Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и радиоэлектроники

филиал «Минский радиотехнический колледж»

**Отчет по практике по программированию**

Выполнила: учащаяся группы

Руководитель практики: Лазицкас Е.А.

Минск 2018

Содержание

Введение.................................................................................................................................3

1 Постановка задачи..............................................................................................................4

1.1 Описание предметной области.......................................................................................4

1.2 Обзор существующих аналогов.....................................................................................7

1.3 Информационная база.....................................................................................................10

1.4 Формулировка задачи.....................................................................................................11

2 Проектирование задачи.....................................................................................................12

2.1 Обоснования выбора средств и среды разработки......................................................12

2.2 Разработка алгоритма решения.....................................................................................13

2.3 Проектирование интерфейса........................................................................................15

3 Реализация.........................................................................................................................17

3.1 Компоненты Windows Forms........................................................................................17

3.2 Описание реализованных методов...............................................................................17

3.3 Описание разработанных классов................................................................................18

4 Тестирование.....................................................................................................................20

5 Применение.......................................................................................................................23

5.1 Установка и условия использование приложения......................................................23

5.2 Требования к аппаратно-программным ресурсам ПК...............................................23

5.3 Демонстрационный пример работы............................................................................23

Заключение..........................................................................................................................30

Список использованных источников................................................................................31

Приложение А Текст программы......................................................................................32

Приложение Б Экранные формы.......................................................................................41

Введение

Для того, чтобы получить опыт, необходимо практиковаться. Это очень важный элемент учёбы, ведь теоретические знания без практических – ничто. Невозможно научиться говорить на иностранном языке, если не произносить слова, не общаться с преподавателем или носителем языка, невозможно научиться писать код, если не практиковаться писать его в среде разработки. Именно поэтому нужно сочетать эти два метода получения знаний для максимизации полученной информации. Но во всём следует ставить цели и задачи, ведь без стратегии никогда не выиграть. Точно так же цели и задачи стоит поставить конкретно, чтобы не отвлекаться на то, что не имеет большого значения. Я поставила для себя следующие цели и задачи:

- данная практика проходит в филиале БГУИР «МРК»;

- необходимо повысить свой уровень написания программ, для этого постоянно практиковаться в их написании;

- научиться работать как в коллективе, так и индивидуально.

1 Постановка задачи

* 1. Описание предметной области

Автоматизации учета принимаемых звонков предназначена для международного отделения Автоматической телефонной станции (далее АТС). Основным предназначением АТС является обеспечение связью своих внутренних абонентов. Выход на городские телефоны ограничивается входящими в АТС городскими линиями. Сама АТС не может увеличить их количество, однако, использование и монтаж АТС позволяет оптимизировать связь благодаря грамотному распределению доступных линий. Это значит, что линия и номер телефона не закреплены за каким-либо конкретным аппаратом, а могут быть задействованы с любого телефона. АТС автоматически определяет свободную линию и проводит звонок (входящий или исходящий) через нее. Если же АТС является одним из элементов некоторой телефонной сети — то данная АТС должна обеспечивать соединение и поддержание связи со всеми абонентами сети — как «своими», так и подключенными к другим АТС сети.

Пользователями являются заведующий отделением и оператор АТС. Используется справочник, в котором прописаны города, их код и тариф за одну минуту. Диапазон работы АТС – страны Европы, а именно, их столицы. Список доступных городов должен обновляться, тем самым давая возможность звонить не только в столицы государств.

При создании записи формируется квитанция. Ей присваивается идентификационный номер, формируемый в зависимости от количества уже созданных квитанций. Квитанция содержит:

* дату разговора;
* код и название города;
* время разговора;
* стоимость (рассчитывается по формуле [тариф города за одну минуту] \* [продолжительность звонка]);
* номер телефона в этом городе;
* номер телефона абонента (абонент – это тот, кто обладает правом постоянного пользования какой-либо услугой, в данной случаем, связи).

Запрашиваемый пользователем отчет содержит:

* общее время разговора для каждого города;
* сумма на указанную дату (вводится с клавиатуры);
* название города, с абонентами которого наибольшее время разговоров.

|  |  |
| --- | --- |
| Тарифы на услуги по предоставлению международных телефонных соединений абонентам сети электросвязи общего пользования, оказываемые юридическим лицам и индивидуальным предпринимателям  Вводятся с 01 сентября 2019 года | Тариф за каждую полную или неполную минуту телефонного соединения без учета налога на добавленную стоимость, белорусских рублей |
| 1 СНГ I | 0,320 |
| 2 СНГ II | 0,352 |
| 3 СНГ III | 0,448 |
| 4 Европа I | 0,328 |
| 5 Европа II | 0,352 |
| 6 Европа III | 0,440 |

|  |  |
| --- | --- |
| Тарифная зона | Название страны |
| СНГ I | Россия (Москва) |
| Украина (Киев) |
| СНГ II | Армения (Ереван) |
| Грузия (Тбилиси) |
| Казахстан (Астана) |
| Кыргызстан (Бишкек) |
| Молдова (Кишинев) |
| Таджикистан (Душанбе) |
| Туркменистан (Ашхабад) |
| Узбекистан (Ташкент) |
| СНГ III | Азербайджан (Баку) |
| ЕВРОПА I | Германия (Берлин) |
| Италия,(Рим) |
| Литва (Вильнюс) |
| Польша (Варшава) |
| ЕВРОПА II | Австрия (Вена) |
| Болгария (София) |
| Босния и Герцеговина (Сараево) |
| Венгрия (Будапешт) |
| Латвия (Рига) |
| Македония (Скопье) |
| Румыния (Бухарест) |
| Сербия (Белград) |
| Словакия (Братислава) |
| Словения (Любляна) |
| Хорватия (Загреб) |
| Черногория (Подгорица) |
| Чехия (Прага) |
| Эстония (Таллинн) |
|  |
| ЕВРОПА III | Албания (Тирана) |
| Андорра (Андора-ла-Велья) |
| Бельгия (Брюссель) |
| Великобритания (Лондон) |
| Гибралтар (Гибралтар) |
| Греция (Афины) |
| Дания (Копенгаген) |
| Ирландия (Дублин) |
| Исландия (Рейкьявик) |
| Испания (Мадрид), включая Канарские о-ва |
| Кипр (Никосия) |
| Лихтинштейн (Вадуц) |
| Люксембург (Люксембург) |
| Мальта (Валлетта) |
| Монако (Монако) |
| Нидерланды (Амстердам) |
| Норвегия (Осло) |
| Португалия (Лиссабон), включая Мадейру и Азорские о-ва |
| Сан-Марино (Сан-Марино) |
| Турция (Анкара), включая Северный Кипр |
| Фарерские о-ва, территория Дании (Торсхавн) |
| Финляндия, (Хельсинки) |
| Франция (Париж), включая о-в Корсика |
| Швейцария (Берн) |
| Швеция (Стокгольм) |

* 1. Обзор существующих аналогов

Аналог – это то, что представляет собою подобие чего-либо, соответствие какому-либо предмету, явлению, понятию.

У большинства программ существуют аналоги, а для того, чтобы не создавать то, что ужу существует необходимо ознакомиться с ними.

Phontage

Рисунок 1 – Окна Phontage

Разработчик: Ericsson-LG, Республика Корея

Статус программы: лицензия

Версия: E.3Df

Интерфейс: русский

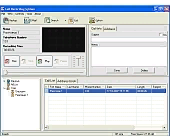
Система: Windows 2000/XP/7

Возможности приложения Phontage:

* авторизация абонента по id и паролю, без указания mac адреса;
* отображение номера звонящего при входящем вызове;
* удержание вызовов, перевод вызовов;
* программируемые клавиши;
* набор из записной книжки, меню быстрого вызова;
* всплывающее окно при входящем звонке;
* различные режимы отображения информации на дисплее;
* отображение списка вызовов;
* запись разговоров на жесткий диск компьютера;
* синхронизация с outlook, excel, goldmine, act;
* пользовательские профили;
* поддержка различных языков;
* регулировка громкости непосредственно из окна программы.

Недостаток: без лицензии допускается подключение к станции не более двух программных ip-терминалов, более 2 - необходима лицензия.

Call recording system

Рисунок 2 – Окно Call recording system

Разработчик: LG-Nortel, Ю.Корея

Статус программы: бесплатная

Версия: A.0Cb

Интерфейс: английский

Размер файла: 6.22 Mb zip-архив

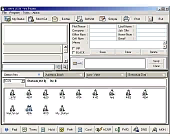
Система: Windows 2000/XP/Vista

Дата: добавлено: 24 марта 2010

Возможности приложения Call recording system:

* авторизация абонента по id и паролю, без указания mac адреса;
* разговоры записываются на жесткий диск компьютера;
* каждый разговор записывается в виде отдельного файла с автоматической фиксацией в базе данных программы номера звонящего, даты, времени и продолжительности;
* запись разговора включается оператором, принявшем вызов, нажатием кнопки на экране компьютера или путем использования горячих клавиш;
* встроенная возможность отправки записанных разговоров через электронную почту;
* поиск по базе данных записанных разговоров;
* прослушивание записанных разговоров;
* опции и настройки пользователя;
* резервное копирование записанных разговоров.

ez-Phone

Рисунок 3 – Окно ez-Phone

Разработчик: LG-Nortel, Ю.Корея

Статус программы: демо-версия на 30 дней

Версия: 3.8 Ab

Интерфейс: английский

Размер файла: 15.3 Mb zip-архив

Система: Windows 2000/XP/Vista

Дата: добавлено: 24 марта 2010

Возможности приложения ez-Phone:

* импорт и экспорт различных баз данных (outlook, gold mine, act!, access, excel);
* совместная работа со списком контактов outlook’a в реальном времени.;
* список зарезервированных наборов (номеров). (операция автоматического вызова);
* 3 вида всплывающих окон для извещения о входящем вызове;
* упрощенный вид рабочего окна (simple mode) и индивидуальный список напоминаний;
* осуществление исходящих вызовов, обработка входящих вызовов: постановка на удержание, перевод, парковка/снятие с парковки, конференция и т.д.;
* журнал вызовов для получения статистической информации;
* использование разнородных баз телефонных номеров;
* просмотр статуса обрабатываемого вызова;
* опции и настройки пользователя;
* отправка sms на мобильный телефон.

Недостаток: бесплатна только на 30 дней и только в демо-версии.

* 1. Информационная база

Информационная база данного ПО должна состоять из следующих входных данных:

* ­информация, которую вводит пользователь для заполнения квитанции звонка (дата, название города, код города, тариф (через запятую), стоимость, количество полных минут разговора, номер телефона звонящего, номер телефона абонента);
* текстовый файл с расширением txt;
* запросы пользователя при нажатии на различные элементы в процессе работы программы, для выполнения различных функций, таких как удаление, изменение, сохранение, определение города с наибольшем временем разговоров, вывод суммы на определенную дату.

А также из следующих выходных данных:

* информация, отображаемая на экране (интерфейс программы);
* стоимость звонка;
* сумма на запрашиваемую пользователем дату;
* название города наибольшим временем разговоров;
* текстовый файл с расширением docx (функция сохранение данных в файл).

К постоянным данным относятся:

* тарифы;
* города.
  1. Формулировка задачи

Создать MDI-приложение с использованием технологии Windows Forms на языке программирования C#, которое позволяет:

* сохранять вводимые данные (квитанции) в файле docx с названием, соответствующим идентификационному номеру квитанции;
* просматривать, изменять, удалять записи из файла;
* выводить результаты работы программы на экран и сохранять в другой текстовый файл;

Реализовать возможность переносить данные из одной квитанции в другую.

Добавить пункты меню для сохранения объектов в файл и загрузки данных из файла. При этом использовать стандартные диалоговые окна и механизм сериализации. В классе добавить поле дата создания объекта. Это поле не сериализовать, а при десериализации заново устанавливать по системной дате.

Основные требования к пользовательскому интерфейсу:

* приложение MDI-формата;
* функциональность (соответствие задачам пользователя);
* понятность и логичность (логичность расположения элементов на форме и удобство при эксплуатации программы);
* обеспечение защиты от человеческих ошибок;
* быстрое обучение пользователя (справка, в которой кратко и понятно описана работа с программой).

2. Проектирование задачи

2.1 Обоснования выбора средств и среды разработки

Выбор средств и среды разработки оказывает огромное влияние на процесс разработки и конечное ПС. Для создания ПС, которое будет удовлетворять всем вашим идеям и требованиям, необходимо выбирать средства и среду, исходя из их возможностей, но и реально смотреть на свои.

* среда разработки – Visual Studio 2019

Мой выбор пал на этот вариант, т.к. данная среда разработки имеет удобный интерфейс, является современной и обладает большими возможностями. В частности, встроенный редактор исходного кода с поддержкой технологии IntelliSense значительно упрощает процесс написания кода, ускоряя его. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Также встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. В среде существует довольно широкий выбор языков программирования, что также является значительным плюсом.

* интерфейс программирования приложений – Windows Forms

Windows Forms – интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework, упрощающий доступ к элементам интерфейса Microsoft Windows. Один из наиболее важных плюсов данного интерфейса – управляемый код – классы, реализующие API для Windows Forms, не зависят от языка разработки, следовательно, это дает возможность программисту использовать данный интерфейс не зависимо от выбора одного из предлагаемых языков средой разработки Visual Studio 2019.

* способ организации графического интерфейса пользователя – MDI

Multiple document interface (MDI) – способ организации графического интерфейса пользователя, предполагающий использование оконного интерфейса, в котором большинство окон (исключая, как правило, только модальные окна) расположены внутри одного общего окна. Этим он и отличается от SDI, в котором окна располагаются независимо друг от друга. У него много преимуществ, одно из которых: уменьшение загромождённости экрана элементами интерфейса, все окна приложения можно прятать/показывать, сворачивать/разворачивать и проводить с ними другие манипуляции, как с одним окном и т.д.;

* язык программирования – C#

C# - объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework. На данный момент C# — один из самых популярных языков программирования. Это важно для разработчиков, поскольку популярность языка прямо пропорциональна тому, насколько для него будут доступны онлайн-материалы. Чаще всего, все обращаются к Google или Stack Overflow для решения задач в разработке и чаще всего можно найти большое количество ответов по C#. Это сэкономит огромное количество времени новичкам при решении различных задач в разработке.

Гибкость языка C# является огромным преимуществом, по сравнению с Delphi. Разнообразие приложений, которые могут быть разработаны с помощью С#, .Net и Visual Studio практически безгранично:

* приложения для Windows;
* мобильные приложения;
* веб-приложения;
* игры;
* приложения для Android и iOS, которые разрабатываются с помощью дополнительных фреймворков, таких как Xamarin или Mono.

Вывод: разработка MDI-приложения с использованием технологии Windows Forms на языке программирования C# в Visual Studio 2019.

2.2 Разработка алгоритма решения

Пользователь будет выбирать действие: открыть квитанции, открыть отчет или справочник, закрыть программу.

При выборе квитанций пользователю будет отображаться перечень уже созданных квитанций, который будет открываться из файла, обновляемого после создания, удаления или изменения квитанций. Выбранная квитанция будет открываться из файла с названием, соответствующим идентификационному номеру квитанции. Теперь ее можно будет изменить или удалить.

При открытии отчета пользователю будет отображаться список всех квитанций. Далее, по требованию пользователя программа сможет вычислить сумму на определенную дату и город с наибольшим временем разговоров. Отчет будет формироваться с помощью поиска определенных данных в файлах квитанций и их использования в расчетах по заданным формулам.

При выборе справочника будет открываться файл со справочными данными в Word.

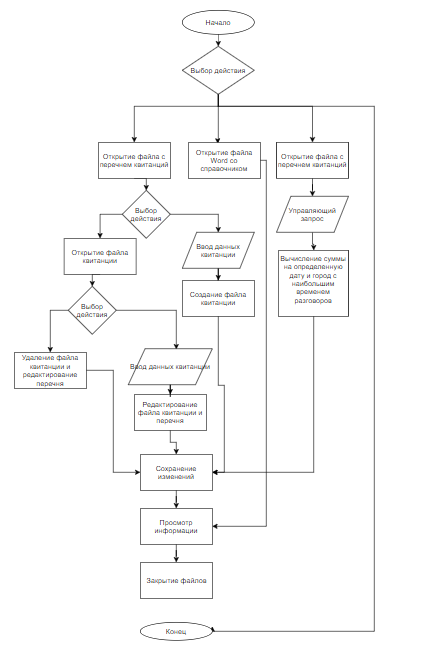


Рисунок 3 – Предполагаемый алгоритм работы программы

* 1. Проектирование интерфейса

В главной форме планируется наличие одной формы с кнопками выбора действия: «Квитанции», «Отчет», «Справочник», «Выход». В форме «Квитанции» будет отображаться перечень квитанций и действия: «Создать», «Назад». При открытии квитанции будут отображаться поля ввода данные: «Сохранить», «Назад» - при создании, и «Изменить», «Удалить» - при изменении квитанции.

Меню действий

Поля ввода данных

«Квитанции» «Отчет»

Меню действий

Перечень квитанций

Форма

Рисунок 4 – Предполагаемый интерфейс

Главное меню

1. Реализация

3.1 Компоненты Windows Forms

Были использованы следующие компоненты:

- Label – элемент управления, предназначенный для отображения текста на форме;

- TextBox – элемент управления, в который можно вводить информацию и использовать ее в дальнейшем;

- Button – элемент управления, при клике на который происходит какое-либо программно- прописанное действие.

- listBox – элемент управления «список».

- dateTimePicker – элемент управления, перехватывающий данные типа DateTime.

* 1. Описание реализованных методов

Form1

private void button1\_Click(object sender, EventArgs e) – для открытия Form4.

private void button2\_Click(object sender, EventArgs e) – для открытия Form3.

private void button3\_Click(object sender, EventArgs e) – для закрытия программы.

private void button4\_Click(object sender, EventArgs e) – для открытия Word сщ справочником.

Form2

private void button1\_Click(object sender, EventArgs e) – для открытия Form4 для создания квитанции.

private void Form2\_Load(object sender, EventArgs e) – обновляет listBox1 сщ списком квитанций из файла «Квитанции»

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e) – открывают выбранную в listBox1 квитанцию в Form6.

Form3

private void Form3\_Load(object sender, EventArgs e) – при загрузке формы загружает в listBox1 список существующих городов и вычисляет время разговора с ними, открывая каждый файл города из папки «Города».

private void button1\_Click(object sender, EventArgs e) – пополняет label1 результатом вычисления суммы на определенную пользователем дату, проходясь по всем файлам квитанции из папки «Квитанции».

private void button2\_Click(object sender, EventArgs e) – пополняет label2 названием города с наибольшим временем разговоров, вычисляемым при инициализации времени разговоров для города из listBox1.

Form4

private void button1\_Click(object sender, EventArgs e) – сохраняет все данный в файл квитанции в папке «Квитанции», идентификационный номер инкрементируется. Создается или редактируется время разговора с указанным городом в папке «Города».

private void button2\_Click(object sender, EventArgs e) – вставляет данные из файла «Буфер обмена».

Form6

private void Form6\_Load(object sender, EventArgs e) – при загрузке поля ввода пополняются данными из выбранного пользователем файла квитанции.

private void button1\_Click(object sender, EventArgs e) – редактирует квитанцию и ее файл, а также время разговора с городом и файл «Квитанции».

private void button2\_Click(object sender, EventArgs e) – удаляет файл квитанции и ее строку в файле «Квитанции».

private void button3\_Click(object sender, EventArgs e) – копирует данные из полей в фай «Буфер обмена».

private void textBox3\_TextChanged(object sender, EventArgs e) – служит для опознания города для редактирования.

* 1. Описание разработанных классов

Разработанная программа имеет 6 классов:

- Program – особый вид классов, в котором находится метод Main, представляющий собой точку входа в программу, содержащий в блоке инициализацию главной формы (Form1).

- Form1 – класс главной формы под названием «International automated telephone exchange» с 4 кнопками. Они вызывают классы, описанные ниже (Form2 («Квитанции» button1), Form3 («Отчет» button2), одна открывает справочник («Справочник» button4) и обеспечивают выход из программы («Выход» button4).

- Form2 – класс формы отображения квитанций. Он служит для выбора пользователем квитанции (listBox1 с обновляющимся списком квитанций) или создания новой («Создать» button1). Вызывает классы Form4 и Form6, описанные ниже.

- Form6 – класс формы изменения («Изменить» button1) , удаления («Удалить» button2) и копирования («Копировать» button3) уже существующей квитанции в файл «Буфер обмена». Содержит обозначающие лейблы: Дата label1, Город label2, Код label3, Время разговора label4, Тариф label5, Номер телефона звонящего label6, Номер телефона абонента label7 – и соответствующие им блоки ввода: dateTimePicker1, textBox3, textBox4, textBox5, textBox6, textBox1, textBox2.

- Form3 – класс формы, отображающий отчет по квитанциям, а именно: список городов с временем разговора с ними listBox1, label1и кнопка «Показать» button1 для вывода суммы на выбранную дату dateTimePicker1, кнопка «Определить город» button2 и label3 для вывода результата в label2.

- Form4 – класс формы создания и вставки квитанции из буфера обмена. Аналогичен Form6 кроме кнопок. Содержит кнопки «Вставить» button2 для вставки данных из файла «Буфер обмена» и «Сохранить» button1 для сохранения данных в отдельный файл.

- Identificator – класс идентификатора, необходимый для его сериализации в файл «Identificator.dat».

1. Тестирование

Тестирование – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определенным образом и выявление ситуаций, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

Существующие на сегодня методы тестирования программного обеспечения не позволяют однозначно и полностью выявить все дефекты и установить корректность функционирования анализируемой программы, поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого программного обеспечения.

Существует несколько признаков, по которым принято производить классификацию видов тестирования. Обычно выделяют следующие:

- по объекту тестирования;

- по знанию системы (тестирование чёрного ящика, тестирование белого ящика, тестирование серого ящика);

- по степени автоматизации;

- по степени изолированности компонентов;

- по времени проведения тестирования;

- по признаку позитивности сценариев;

- по степени подготовленности к тестированию.

На данный момент времени выделяют несколько видов тестирования по доступу к программному коду:

- тестирование «белого ящика» - тестирование программного кода с доступом к коду;

- тестирование «черного ящика» - тестирование без доступа к коду продукту;

- тестирование «серого ящика» - тестирование, основанное на ограниченном знании внутренней структуры программного продукта. Часть говорят, что это смесь тестирования «белого ящика» и «черного ящика», но это неверно. В данном случае тестировщик не работает с кодом программного продукта, но он знаком с внутренней структурой программы или приложения и взаимодействием между компонентами.

При тестировании разработанной программы использовалось тестирование «черного ящика» или поведенческое тестирование — метод тестирования функционального поведения объекта (программы, системы) с точки зрения внешнего мира, при котором не используется знание о внутреннем устройстве тестируемого объекта.

Стратегия поведенческого теста исходит из технических требований и их спецификаций (таблица 1).

Таблица 1 – Тестирование программного средства

|  |  |  |
| --- | --- | --- |
| Действие | Ожидание | Получено |
| Запуск программы | Открытие программы, а именно родительской формы | В результате тестов получено полное соответствие ожиданиям |
| Реакция на нажатие Квитанции | Открытие формы со списком квитанций |
| Реакция на нажатие Отчет | Открытие формы отчета по разговорам |
| Реакция на нажатие Справочник | Открытие программы MS Word с файлом «Guide» |
| Реакция на нажатие Выход | Закрытие всех форм, то есть завершение программы |
| Реакция на нажатие Квитанции => Создать | Открытие формы с полями для ввода данных о квитанции |
| Реакция на нажатие Квитанции => Создать => Сохранить | Сохранение данных из полей в создаваемый файл Квитанции/[идентификационный номер].txt, редактирование файла Квитанции.txt, создание или редактирование файла Города/[город абонента].txt. Отображается диалоговое окно с сообщением о сохранении квитанции в файл. |
| Реакция на нажатие Квитанции => Создать => Вставить | Заполнение полей данными из файла Буфер обмена.txt |
| Реакция на нажатие Квитанции => [квитанция из listBox1] | Открытие формы с заполненными полями из файла Квитанции/[идентификационный номер].txt. |
| Реакция на нажатие Квитанции => [квитанция из listBox1] => Изменить | Редактирование файла Квитанции/[идентификационный номер].txt, редактирование файла Квитанции.txt, создание или редактирование файла Города/[город абонента].txt. Отображается диалоговое окно с сообщением о редактировании квитанции в файле. |
| Реакция на нажатие Квитанции => [квитанция из listBox1] => Скопировать | Копирование данных из полей в файл Буфер обмена.txt. Отображается диалоговое окно с сообщением о копировании данных квитанции в файл. |
| Реакция на нажатие Квитанции => [квитанция из listBox1] => Удалить | Удаление файла Квитанции/[идентификационный номер].txt, редактирование файла Квитанции.txt, редактирование файла Города/[город абонента].txt |
| Реакция на нажатие Отчет => Показать | Отображение общей суммы разговоров на указанную пользователем дату в label1 |
| Реакция на нажатие Отчет => Определить город | Отображение названия города с наибольшим временем разговоров в label3 |

1. Применение
   1. Установка и условия использования приложения

Для работы в приложении установка не требуется.

Данное ПО не нуждается в дополнительных файлах. Они создаются автоматически создаются, редактируются и удаляются в папке \IATE\bin\Debug. Редактирование файлов может повлечь за собой поломку всей программы.

5.2 Требования к аппаратно-программным ресурсам ПК

Минимальные системные требования:

Microsoft Windows XP/Vista/7/8/8.1/10; Intel Pentium/Atom/Celeron 1,2 ГГц, ОЗУ 1 Гб, свободное место на жестком диске – 500 кб.

Рекомендуемые системные требования:

Intel Pentium/Atom/Celeron/Core i3 1,5 ГГц, ОЗУ 2 Гб, свободное место на жёстком диске – 1Мб.

5.3 Демонстрационный пример работы

При открытии программы пользователю открывается меню (рисунок 1). Для открытия списка квитанций пользователь нажимает кнопку «Квитанции» и – открывается окно «Квитанции» (рисунок 2).

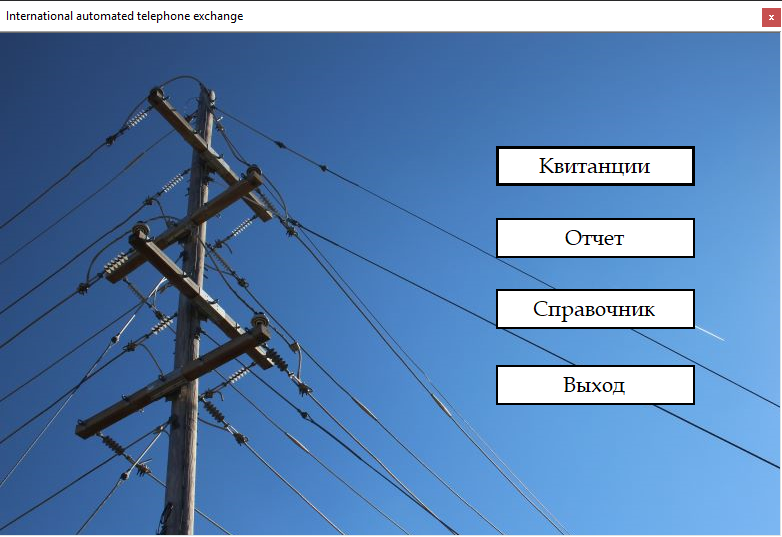


Рисунок 1 – Главное меню

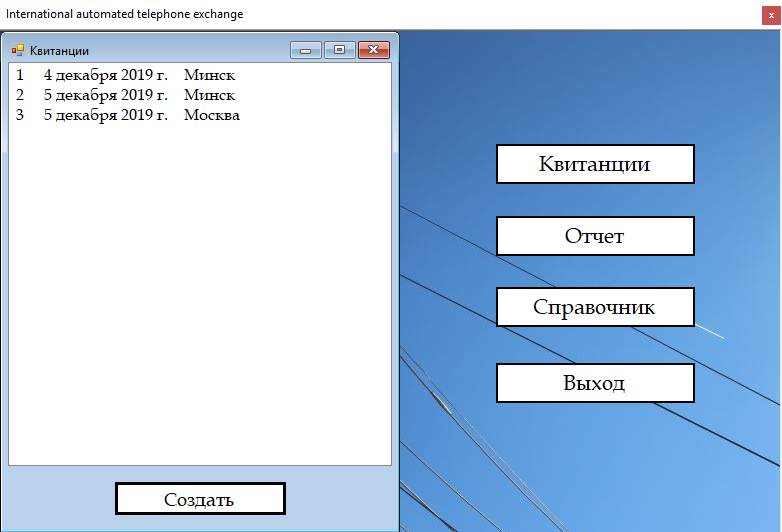


Рисунок 2 – Окно Квитанции

При нажатии на кнопку «Создать» открывается окно создания квитанции (рисунок 3). При нажатии кнопки «Сохранить» отображается диалоговое окно сообщением о сохранении квитанции (рисунок 4). При нажатии кнопки «Вставить» в поля вставляются уже готовые значения из буфера обмена.

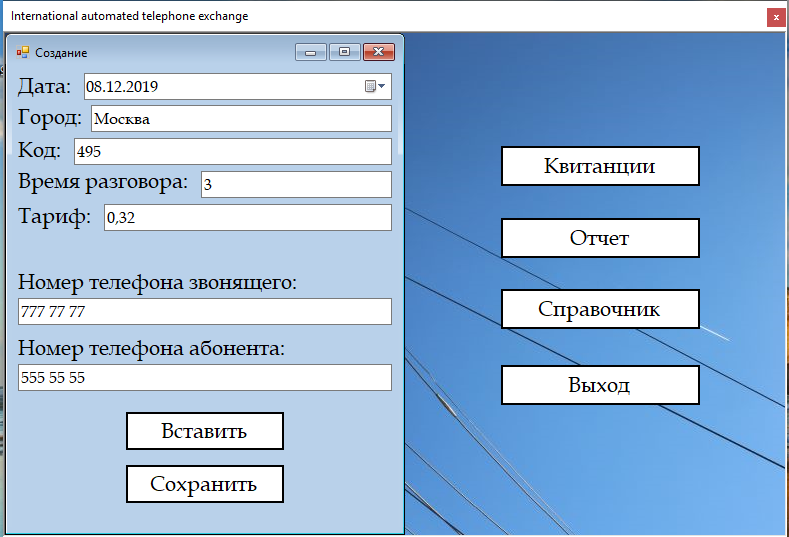


Рисунок 3 – Окно создания квитанции

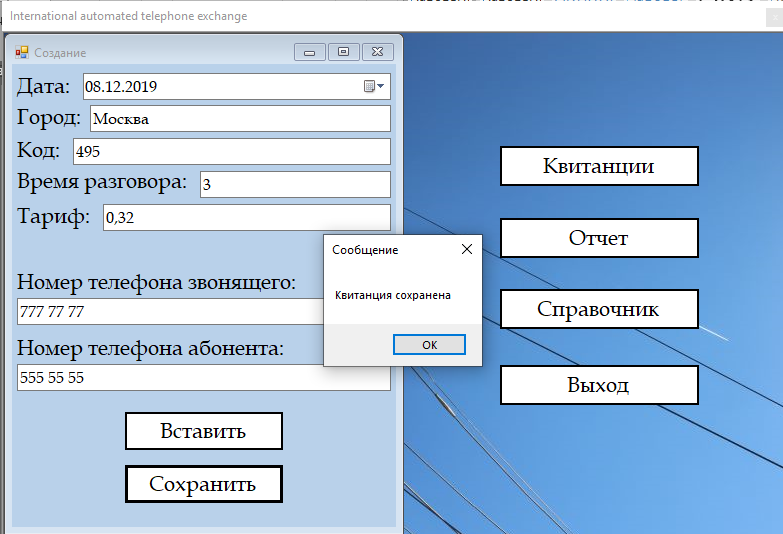


Рисунок 4 – Диалоговое окно

При нажатии на одну из квитанций в окне «Квитанции» открывается окно редактирования квитанций (рисунок 5). При нажатии кнопок «Изменить», «Скопировать» или «Удалить» отображается диалоговое окно с соответствующим сообщением (рисунок 6, рисунок 7 и рисунок 8).

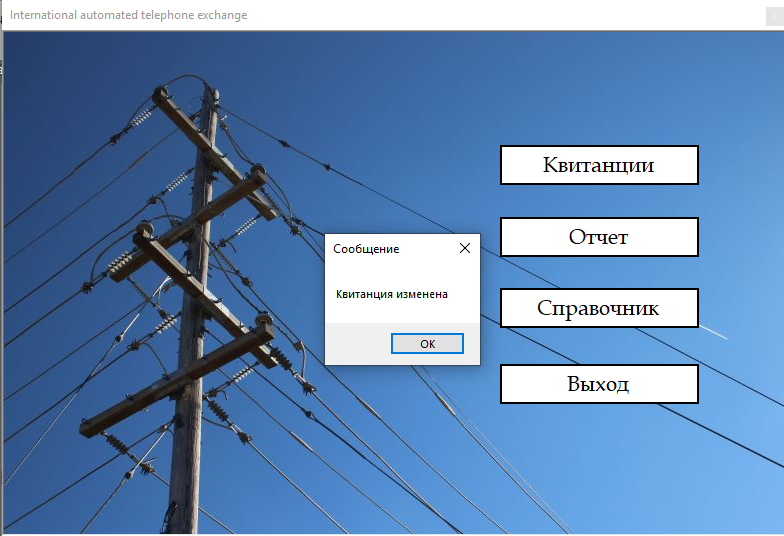


Рисунок 6 – Диалоговое окно

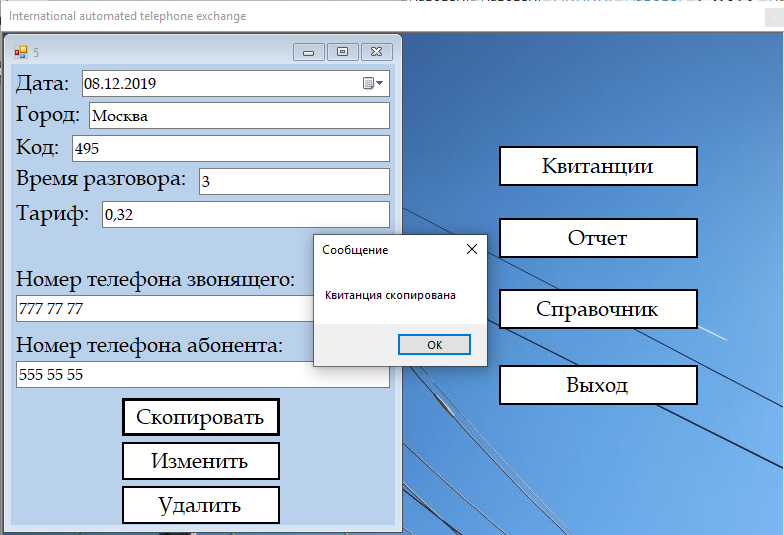


Рисунок 7 – Диалоговое окно

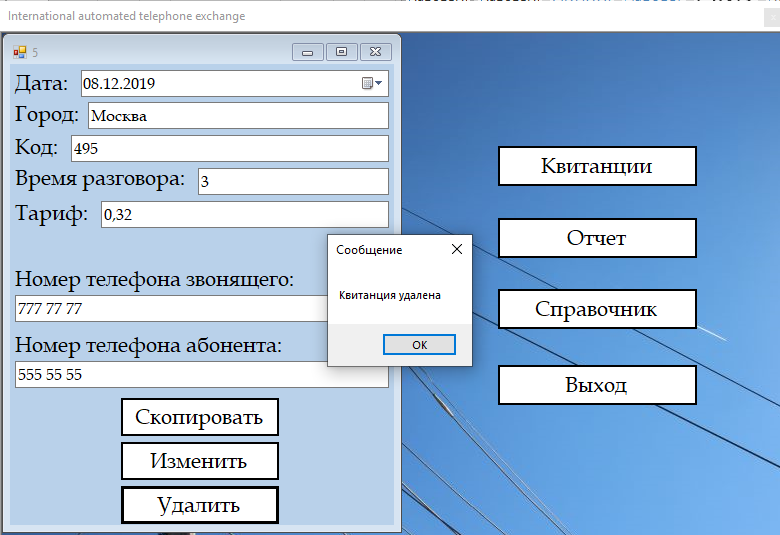


Рисунок 8 – Диалоговое окно

При удалении квитанции на ее месте в списке появляется информации о том, что она удалена (рисунок 9).

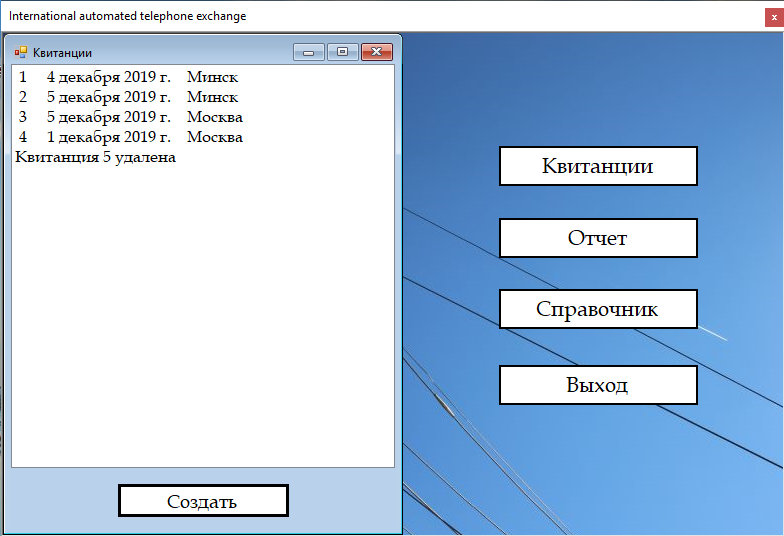


Рисунок 9 – Список квитанций

При нажатии на кнопку «Отчет» пользователю отображается форма с отчетом о времени разговора с каждый городом (рисунок 10). По желанию пользователя при нажатии на кнопку «Показать» форма отображает сумму на выбранную дату, а на кнопку «Определить город» - название города с наибольшим временем разговора (рисунок 11).

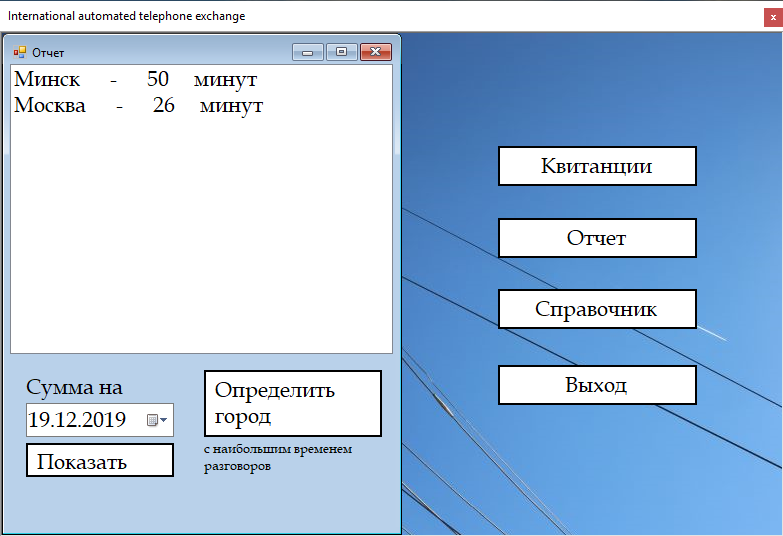


Рисунок 10 – Отчет

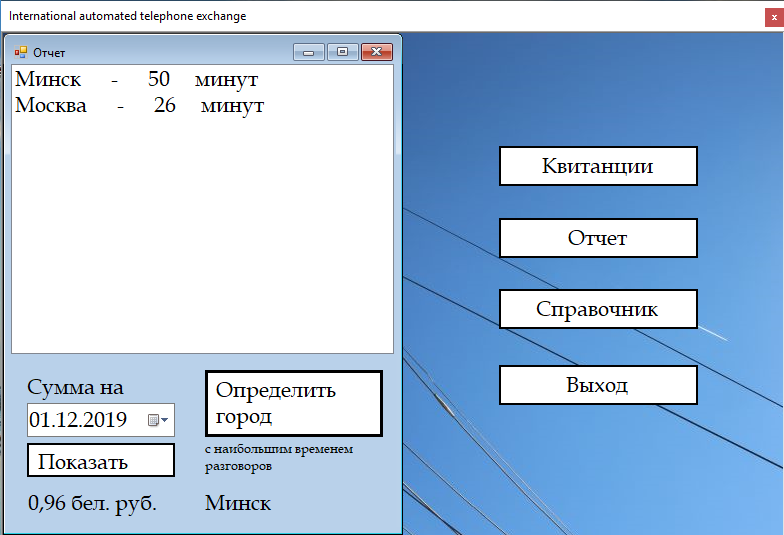


Рисунок 11 – Отчет с дополнительной информацией

При нажатии на кнопку «Справочник» открывается файл Guide.docx (рисунок 12).

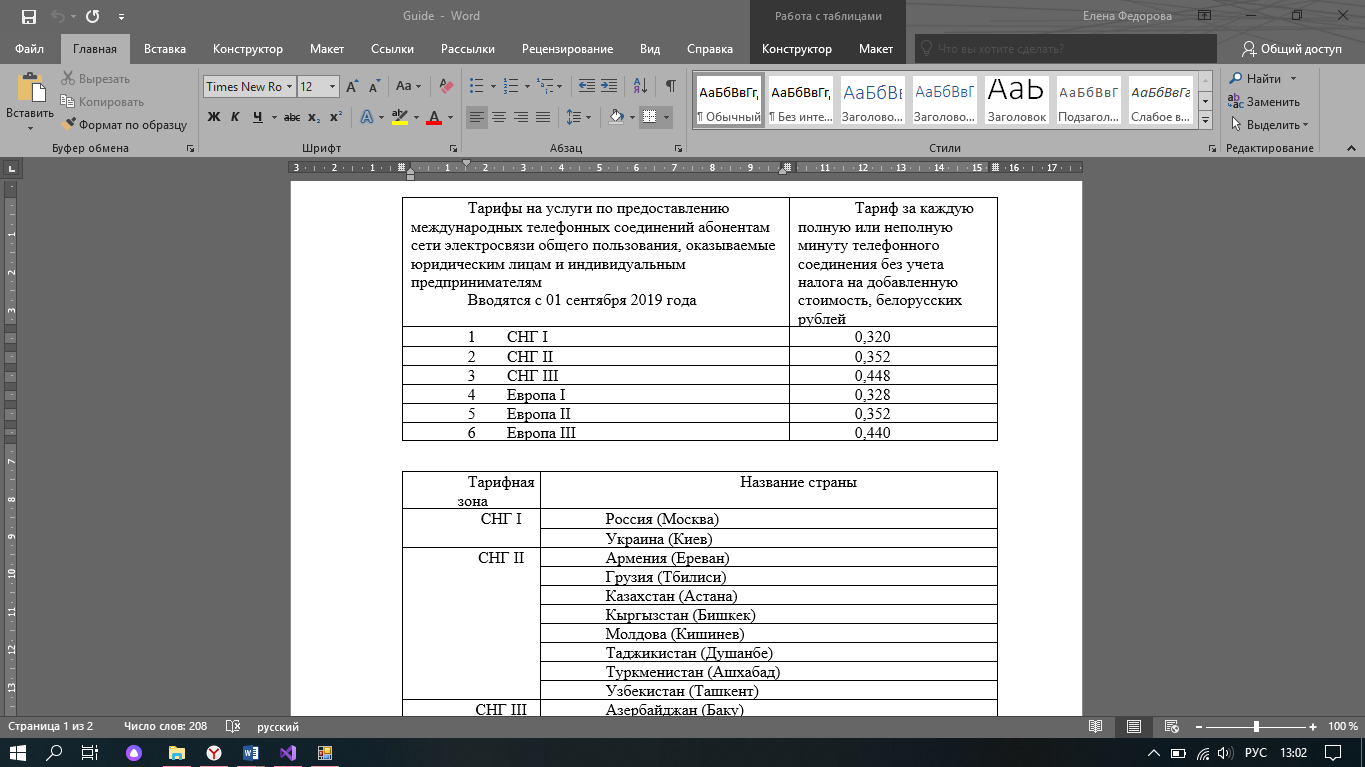


Рисунок 12 – Справочник

Завершить кнопку можно нажатием кнопки «Выход».

Заключение

Практика по программированию дала возможность получить дополнительный опыт в среде программирования Microsoft Visual Studio 2019, а также улучшить навыки в работе с языком программирования C#, что подразумевает под собой получение новых знаний, изучение предметной области и закрепление существующих умений.

Первым этапом разработки приложения после получения задания стало изучение предметной области, что дало ясное представление о том, какой должна быть программа, в частности с какими конкретными данными она должна работать и какие функции выполнять. Предметной областью являлись международные телефонные станции, изучая которые сформировалось преставление об их структуре и форме квитанций, необходимых для корректного хранения информации. Оперируя полученными данными был разработан простой и понятный интерфейс. Особое внимание уделялось формам работы с данными, т.к. они представляют собой полную информацию о международных звонках.

В дальнейшем изучалось создание MDI-приложения с использованием технологии WindowsForms на языке программирования C#.

Используя различные источники, было выполнено задание и улучшены навыки работы в среде разработки Visual Studio 2019 на языке программирования C#.

Список использованных источников

1. Албахарв, Джозеф. С# 6.0. Справочник. Полное описание языка, 6-е изд./ Албахарв, Джозеф : Пер. с англ. - М. : ООО
2. Стиллмен Э., Грин Дж. Изучаем C#. 3-е изд. — СПб.: Питер, 2014. — 816 с.: ил. — (Серия «Head First O’Reilly»).ISBN 978-5-496-00867-9
3. Шилдт, Герберт. C# 4.0: полное руководство.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2011. —1056 с.: ил. — Парал. тит. англ.
4. Хейлсберг А., Торгерсен М., Вилтамут С., Голд П. Язык программирования C#. Классика Computers Science. 4-е изд./ — СПб.: Пи-тер, 2012. — 784 с.: ил.
5. C#. Введение в программирование. Учебное пособие/Марченко А.Л. Издательство Московского университета, 2005

Интернет источники

1. <https://www.bestprog.net/ru/2016/11/13/>
2. [https://metanit.com](https://metanit.com/sharp/tutorial/6.4.php)
3. [https://docs.microsoft.com](https://docs.microsoft.com/ru-ru/dotnet/framework/winforms/controls/datagridview-control-windows-forms)

Приложение А

(обязательное)

Текст программы

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 IATE

{

public partial class Form1 : Form

{

static public Form1 mainParent;

public Form1()

{

InitializeComponent();

mainParent = this;

}

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

{

this.Close();

}

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

{

Form2 newForm = new Form2();

newForm.Show();

}

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

{

Form3 newForm = new Form3();

newForm.Show();

}

private void button4\_Click(object sender, EventArgs e)

{

System.Diagnostics.Process.Start(@"Guide.docx");

}

}

}

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 IATE

{

public partial class Form1 : Form

{

static public Form1 mainParent;

public Form1()

{

InitializeComponent();

mainParent = this;

}

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

{

this.Close();

}

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

{

Form2 newForm = new Form2();

newForm.Show();

}

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

{

Form3 newForm = new Form3();

newForm.Show();

}

private void button4\_Click(object sender, EventArgs e)

{

System.Diagnostics.Process.Start(@"Guide.docx");

}

}

}

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 IATE

{

public partial class Form1 : Form

{

static public Form1 mainParent;

public Form1()

{

InitializeComponent();

mainParent = this;

}

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

{

this.Close();

}

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

{

Form2 newForm = new Form2();

newForm.Show();

}

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

{

Form3 newForm = new Form3();

newForm.Show();

}

private void button4\_Click(object sender, EventArgs e)

{

System.Diagnostics.Process.Start(@"Guide.docx");

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace IATE

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

this.MdiParent = Form1.mainParent;

}

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

{

Form4 newForm = new Form4();

newForm.Show();

Close();

}

private void Form2\_Load(object sender, EventArgs e)

{

StreamReader file = new StreamReader("Квитанции.txt");

while (!file.EndOfStream)

{

listBox1.Items.Add(file.ReadLine());

}

file.Close();

}

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

string expression = "";

for (int i = 0; i < 6; i++)

{

expression += listBox1.SelectedItem.ToString()[i];

}

Program.exchange = Convert.ToInt32(expression).ToString();

Program.exchange2 = listBox1.SelectedItem.ToString();

string id = expression;

Form6 newForm = new Form6();

newForm.Text = id;

newForm.Show();

Close();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace IATE

{

public partial class Form3 : Form

{

int max\_time = 0;

string max\_city;

public Form3()

{

InitializeComponent();

this.MdiParent = Form1.mainParent;

}

private void Form3\_Load(object sender, EventArgs e)

{

string time;

DirectoryInfo dir = new DirectoryInfo(@"Города/");

foreach (var item in dir.GetFiles())

{

StreamReader file = new StreamReader("Города/" + item.Name);

time = file.ReadLine();

listBox1.Items.Add((item.Name + "- " + time + " минут").Replace(".txt", " "));

file.Close();

if (Convert.ToInt32(time) > max\_time)

{

max\_time = Convert.ToInt32(time);

max\_city = item.Name;

}

}

}

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

{

int temp, time;

double tariff, result=0;

string current\_date;

string date = dateTimePicker1.Value.ToString().Replace("0:00:00", "").Replace(" ", "");

DirectoryInfo dir = new DirectoryInfo(@"Квитанции/");

foreach (var item in dir.GetFiles())

{

StreamReader file = new StreamReader("Квитанции/" + item.Name);

current\_date = file.ReadLine();

if (current\_date == date)

{

file.ReadLine();

file.ReadLine();

time = Convert.ToInt32(file.ReadLine());

tariff = Convert.ToDouble(file.ReadLine());

result += time \* tariff;

}

file.Close();

}

label2.Text = result.ToString() + " бел. руб.";

}

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

{

label4.Text = max\_city.Replace(".txt", "");

}

}

}

using System;

using System.IO;

using System.Windows.Forms;

using System.Runtime.Serialization.Formatters.Binary;

namespace IATE

{

public partial class Form4 : Form

{

DateTime date;

string city;

string code;

int time;

float tariff;

string caller;

string subscriber;

public Form4()

{

InitializeComponent();

this.MdiParent = Form1.mainParent;

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

city = textBox3.Text;

}

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

date = dateTimePicker1.Value;

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

code = textBox4.Text;

}

private void textBox5\_TextChanged(object sender, EventArgs e)

{

time = Convert.ToInt32(textBox5.Text);

}

private void textBox6\_TextChanged(object sender, EventArgs e)

{

tariff = Convert.ToSingle(textBox6.Text);

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

caller = textBox1.Text;

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

subscriber = textBox2.Text;

}

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

{

Program.iden.id += 1;

string address = "Квитанции/" + Convert.ToString(Program.iden.id) + ".txt";

StreamWriter file = new StreamWriter(address);

file.WriteLine(dateTimePicker1.Value.ToString().Replace("0:00:00", "").Replace(" ", ""));

file.WriteLine(textBox3.Text);

file.WriteLine(textBox4.Text);

file.WriteLine(Convert.ToInt32(textBox5.Text));

file.WriteLine(Convert.ToSingle(textBox6.Text));

file.WriteLine(textBox1.Text);

file.WriteLine(textBox2.Text);

file.Close();

using (FileStream fs = new FileStream("Identificator.dat", FileMode.OpenOrCreate))

{

Program.formatter.Serialize(fs, Program.iden);

}

Program.exchange = " " + Convert.ToString(Program.iden.id) + " " + date.ToString("D") +" " + city;

StreamWriter file2 = new StreamWriter("Квитанции.txt", true);

file2.WriteLine(Program.exchange);

file2.Close();

int old\_time;

if (System.IO.File.Exists("Города/" + city + ".txt"))

{

StreamReader reader = new StreamReader("Города/" + city + ".txt");

old\_time = Convert.ToInt32(reader.ReadLine());

reader.Close();

StreamWriter writer = new StreamWriter("Города/" + city + ".txt");

writer.WriteLine(time+old\_time);

writer.Close();

}

else

{

System.IO.File.WriteAllText("Города/" + city + ".txt", time.ToString());

}

MessageBox.Show(

"Квитанция сохранена",

"Сообщение",

MessageBoxButtons.OK

);

}

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

{

StreamReader file = new StreamReader("Буфер обмена.txt");

dateTimePicker1.Value = Convert.ToDateTime(file.ReadLine());

textBox3.Text = file.ReadLine();

textBox4.Text = file.ReadLine();

textBox5.Text = file.ReadLine();

textBox6.Text = file.ReadLine();

textBox1.Text = file.ReadLine();

textBox2.Text = file.ReadLine();

file.Close();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace IATE

{

public partial class Form6 : Form

{

bool flag = false;

string id = Program.exchange;

int old\_time, old\_city\_time;

string old\_city;

public Form6()

{

InitializeComponent();

this.MdiParent = Form1.mainParent;

}

private void Form6\_Load(object sender, EventArgs e)

{

StreamReader file = new StreamReader("Квитанции/" + id + ".txt");

dateTimePicker1.Value = Convert.ToDateTime(file.ReadLine());

textBox3.Text = file.ReadLine();

old\_city = textBox3.Text;

textBox4.Text = file.ReadLine();

textBox5.Text = file.ReadLine();

old\_city\_time = Convert.ToInt32(textBox5.Text);

textBox6.Text = file.ReadLine();

textBox1.Text = file.ReadLine();

textBox2.Text = file.ReadLine();

file.Close();

}

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

{

StreamWriter file = new StreamWriter("Квитанции/" + id + ".txt");

file.WriteLine(dateTimePicker1.Value.ToString().Replace("0:00:00", "").Replace(" ", ""));

file.WriteLine(textBox3.Text);

file.WriteLine(textBox4.Text);

file.WriteLine(textBox5.Text);

file.WriteLine(textBox6.Text);

file.WriteLine(textBox1.Text);

file.WriteLine(textBox2.Text);

file.Close();

StreamReader reader = new StreamReader("Квитанции.txt");

string kvitations = reader.ReadToEnd();

reader.Close();

kvitations = kvitations.Replace(Program.exchange2, " " + Convert.ToString(id) + " " + dateTimePicker1.Value.ToString("D")+ " " + textBox3.Text);

StreamWriter writer = new StreamWriter("Квитанции.txt");

writer.Write(kvitations);

writer.Close();

int time;

if (System.IO.File.Exists("Города/" + textBox3.Text + ".txt"))

{

StreamReader city = new StreamReader("Города/" + textBox3.Text + ".txt");

time = Convert.ToInt32(city.ReadLine());

city.Close();

}

else

{

StreamWriter city7 = new StreamWriter("Города/" + textBox3.Text + ".txt");

city7.WriteLine(0);

time = 0;

city7.Close();

}

if (flag)

{

StreamWriter city2 = new StreamWriter("Города/" + textBox3.Text + ".txt");

city2.WriteLine(time + Convert.ToInt32(textBox5.Text));

city2.Close();

StreamReader city6 = new StreamReader("Города/" + old\_city + ".txt");

old\_time = Convert.ToInt32(city6.ReadLine());

city6.Close();

StreamWriter city4 = new StreamWriter("Города/" + old\_city + ".txt");

city4.WriteLine(old\_time - old\_city\_time);

city4.Close();

}

else

{

StreamWriter city3 = new StreamWriter("Города/" + textBox3.Text + ".txt");

city3.WriteLine(Convert.ToInt32(time) - old\_city\_time + Convert.ToInt32(textBox5.Text));

city3.Close();

}

Close();

MessageBox.Show(

"Квитанция изменена",

"Сообщение",

MessageBoxButtons.OK

);

Form2 newForm = new Form2();

newForm.Show();

}

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

{

FileInfo fileInf = new FileInfo("Квитанции/" + id + ".txt");

fileInf.Delete();

StreamReader reader = new StreamReader("Квитанции.txt");

string kvitations = reader.ReadToEnd();

reader.Close();

kvitations = kvitations.Replace(Program.exchange2, "Квитанция " + id.ToString() + " удалена");

StreamWriter writer = new StreamWriter("Квитанции.txt");

writer.Write(kvitations);

writer.Close();

StreamReader city = new StreamReader("Города/" + textBox3.Text + ".txt");

int time = Convert.ToInt32(city.ReadLine());

city.Close();

StreamWriter city2 = new StreamWriter("Города/" + textBox3.Text + ".txt");

city2.WriteLine(Convert.ToInt32(time) - old\_time);

city2.Close();

MessageBox.Show(

"Квитанция удалена",

"Сообщение",

MessageBoxButtons.OK

);

Close();

Form2 newForm = new Form2();

newForm.Show();

}

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

{

StreamWriter file = new StreamWriter("Буфер обмена.txt");

file.WriteLine(dateTimePicker1.Value.ToString().Replace("0:00:00", "").Replace(" ", ""));

file.WriteLine(textBox3.Text);

file.WriteLine(textBox4.Text);

file.WriteLine(textBox5.Text);

file.WriteLine(textBox6.Text);

file.WriteLine(textBox1.Text);

file.WriteLine(textBox2.Text);

file.Close();

MessageBox.Show(

"Квитанция скопирована",

"Сообщение",

MessageBoxButtons.OK

);

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

flag = true;

}

}

}

Приложение Б

(справочное)

Экранные формы

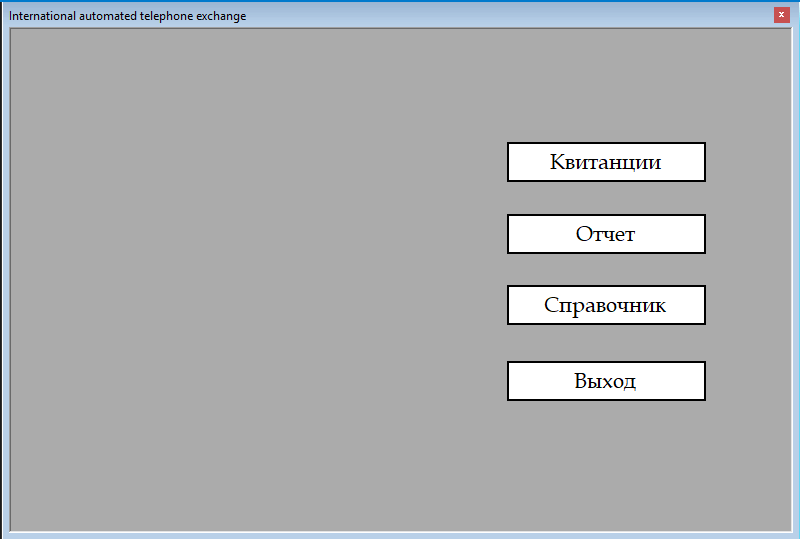


Рисунок Б.1 – Form1

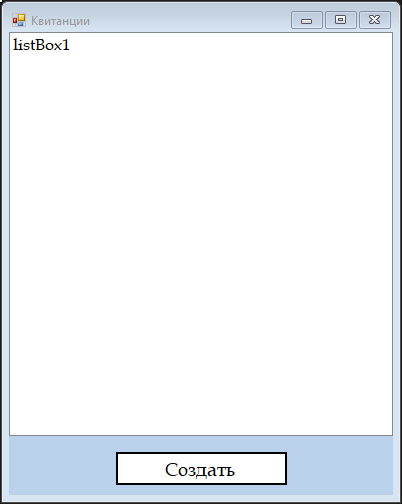


Рисунок Б.2 – Form 2

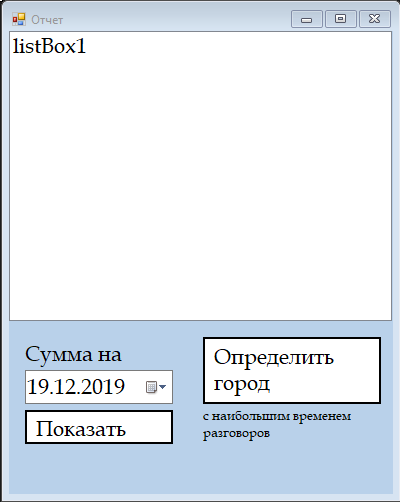


Рисунок Б.3 – Form 3

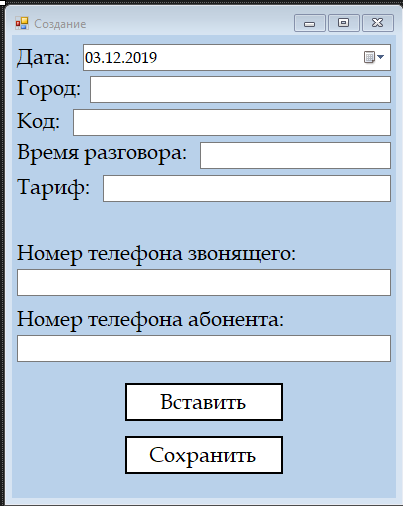


Рисунок Б.4 – Form 4

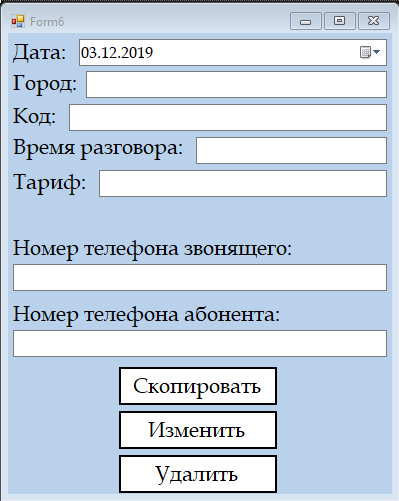


Рисунок Б.5 – Form 6