистрации вкладчика задаётся его имя, фамилия, телефон, паспортные данные, сумма вложения, проценты и валюта хранимых денег.

В результате написания курсового проекта был создан полноценный программный продукт. Приложение было разработано полностью в соответствии с составленным планом. Так же были изучены отдельные фазы разработки программного обеспечения и компоновка их в единое целое. Не маловажным является, что не только закреплены знания по дисциплине, а также получен опыт в разработке программного продукта, изучены стандарты оформления документации.

В дальнейшем возможно расширение функционала программы. Например добавление роли администратора, составление схем по прибыли и убыткам банка.

ПЕРЕЧЕНЬ ССЫЛОК

1. Аксенов К. А. Моделирование и принятие решений в организационно-технических системах [Текст] / К. А. Аксенов. – Екатеринбург: Урал, 2015. – 103 с.
2. Microsoft Developer Network. Библиотека MSDN. Разработка на .NET [Электронный ресурc] / MSDN – сеть разработчиков Microsoft. – Режим доступа : www/URL:https://msdn.microsoft.com/ru-ru/ – 25.05.2016 г. – Загл. с экрана.
3. Microsoft Developer Network. Библиотека MSDN. Разработка на .NET [Электронный ресурc] / MSDN – сеть разработчиков Microsoft. – Режим доступа : www/URL:https://msdn.microsoft.com/ru-ru/ – 25.05.2016 г. – Загл. с экрана.
4. Бондарев В. М. «Объектно-ориентированное программирование на С#» [Текст]: учеб. Пособ / В. М. Бондарев. – Х.: Смит, 2009. – 224 с.;
5. Документація. Звіти у сфері науки і техніки : Структура і правила оформлення : ДСТУ 3008-95. – [Чинний від 1996–01–01]. – К. : Держстандарт України, 1995. – 37 с.

Приложение А

Фрагменты кода программы

//пополнение счёта.

public void receiving(double money)

{

Dep\_Money += Math.Round(money, 2);

}

//снятие денег.

public bool dispensing(double money)

{

if (Procent\_Money >= money)

{

Procent\_Money -= money;

return true;

}

if (Procent\_Money + Dep\_Money >= money)

{

Dep\_Money += Procent\_Money - money;

Procent\_Money = 0;

return true;

}

MessageBox.Show("Не достаточно денег на счету!");

return false;

}

Класс Work\_File

static class Work\_File

{

//запись последней даты

public static void work\_date(string date)

{

string path = @"D:\Data\date.txt";

using (StreamWriter sw = File.AppendText(path))

{

sw.WriteLine(date);

}

}

//создание файла со вкладчиками которые существуют на момент наступление даты date.

public static void work\_dep(List<depositor> dep,string date)

{

string path = @"D:\Data\" + date+".txt";

using (StreamWriter sw = File.AppendText(path))

{

foreach(depositor d in dep)

{

sw.WriteLine("DEP");

sw.WriteLine(d.FirstName);

sw.WriteLine(d.SecondName);

sw.WriteLine(d.Bank\_Book);

sw.WriteLine(d.Pasport);

sw.WriteLine(d.Phone);

sw.WriteLine(d.Pasport\_Party);

sw.WriteLine(d.Dep\_Money);

sw.WriteLine(d.Procent\_Money);

sw.WriteLine(d.Procent);

sw.WriteLine(d.Procent\_Name);

sw.WriteLine(d.Valuta);

sw.WriteLine(d.ChangeDate);

}

}

path = @"D:\Data\Change\_" + date + ".txt";

using (StreamWriter sw = File.AppendText(path))

{

}

}

//получение денег за день.

public static void money\_up(List<depositor> dep,string date)

{

DateTime dt = DateTime.Parse(date);

int day=365;

if (DateTime.IsLeapYear(dt.Year))

{

day = 366;

}

foreach (depositor d in dep)

{

if (d.Procent\_Name == "Умные")

{

d.Procent\_Money += Math.Round((d.Procent\_Money+d.Dep\_Money) \* d.Procent / 100/ day,2);

}

else

{

d.Procent\_Money += Math.Round(d.Dep\_Money \* d.Procent / 100/ day,2);

}

}

}

//добавление вкладчика

public static void Add(string f\_name,string s\_name,string bank\_book,string phone,string p\_pasport,string pasport,double sum,string value,double procent,string name\_procent,string date)

{

depositor dep = new depositor();

string path = @"D:\Data\Change\_" + date+".txt";

using (StreamWriter sw = File.AppendText(path))

{

sw.WriteLine("Add");

sw.WriteLine(dep.FirstName = f\_name);

sw.WriteLine(dep.SecondName = s\_name);

sw.WriteLine(dep.Bank\_Book = bank\_book);

sw.WriteLine(dep.Pasport = pasport);

sw.WriteLine(dep.Phone = phone);

sw.WriteLine(dep.Pasport\_Party = p\_pasport);

sw.WriteLine(dep.Dep\_Money = sum);

sw.WriteLine(dep.Procent = procent);

sw.WriteLine(dep.Procent\_Name = name\_procent);

sw.WriteLine(dep.Valuta = value);

sw.WriteLine(dep.ChangeDate = date);

}

}

//удаление вкладчика

public static void Delete(string bank\_book,string date)

{

string path = @"D:\Data\Change\_" + date+".txt";

using (StreamWriter sw = File.AppendText(path))

{

sw.WriteLine("Delete");

sw.WriteLine(bank\_book);

}

}

//наступление следующего дня.

public static string Next\_Day(string date)

{

string path = @"D:\Data\" +date+".txt";

List<depositor> depositors = new List<depositor>();

using (StreamReader sr = File.OpenText(path))

{

string s;

while ((s = sr.ReadLine()) != null)

{

if (s == "DEP")

{

depositor dep = new depositor();

dep.FirstName = sr.ReadLine();

dep.SecondName = sr.ReadLine();

dep.Bank\_Book = sr.ReadLine();

dep.Pasport = sr.ReadLine();

dep.Phone = sr.ReadLine();

dep.Pasport\_Party = sr.ReadLine();

dep.Dep\_Money = double.Parse(sr.ReadLine());

dep.Procent\_Money = double.Parse(sr.ReadLine());

dep.Procent = double.Parse(sr.ReadLine());

dep.Procent\_Name = sr.ReadLine();

dep.Valuta = sr.ReadLine();

dep.ChangeDate = sr.ReadLine();

depositors.Add(dep);

}

}

}

path = @"D:\Data\Change\_" + date+".txt";

using (StreamReader sr = File.OpenText(path))

{

string s;

while ((s = sr.ReadLine()) != null)

{

if (s == "Add")

{

depositor dep = new depositor();

dep.FirstName = sr.ReadLine();

dep.SecondName = sr.ReadLine();

dep.Bank\_Book = sr.ReadLine();

dep.Pasport= sr.ReadLine();

dep.Phone= sr.ReadLine();

dep.Pasport\_Party= sr.ReadLine();

dep.Dep\_Money= double.Parse(sr.ReadLine());

dep.Procent= double.Parse(sr.ReadLine());

dep.Procent\_Name= sr.ReadLine();

dep.Valuta= sr.ReadLine();

dep.ChangeDate= sr.ReadLine();

dep.Procent\_Money= 0;

depositors.Add(dep);

}

if (s == "Delete")

{

s = sr.ReadLine();

foreach(depositor dep in depositors)

{

if (dep.Bank\_Book == s)

{

depositors.Remove(dep);

break;

}

}

}

if (s == "Money\_up")

{

s = sr.ReadLine();

foreach(depositor dep in depositors)

{

if (dep.Bank\_Book == s)

{

s = sr.ReadLine();

dep.receiving(double.Parse(s));

dep.ChangeDate = date;

break;

}

}

}

if (s == "Money\_down")

{

s = sr.ReadLine();

foreach (depositor dep in depositors)

{

if (dep.Bank\_Book == s)

{

s = sr.ReadLine();

dep.dispensing(double.Parse(s));

dep.ChangeDate = date;

break;

}

}

}

}

}

money\_up(depositors, date);

string date1 = DateTime.Parse(date).AddDays(1).ToString();

date = date1.Substring(0, 10);

work\_dep(depositors, date);

return date;

}

}