МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

КУРСОВОЙ ПРОЕКТ
ЗАЩИЩЕН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Старший преподаватель		Е.О.Шумова
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОМУ ПРОЕКТУ

Разработка приложения для организации взаимодействия объектов при заданных критериях

по дисциплине: ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	М011		П.Н.Казакова
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

Задание
на курсовой проект по дисциплине
«Объектно-ориентированное программирование»

Студенту группы М011 (№ группы) П.Н.Казакова (ФИО)

Тема «Разработка приложения для организации взаимодействия объектов при заданных критериях»

Исходные данные: Разработка системы классов для обеспечения работы аптеки

Проект должен содержать:

- анализ предметной области
- разработку классов
- разработку тестового приложения
- оформление пояснительной записки по результатам выполнения проекта
- создание презентации к проекту

Срок сдачи курсового проекта 10.12.2022 г.

Руководитель курсового проекта ст.преп. Е.О.Шумова

Дата выдачи задания 01.09.2022 г.

Содержание

Введение

Аптека — особая специализированная организация системы здравоохранения, занимающаяся изготовлением, фасовкой, анализом и продажей лекарственных средств.

Данная организация будет необходима всегда, потому что каждый год болеет множество людей, тем самым они нуждаются в приобретении лекарственных средств, назначенных врачом.

Работа посвящена разработке программного обеспечения информационной системы аптеки, позволяющего облегчить работу провизора, связанную с ручным поиском необходимого лекарства, запоминанием наличия препаратов, а также учете купленных товаров.

Данная система выполняет основную задачу – оформление заказа лекарственных средств со стороны пользователя. Поиск товаров можно осуществить на основе показаний, цене или названию лекарства. Со стороны провизора аптеки выполняется функция продажи лекарства, передачи заказа покупателю, а также учет купленных препаратов.

1 Постановка задачи

1.1 Анализ предметной области

Рассматриваемая предметная область проекта – аптека. Взаимодействующие между собой, покупатель и фармацевт являются субъектами, составляющими основу предметной области, а лекарство представляет собой используемый объект.

В связи с этим необходимо разработать удобную систему для пользователя, содержащую следующие сущности:

- Лекарственные препараты
- Заказы
- Корзина
- Учет лекарств

1.2 Формулировка технического задания

Для разрабатываемого программного продукта можно выделить следующие функциональные требования:

1. В приложении должно быть предусмотрено два режима использования: режим фармацевта аптеки и режим покупателя;
2. Приложение должно обеспечивать добавление в корзину, удаление из корзины, покупку товаров со стороны фармацевта;
3. Пополнение лекарственных препаратов со стороны фармацевта
4. Сбор статистики по ходу пользования приложения;
5. Поиск необходимого товара по критериям: название, болезнь, цена.
6. Создание заказов со стороны покупателя
7. Предотвратить неправильные запросы приложению от пользователя, как при помощи уведомлений пользователя, так и заблокировав определённую функцию до выполнения необходимых требований, приводящих ее в активное состояние.

8. Вся информация должна сохраняться в базе данных и обновляться при каждом пользовательском действии.

2 Проектирование классов

2.1 Классы сущностей

К классам сущностей относятся классы:

- preparation (id, название, показание для применения, цена и количество в наличии),
- Analitic (id, название, цена за штуку, количество, дата, общая стоимость),
- Order (номер заказа, номер телефона, название, цена, статус заказа),
- basket (id, название, цена, количество, общая стоимость),

описывающие основные свойства препарата, статистики, заказа и корзины соответственно.

2.2 Управляющие классы

В приложении также были реализованы управляющие классы:

- preparations_list (производный от класса preparation, содержит поле List<preparation> assorty = new List<preparation>()),
- analitic_list (производный от класса Analitic, содержит поле List<Analitic> bas = new List<Analitic>()),
- order_list (производный от класса order, содержит поле List<order> bas = new List<order>())
- basket_list (производный от класса basket, содержит поле List<basket> bas = new List<basket>())
- абстрактный класс Check

Эти производные классы используются для осуществления управления классами сущностей и создания списка, содержащего объекты классов сущностей (классы сущностей preparation, Analitic, order basket соответственно).

2.3 Интерфейсные классы

Для реализации интерфейса приложения создано 4 класса форм. FormApteka представляет собой главную форму для выбора возможного функционала. С данной формы можно непосредственно перейти на вкладки formAdmin и formClient для выполнения функционала фармацевта и покупателя соответственно. Так же имеется класс formHelp представляющий собой форму со справкой по использованию приложения.

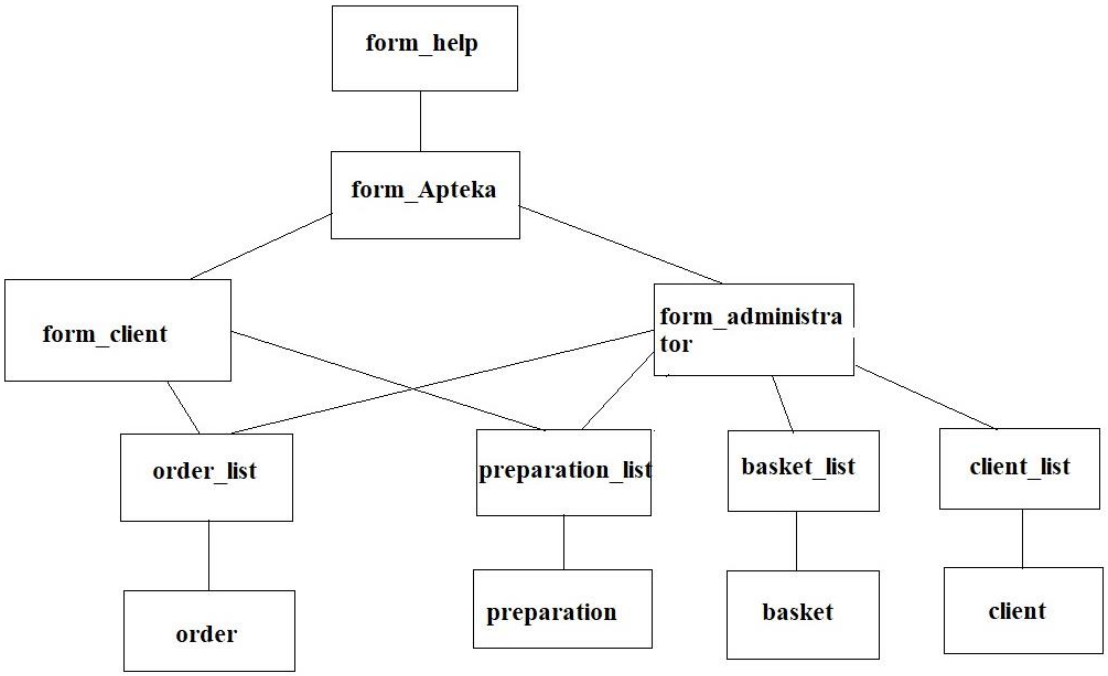

Рисунок 2.1. Структура классов

3 Разработка приложения

3.1. Разработка интерфейса приложения

Пользовательский интерфейс приложения реализуется на языке C# с платформой .NET Framework в среде разработки Microsoft Visual Studio на основе сформулированного ТЗ.

Так как одним из функциональных требований приложения является возможность двух режимов (провизора и покупателя), располагаем на форме form_Apteka (главной форме приложения) две кнопки (элементы управления Button), при нажатии на одну из которых, создается форма провизора или покупателя:

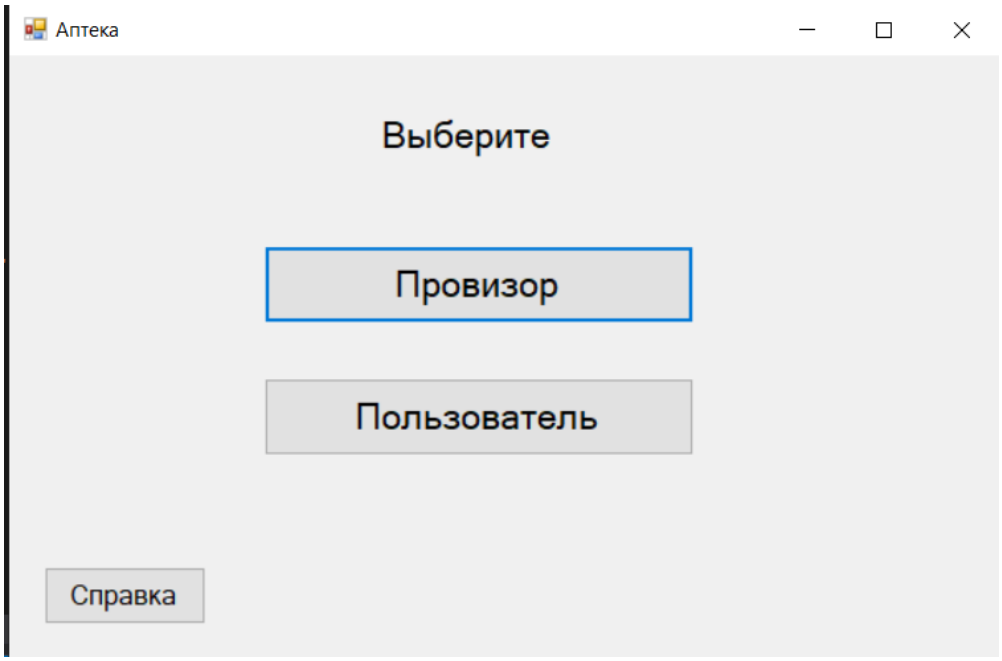

Рисунок 3.1. Главная форма приложения

Также на главной форме, в соответствии с поставленными функциональными требованиями, располагаем кнопку подсказки, при нажатии на которую создаем объект класса form_help (форма).

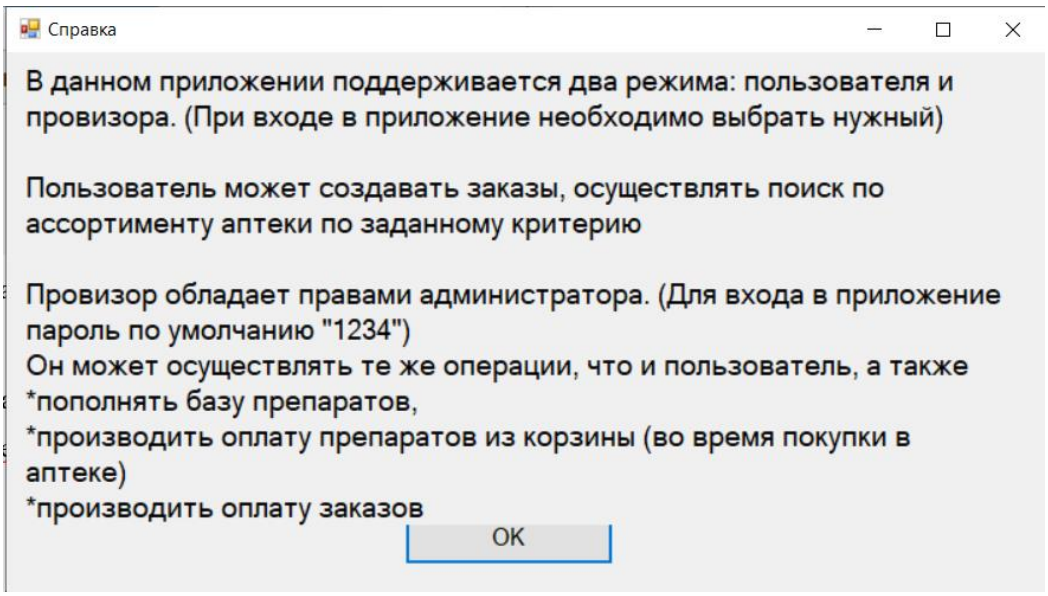

Рисунок 3.2. Форма со справкой

Для реализации остальных функциональных требований приложения на формы, отвечающие за действия покупателя и провизора, добавим элемент-контейнер TabControl (вкладки).

На форме покупателя расположим 3 вкладки: «Меню», «Корзина», «Справка».

На вкладке «Меню» и «Корзина» добавим элементы управления DataGridView, выводящие данные из хранилища DataSet на форму (таблицы «Препараты» и «Корзина» соответственно).

Для поиска лекарства на вкладку «Меню» добавим элемент управления TextBox, предоставляющий пользователю ввести необходимое данное (название, назначение, цену), а также элементы управления ComboBox, для выбора, по какому критерию будет осуществляться поиск, и количества лекарств. Добавим также элементы управления Button (кнопки «Найти», «Добавить в корзину», «Выход в меню авторизации»).

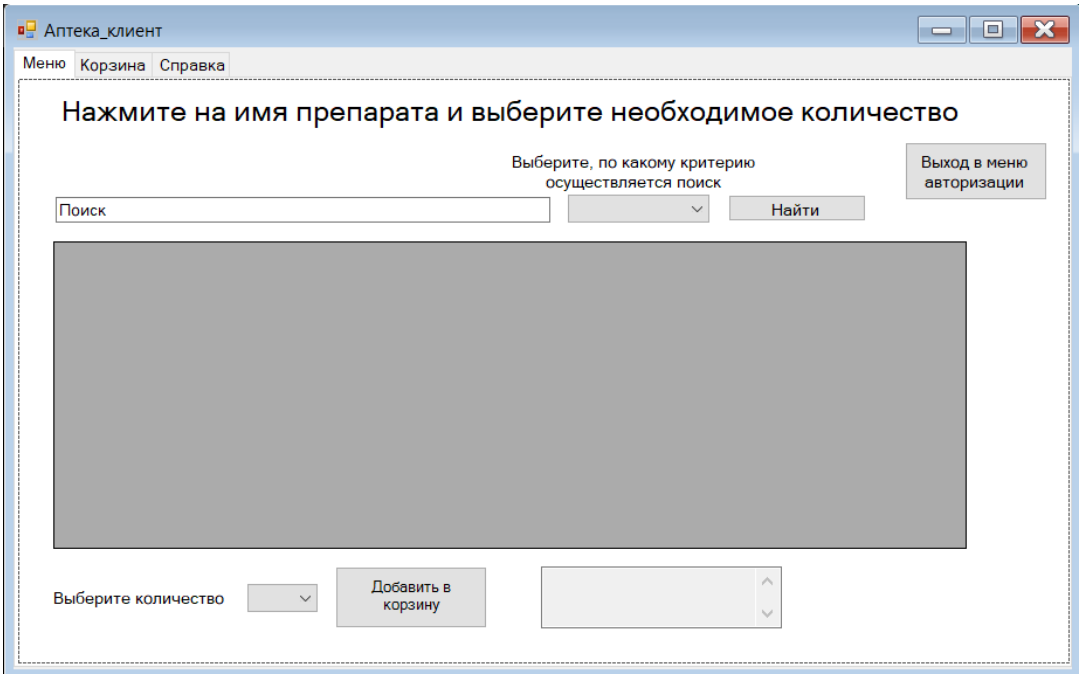

Рисунок 3.3. Вкладка «Меню» в конструкторе формы покупателя

На вкладку «Корзина» добавим элемент управления TextBox, предоставляющий пользователю ввести номер телефона, а также элементы управления Button (кнопки «Удалить выбранный товар», «Заказать»).

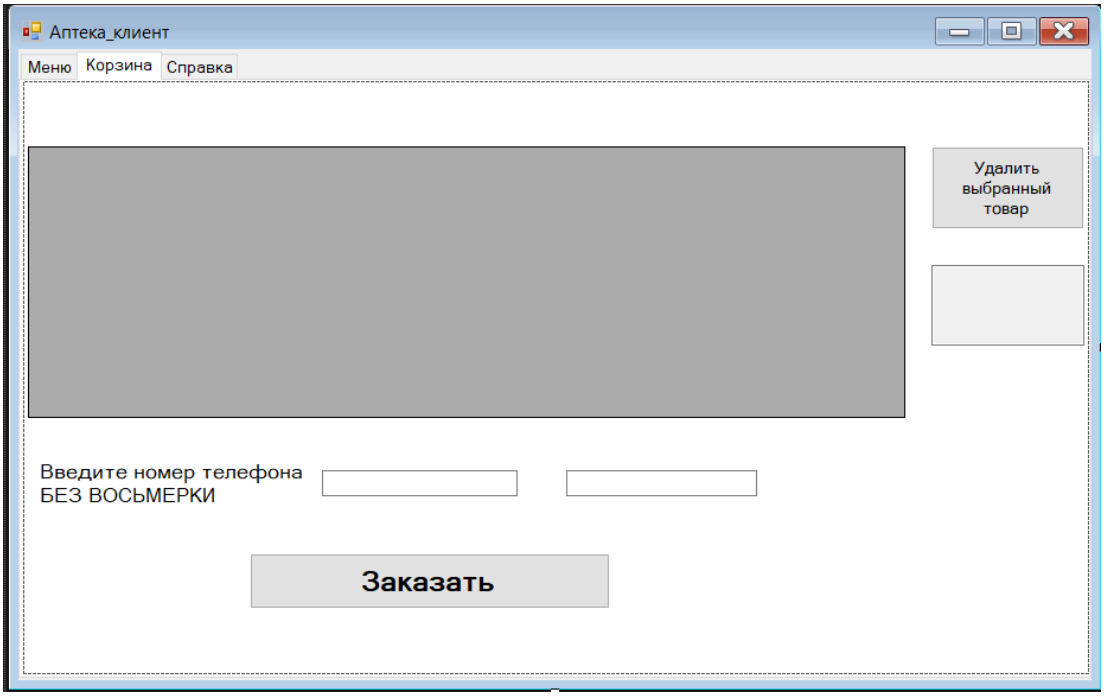

Рисунок 3.4. Вкладка «Корзина» в конструкторе формы покупателя

На вкладке «Справка» содержится информация для работы с приложением

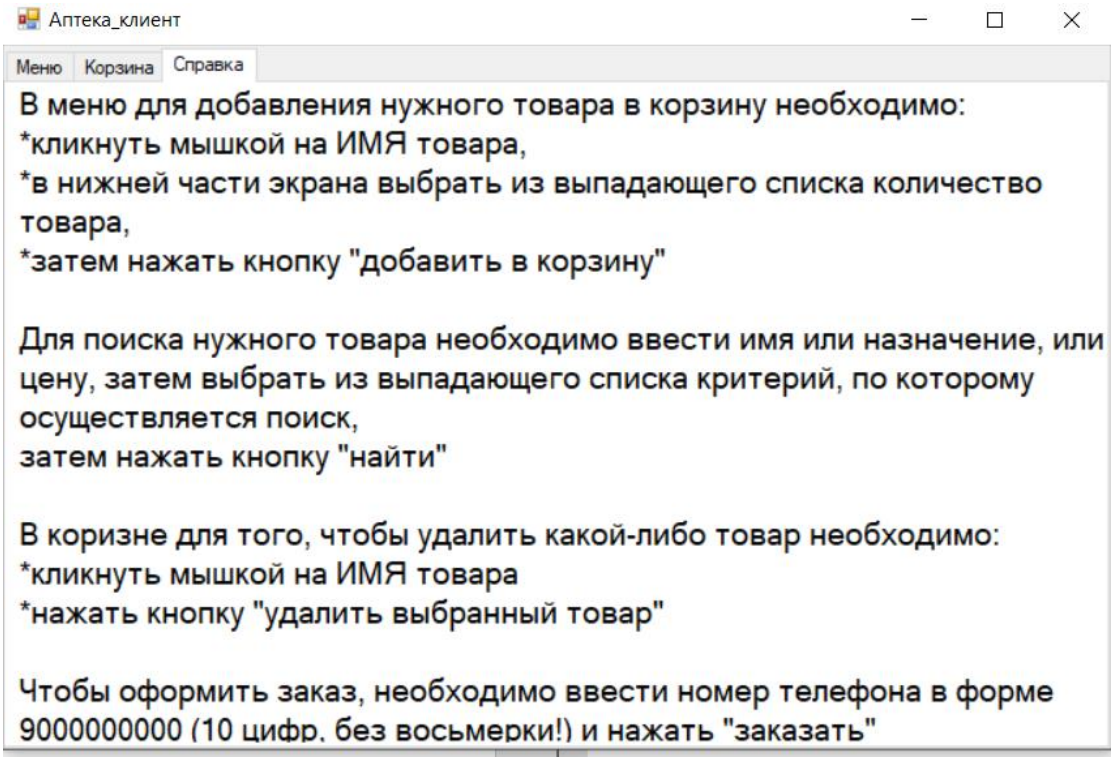

Рисунок 3.5. Вкладка «Справка» в конструкторе формы покупателя

Для входа в форму провизора добавим форму авторизации для ввода пароля администратора, содержащую элементы управления TextBox и Button (кнопка «ОК»).

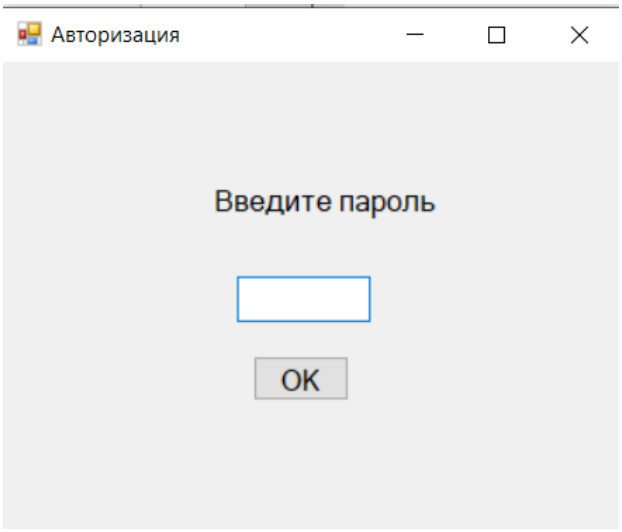

Рисунок 3.6. Форма «Авторизация»

На форме провизора расположим 6 вкладок: «Меню», «Корзина», «Заказы», «Учет», «Добавить лекарство», «Справка».

На вкладке «Меню», «Корзина», «Заказы», «Учет» добавим элементы управления DataGridView, выводящие данные из хранилища DataSet на форму (таблицы «Препараты» и «Корзина», «Заказы», «Аналитика» соответственно).

Для поиска лекарства на вкладку «Меню» добавим элемент управления TextBox, предоставляющий пользователю ввести необходимое данное (название, назначение, цену), а также элементы управления ComboBox, для выбора, по какому критерию будет осуществляться поиск, и количества лекарств. Добавим также элементы управления Button (кнопки «Найти», «Добавить в корзину», «Выход в меню авторизации»).

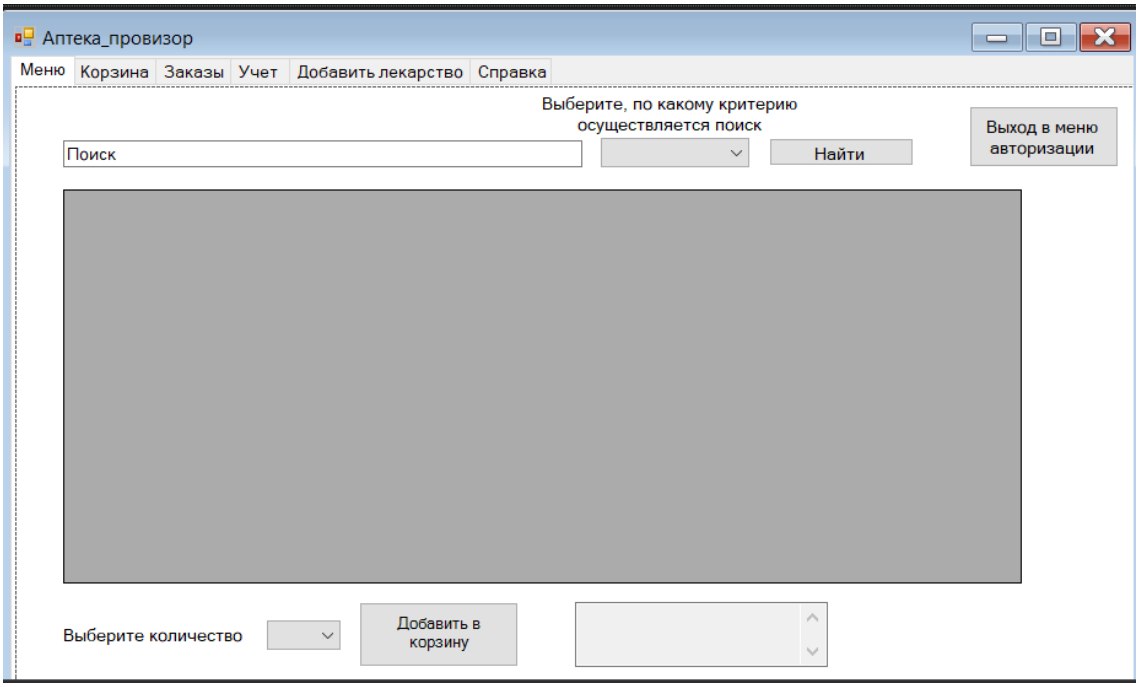

Рисунок 3.7. Вкладка «Меню» в конструкторе формы провизора

На вкладку «Корзина» добавим элемент управления Button (кнопки «Удалить выбранный товар», «Купить»).

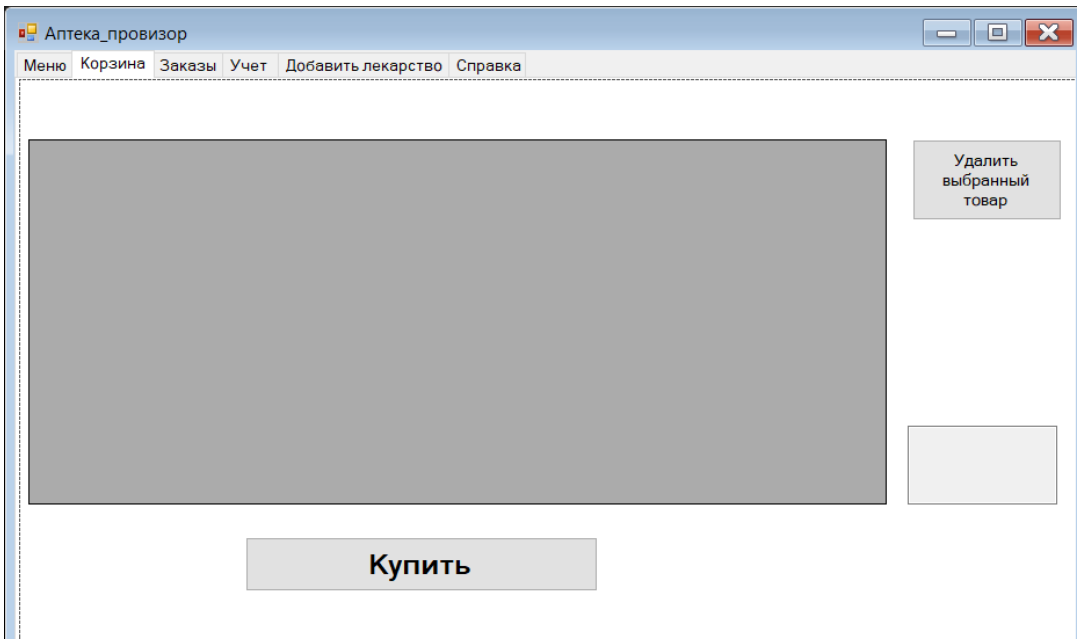

Рисунок 3.8. Вкладка «Корзина» в конструкторе формы провизора

На вкладку «Заказы» добавим элемент управления Button (кнопка «Добавить в корзину заказ»)

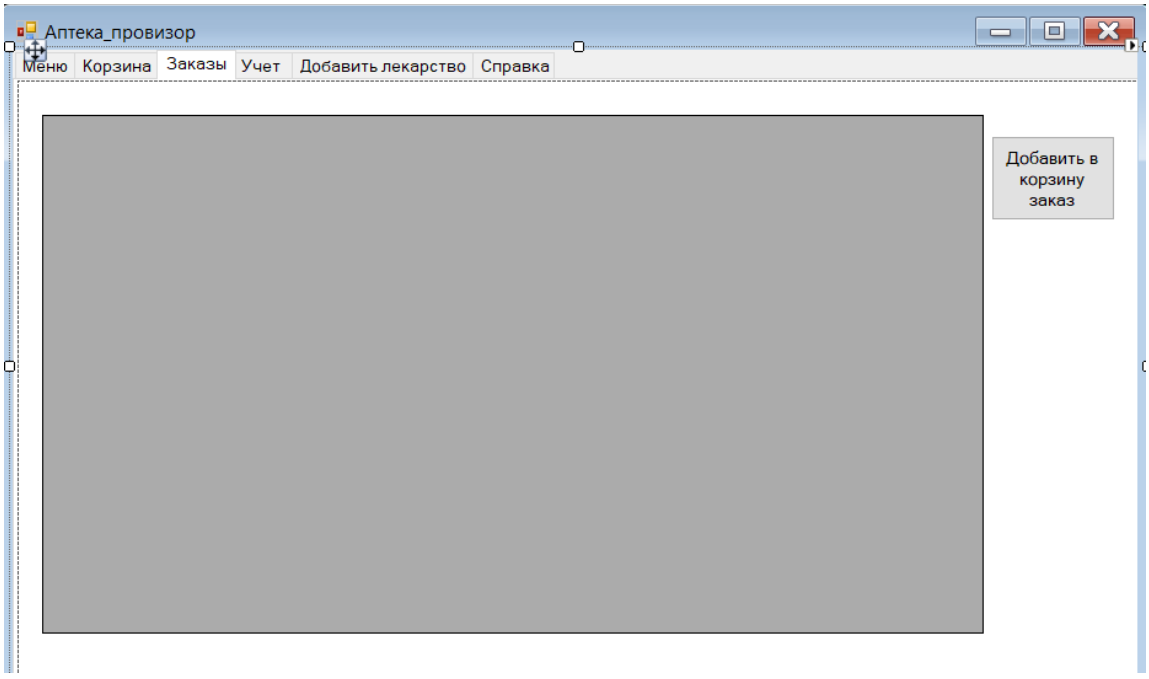

Рисунок 3.9. Вкладка «Заказы» в конструкторе формы провизора

На вкладку «Учет» добавим элемент управления TextBox, на котором будет отображаться общая сумма продажи и количество проданных лекарств.

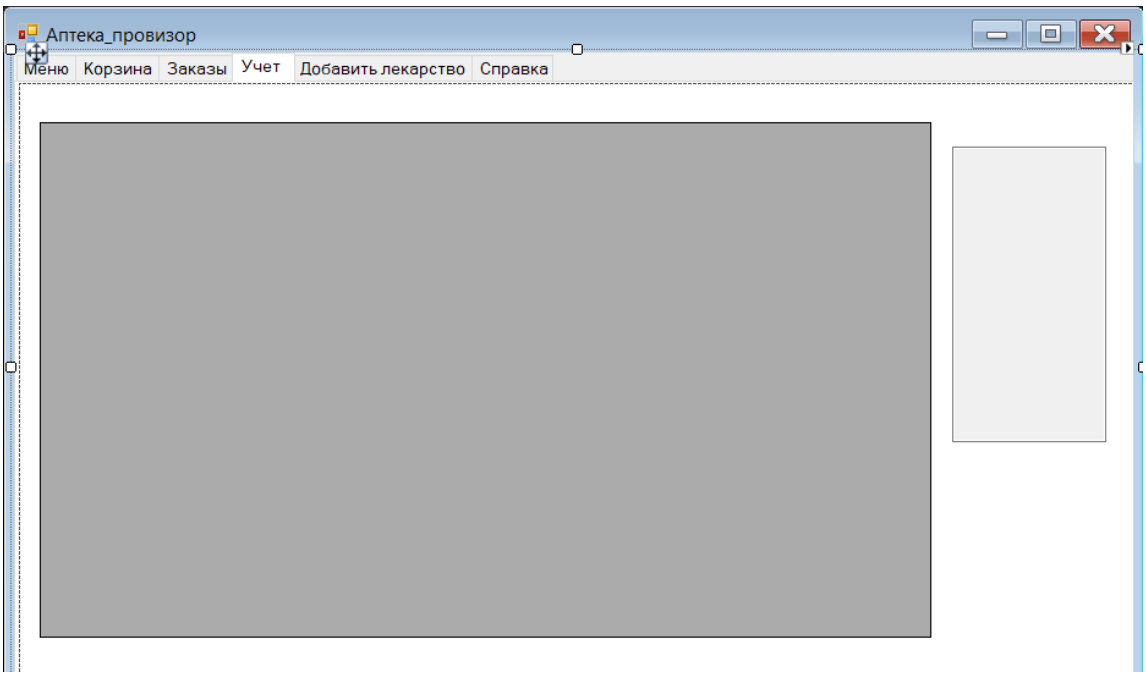

Рисунок 3.10. Вкладка «Учет» в конструкторе формы провизора

Для пополнения базы лекарств добавим на вкладку «Добавить лекарство» определенное количество элементов управления TextBox, предоставляющих провизору ввести в них необходимые данные (название, назначение, цену и количество), а также элемент управления Button (кнопка «Добавить»), фиксирующий результат ввода:

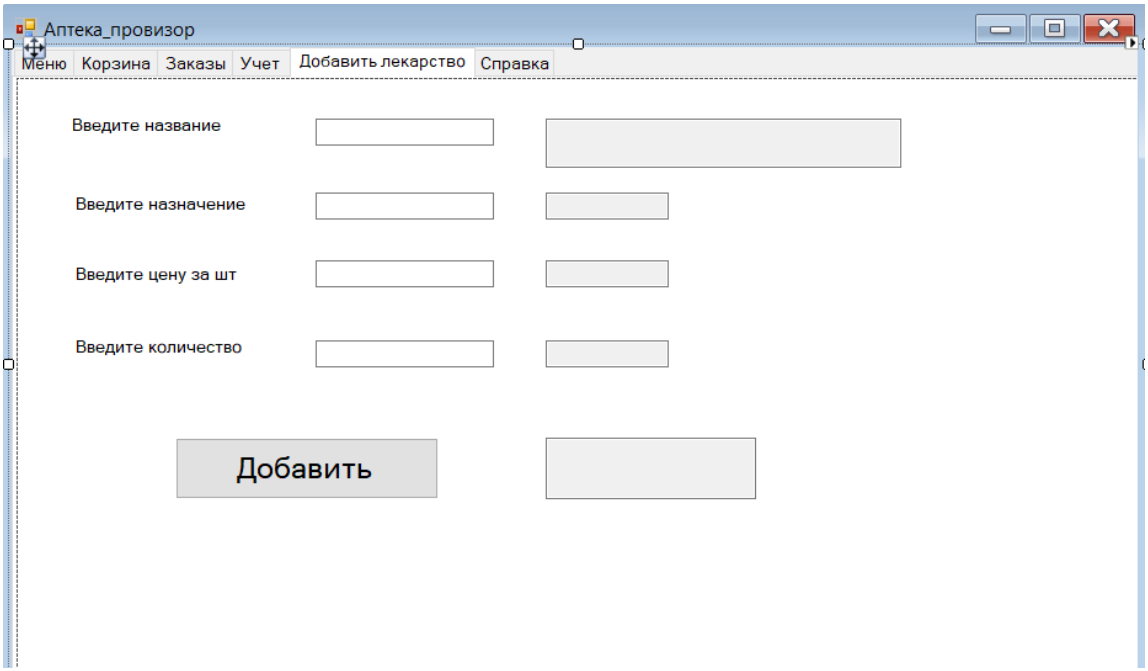

Рисунок 3.11. вкладка «Добавить лекарство» в конструкторе формы провизора

На вкладке «Справка» содержится информация для работы с приложением

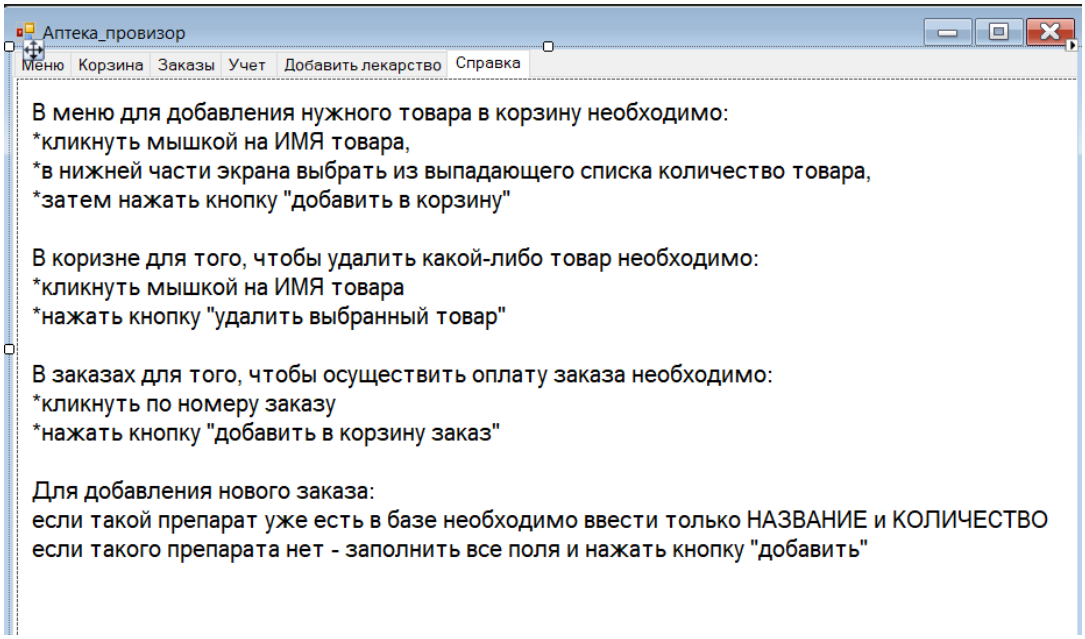

Рисунок 3.11. Вкладка «Справка» в конструкторе формы провизора

3.2. Реализация классов

Таблица 3.1. Классы и их методы

Название класса	Методы	Описание
Preparation	1. public int Id {get; set;} 2. public string Name {get; set;} 3. public string Indication { get; set; } 4. public float Price {get; set;} 5. public int Count { get; set; }	1. метод, возвращающий id препарата 2. метод, возвращающий название лекарства 3. метод, возвращающий показание к применению 4. метод, возвращающий цену препарата 5. метод, возвращающий количество оставшихся лекарств
Basket	1.public int Id { get; set; } 2.public float All_Price { get; set; } 3.public string Name { get; set;} 4.public float Price { get; set;} 5.public int Count { get; set; }	1. метод, возвращающий id препарата 2. метод, возвращающий общую стоимость лекарств 3. метод, возвращающий название лекарства

		4. метод, возвращающий цену препарата 5. метод, возвращающий количество, добавленных в корзину лекарств
Order	1.public int Number_order { get; set; } 2.public float All_Price { get; set; } 3.public string Phone { get; set;} 4.public string Preparations { get; set} 5.public string Status { get; set; }	1. метод, возвращающий номер заказа 2. метод, возвращающий общую стоимость заказа 3. метод, возвращающий номер телефона 4. метод, возвращающий названия лекарств в заказе 5. метод, возвращающий статус заказа
Analitic	1. public int Id { get; set; } 2. public string Name {get; set;} 3. public double Price { get; set; } 4. public int Count { get; set; } 5. public string Date {get; set;} 6. public double All_price { get; set; }	1. метод, возвращающий id препарата 2. метод, возвращающий название лекарства 3. метод, возвращающий цену препарата 4. метод, возвращающий количество проданных лекарств 5. метод, возвращающий дату продажи 6. метод, возвращающий общую сумму продажи
check	1.public bool StringCheck(string str) 2. public bool checkdigit(string num) 3. public bool checkNumber(string num)	1. метод для проверки введенной строки 2. метод для проверки введенного числа 3. метод для проверки введенного номера телефона
Preparation_list	1.public void add (preparation preparat) 2. public void add(int id, string name, string indication, float price, int count) 3. public void clear () 4. public int get_count_preparation() 5. public int getcount(int id) 6. public bool getname(string name)	1. метод добавления нового лекарства в список 2. метод добавления нового лекарства в список по параметрам 3. метод очищения списка с лекарствами 4. метод, возвращающий количество лекарств в списке

		5. метод для получения информации об остатке необходимого лекарства 6. метод для проверки наличия данного лекарства в списке
Basket_list	1. public void add(int id, string name, float price, int count, float all_price) 2. public int get_count_basket() 3. public bool get_name_basket(string name)	1. метод добавления нового лекарства в корзину по параметрам 2. метод, возвращающий количество лекарств в корзине 3. метод для проверки наличия данного лекарства в корзине
Order_list	1.public void add(int number_order, string phone, string preparations, float all_price, string status) 2. public int get_count_list_in_orders() 3.public bool get_number(string number)	1. метод добавления нового заказа в список по параметрам 2. метод, возвращающий количество заказов в списке 3. метод для проверки существующего номера заказа в списке заказов
Analitic_list	1.public void add(int id, string name, double price, int count, string date, double all_price) 2. public bool get_name_analitic(string name) 3. public int get_count_list_in_analitic() 4. public void clear() 5. public bool get_now_date_in_list(string name, string date)	1. метод добавления нового лекарства в список учета по параметрам 2. метод, возвращающий название препарата в списке учета 3. метод, возвращающий количество препаратов в списке учета 4. метод для очистки списка учета 5. метод для проверки текущей даты в списке учета

1. Класс preparation

Хранит информацию о конкретном лекарстве (id, название, показание для применения, цена и количество в наличии)

```
class preparation
    {
        int id;
        string name;
        string indication;
        float price;
```

```
        int count;

        public preparation(int id, string name, string indication, float price, int
count)
        {
            this.id = id;
            this.name = name;
            this.indication = indication;
            this.price = price;
            this.count = count;
}
```

2. Класс basket

   Хранит информацию о товаре в корзине (id, название, цена, количество и общая стоимость)

```
class basket
    {
        int id;
        string name;
        float price;
        int count;
        float all_price;

        public basket(int id, string name,float price, int count, float all_price = 0)
        {
            this.id = id;
            this.name = name;
            this.price = price;
            this.count = count;
            this.all_price = all_price;
}
```

3. Класс Analitic

   Хранит информацию о купленном товаре (название, цена, количество, дата, общая стоимость)

```
class Analitic
    {
        string name;
        double price;
        int count;
        string date;
        double all_price;
        public Analitic(string name, double price, int count, string date, double
all_price)
        {
            this.name = name;
            this.price = price;
            this.count = count;
            this.date = date;
            this.all_price = all_price;
}
```

4. Класс order

   Хранит информацию о заказе (номер заказа, номер телефона, список лекарств, сумма заказа, статус заказа)

```
       class order
           {
               int number_order;
```

```
            string phone;
            string preparations;
            float all_price;
            string status;
            public order(int number_order, string phone, string preparations, float
    all_price, string status)
            {
                this.number_order = number_order;
                this.phone = phone;
                this.preparations = preparations;
                this.all_price = all_price;
                this.status = status;
      }
```

5. Класс check

   Абстрактный класс, не имеет полей. Предназначен для проверки введенных пользователем данных

6. Класс preparation_list

   Хранит список препаратов и методы для осуществления управления списком (добавление, подсчет количества элементов в списке, очищение списка)

```
class preparations_list
    {
        public List<preparation> assorty = new List<preparation>();
        public preparations_list()
        {
            List<preparation> assorty = new List<preparation>();
}
```

7. Класс basket_list

   Хранит список товаров в корзине и методы для осуществления управления списком (добавление, подсчет количества элементов в списке, очищение списка)

```
class Basket_list
    {
        public List<basket> bas = new List<basket>();
        public Basket_list()
        {
            List<basket> bas = new List<basket>();
    }
```

8. Класс order_list

   Хранит список заказов и методы для осуществления управления списком (добавление, подсчет количества элементов, проверка на уникальность номера заказа)

```
    class order_list
        {
            public List<order> list = new List<order>();
            public order_list()
            {
                List<order> list = new List<order>();
        }
```

9. Класс analitic_list

   Хранит список купленных товаров и методы для осуществления управления списком (добавление, подсчет количества элементов в списке, удаление товара из списка)

```
class analitic_list
    {
        public List<Analitic> list = new List<Analitic>();
        public analitic_list()
        {
            List<Analitic> list = new List<Analitic>();
}
```

3.3. Разработка тестового приложения

Тестовое приложение разрабатывалось на четырех формах.

Первая форма form_Apteka отвечала за создание форм, реализующих действия провизора и покупателя.

Объект класса form_administrator создает следующие объекты и переменные:

```
public string selectedState, selectedCount, count1, name1, price1, id1, all_price,
status, order_number=""; // вспомогательные переменные для проверок и управления
preparations_list preparat = new preparations_list(); // список лекарств
Basket_list bas = new Basket_list(); // список лекарств в корзине
analitic_list list = new analitic_list(); // список проданных лекарств
order_list order = new order_list(); // список заказов
check check = new check(); // объект класса для проверки введенных данных
public float sum = 0; //общая стоимость для корзины
public double sum1; //общая стоимость для аналитики
public int count = 0;
```

Для удобного хранения и использования данных из БД используются списки созданных пользовательских классов (с помощью класса List <T> из пространства имен System.Collections.Generic).

Объект класса Form_client создает следующие объекты и переменные:

```
public string selectedState, selectedCount, count1, name1, price1, id1, all_price,//
вспомогательные переменные для проверок и управления
preparations_list preparat = new preparations_list(); // список лекарств
Basket_list bas = new Basket_list(); // список лекарств в корзине
order_list order = new order_list(); // список заказов
check check = new check(); // объект класса для проверки введенных данных
public float sum = 0; //общая стоимость для корзины
public double sum1; //общая стоимость для аналитики
```

4 Тестирование

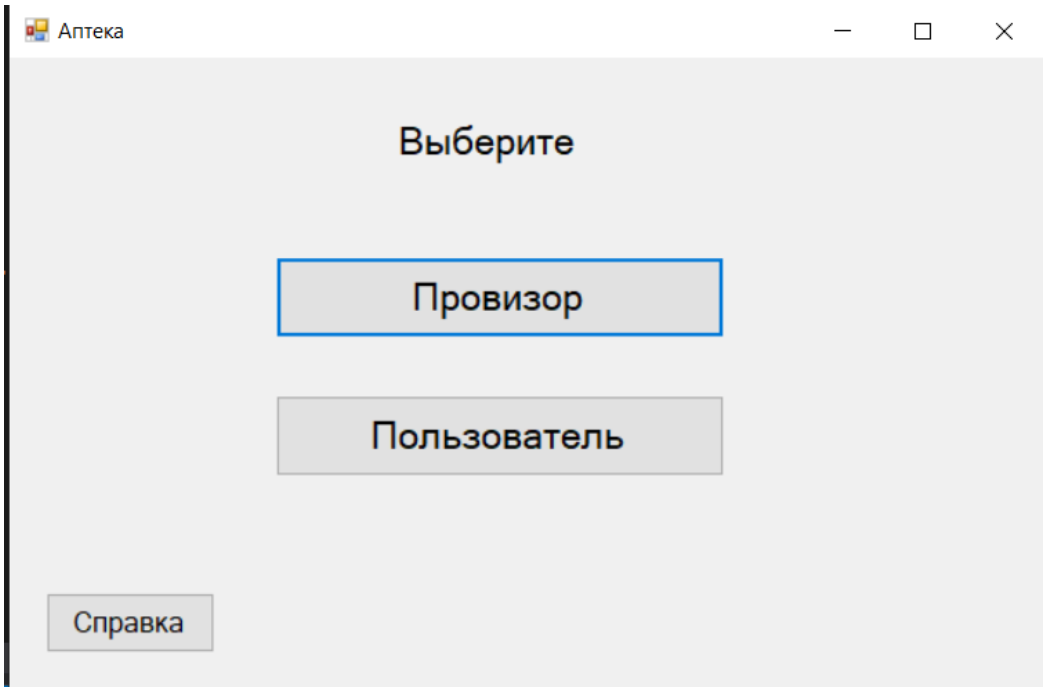

Рисунок 4.1 Форма меню при входе

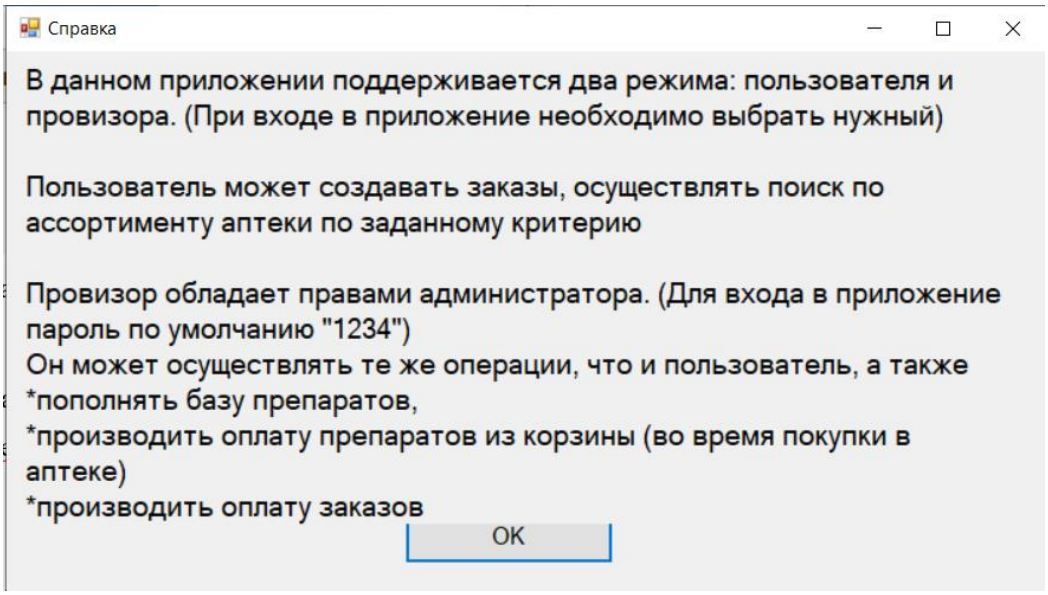

Рисунок 4.2 Нажатие кнопки "Справка"

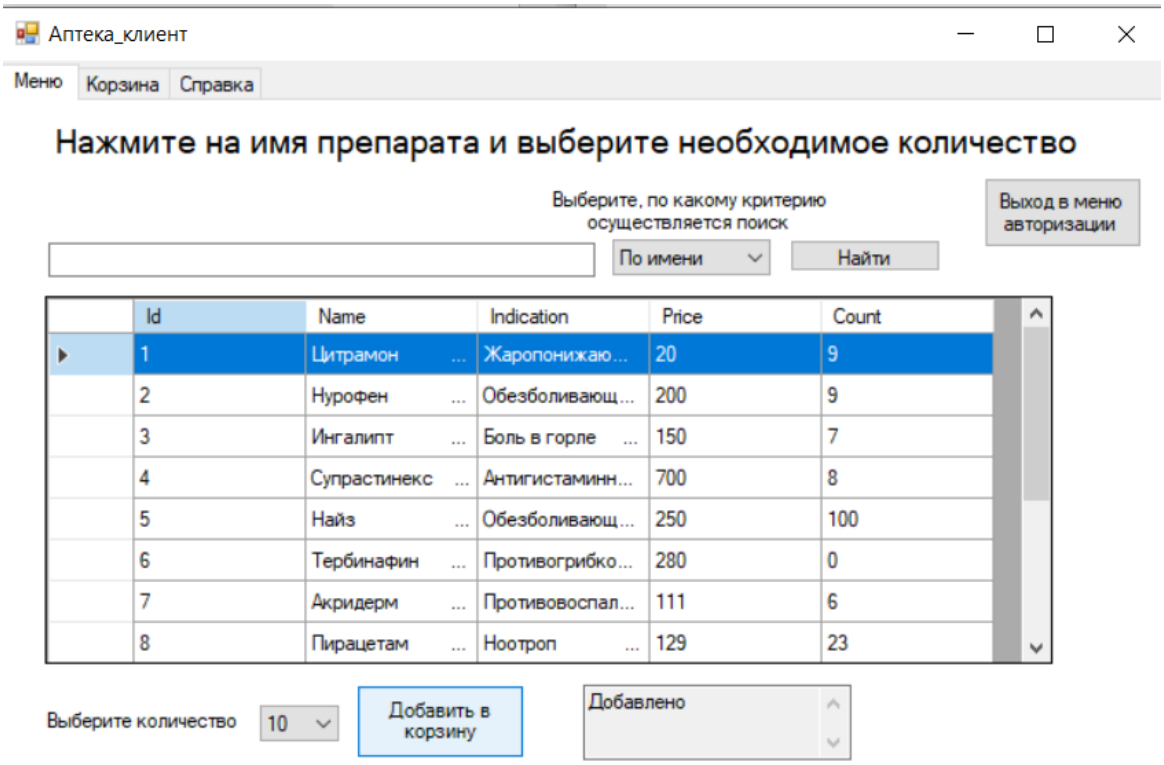

Рисунок4.2 Нажатие кнопки "Покупатель" и выбор вкладки "Меню", добавление 10 штук «Найза» в корзину

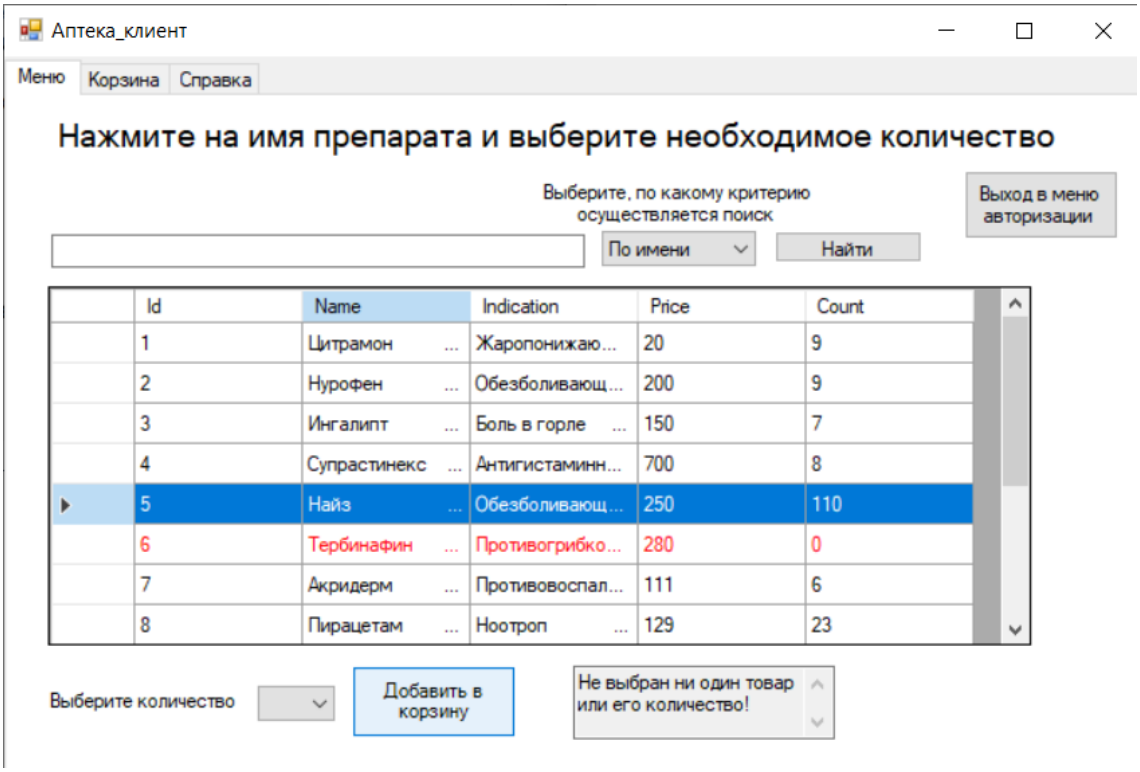

Рисунок 3.4 Попытка добавить лекарство, не выбрав его количество

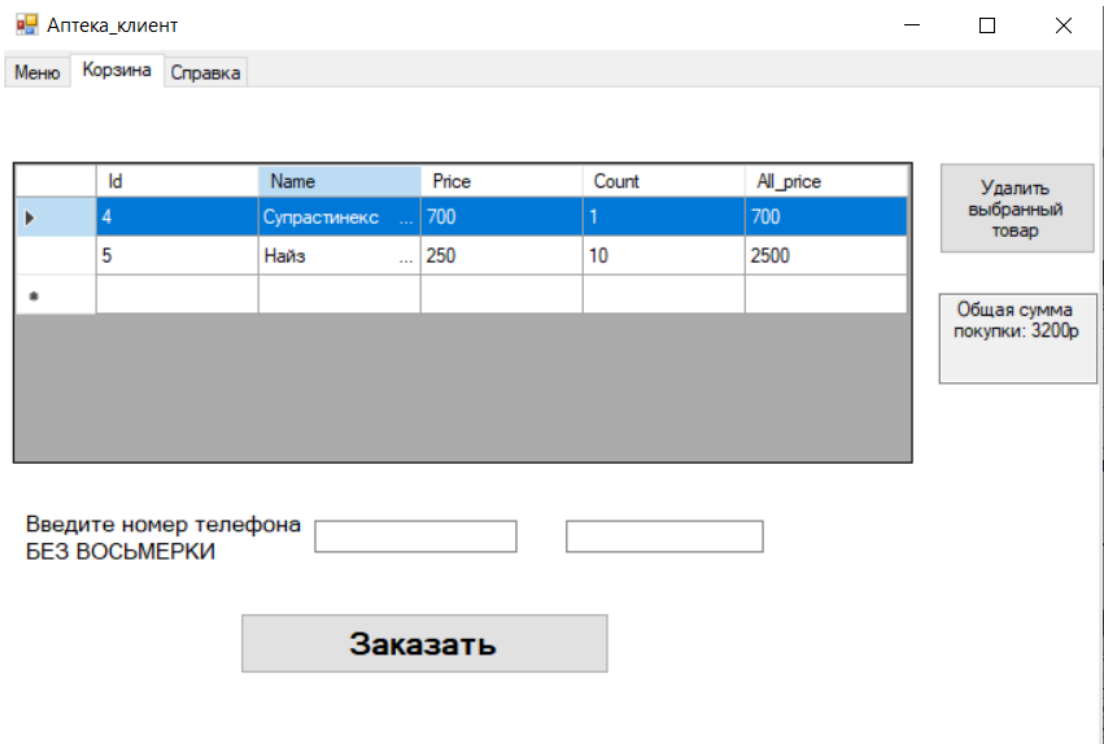

Рисунок 4.4 Вкладка корзина

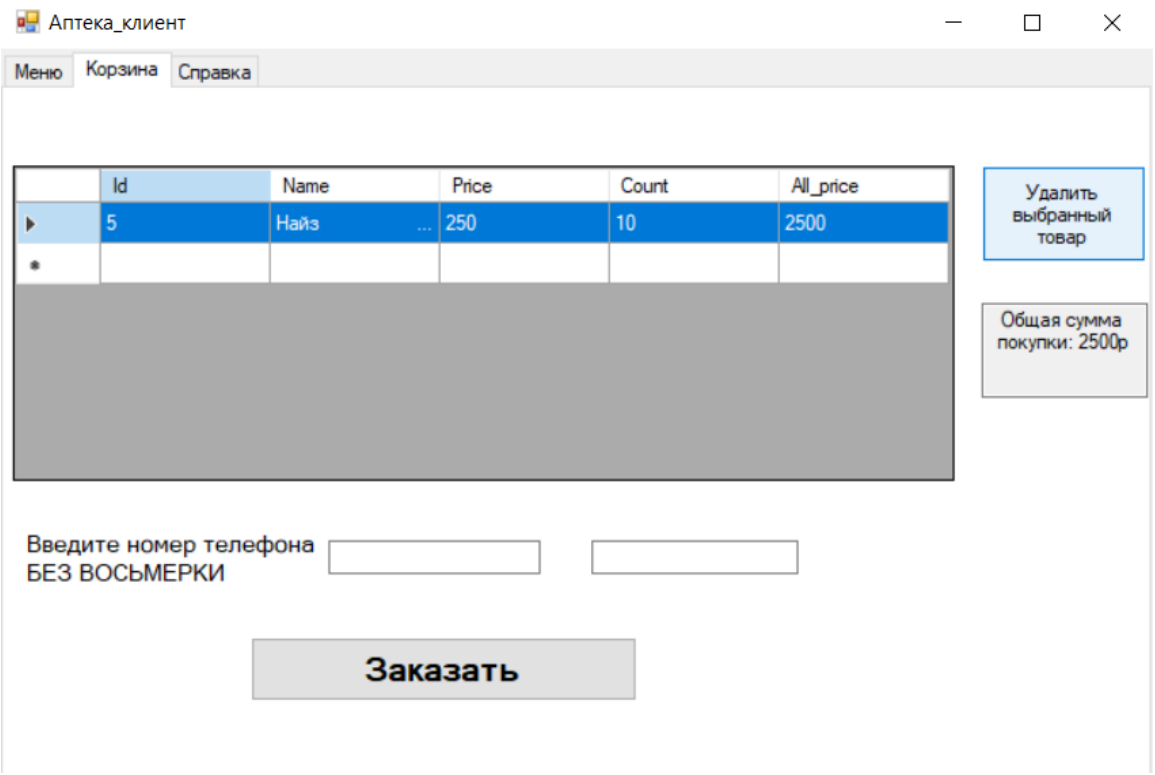

Рисунок 4.5 Нажатие кнопки "Удалить выбранный товар" ("Супрастинекс" удален)

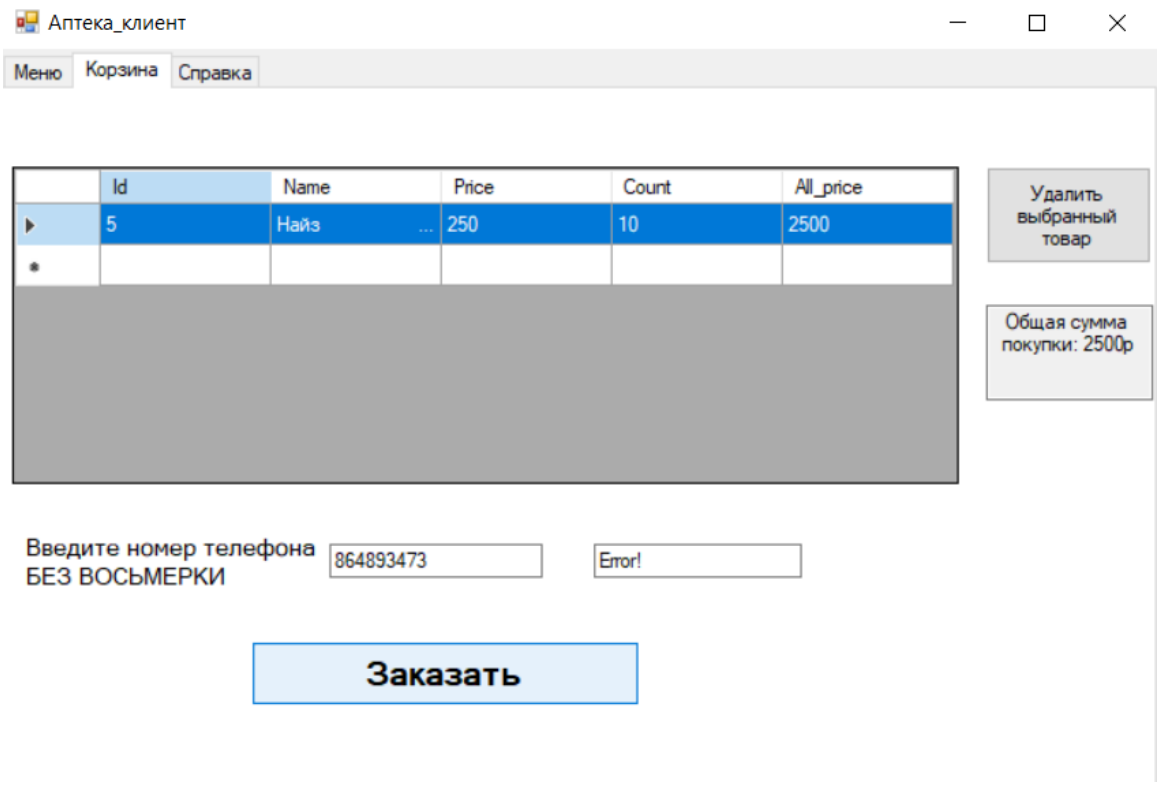

Рисунок 4.6 Ввод неверного номера телефона или пустое поле и нажатие кнопки "Заказать"

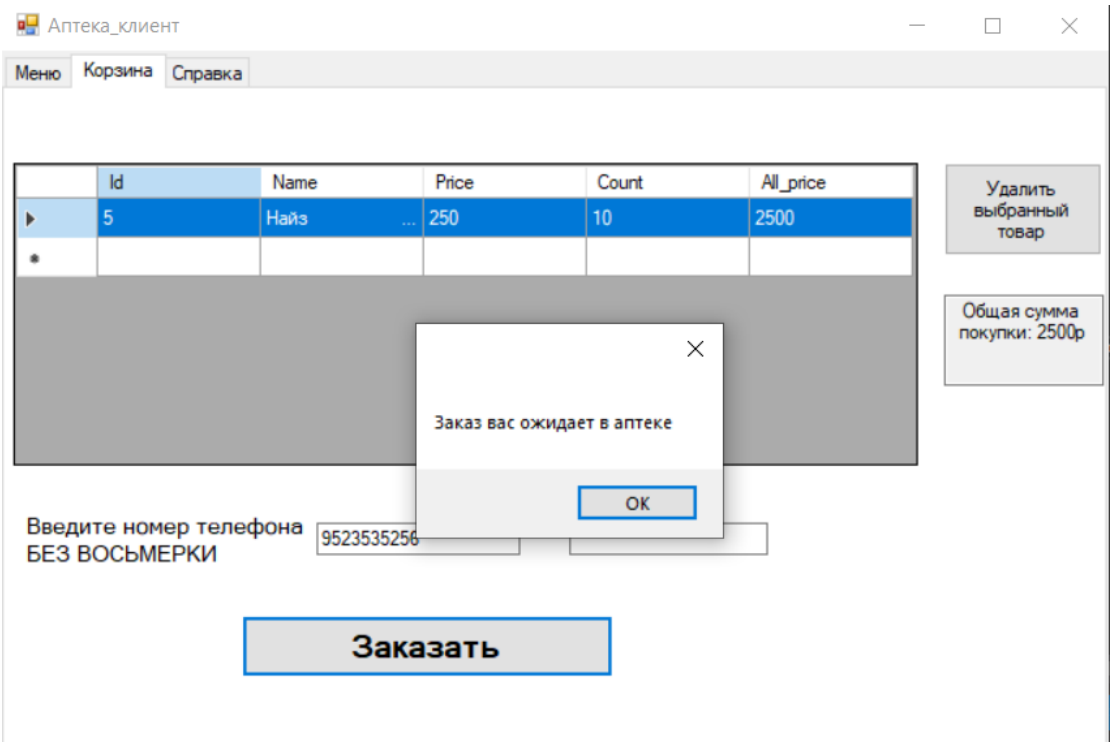

Рисунок 4.7 Оформление заказа

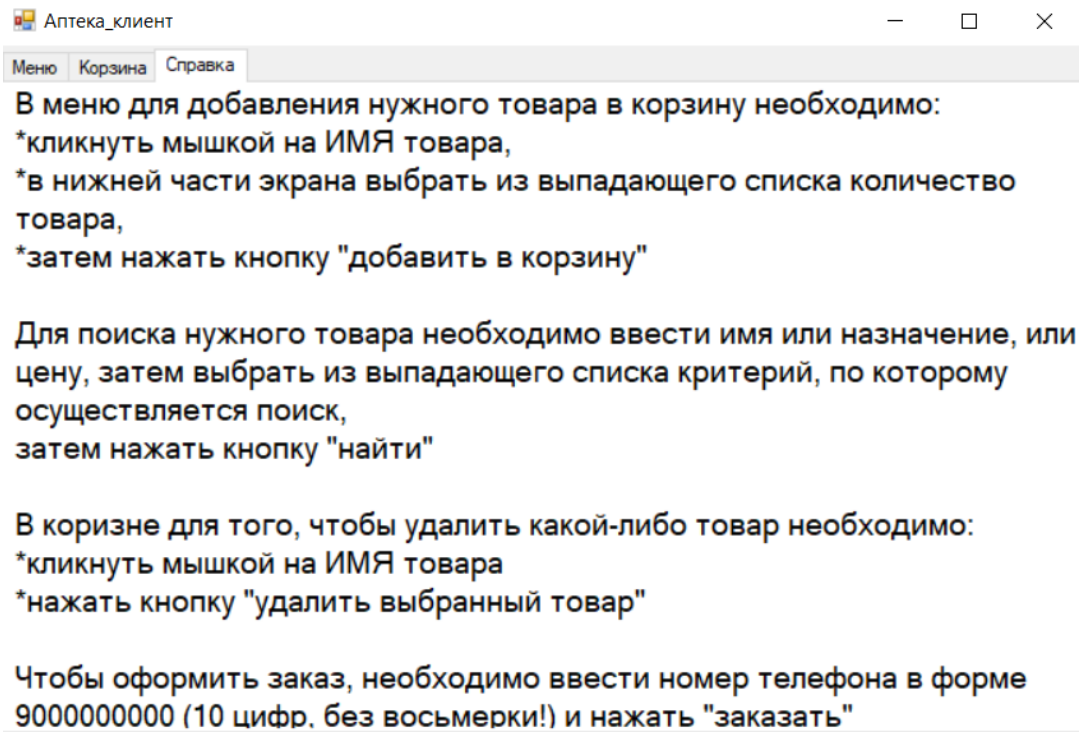

Рисунок 4.8 Вкладка "Справка"

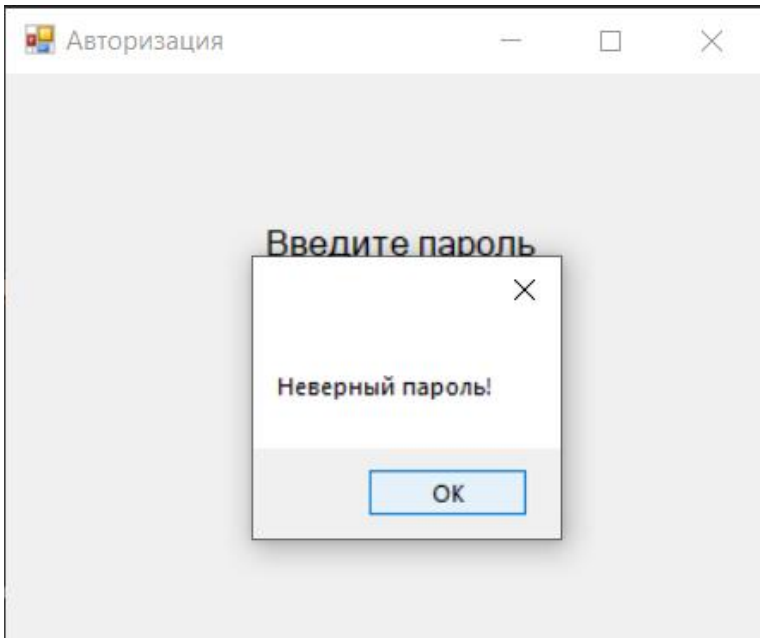

Рисунок 4.9 Нажатие кнопки "Провизор" из главного меню и ввод неверного пароля

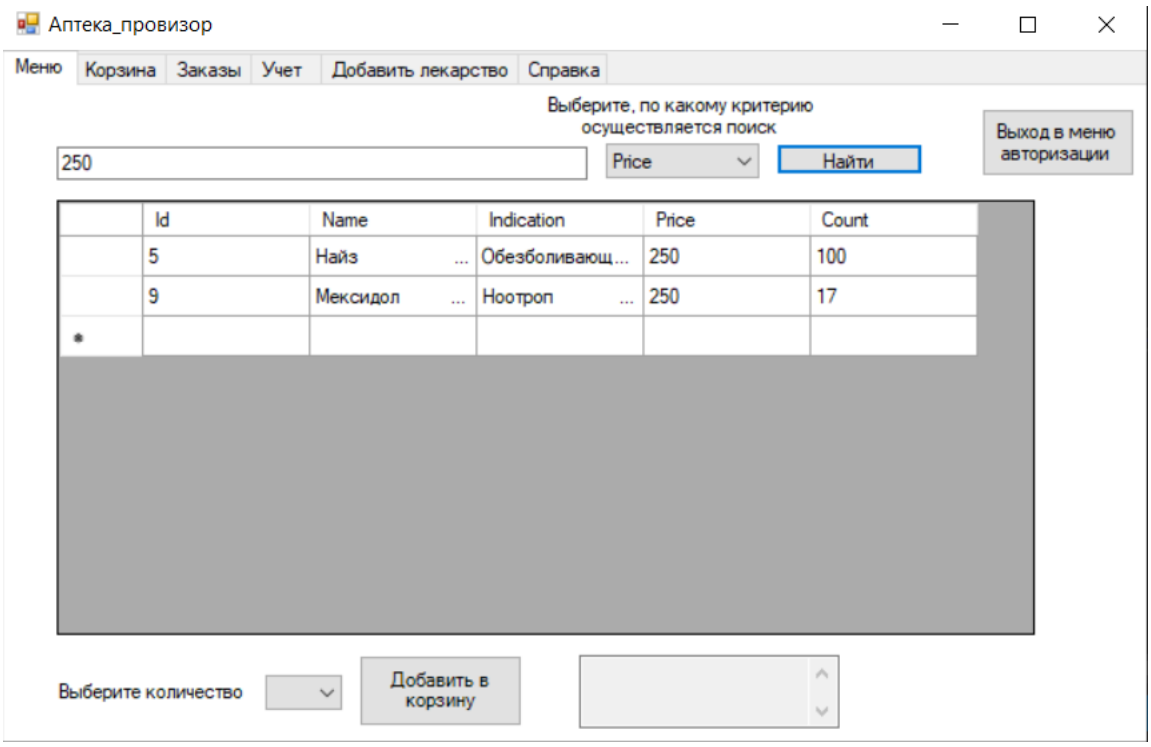

Рисунок 4.10 Поиск по цене

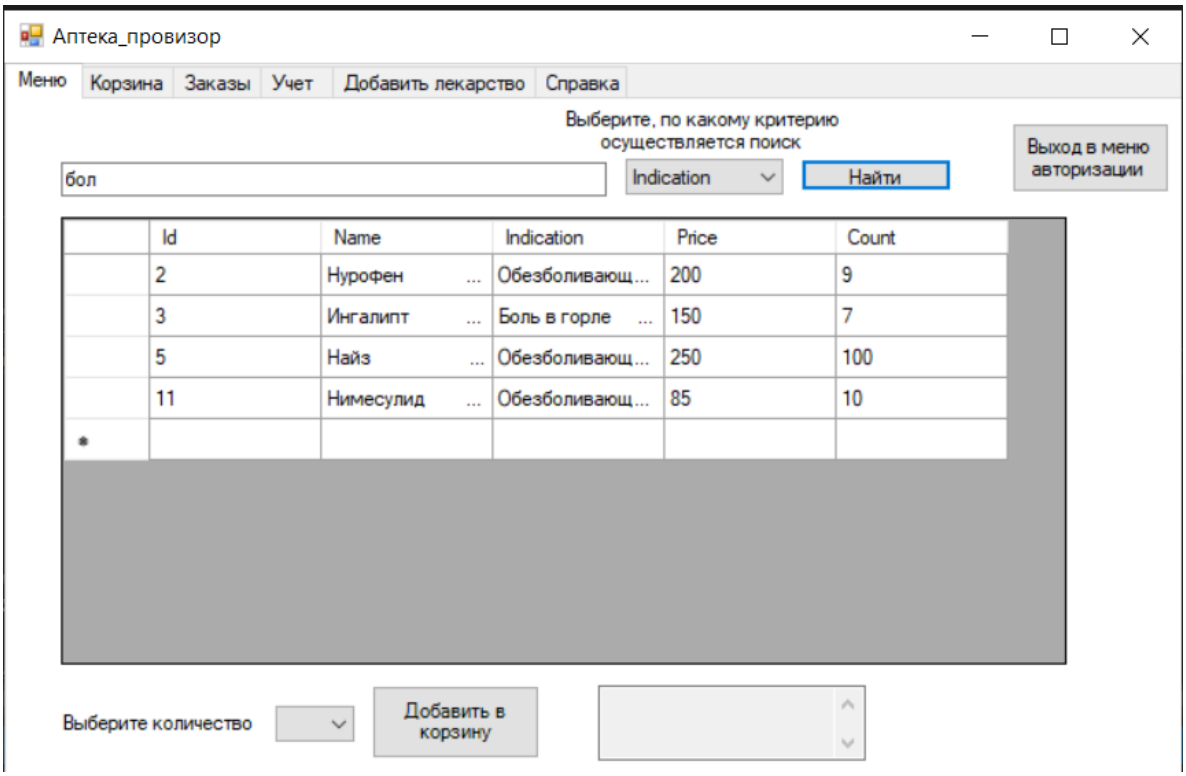

Рисунок 4.11 Поиск по назначению

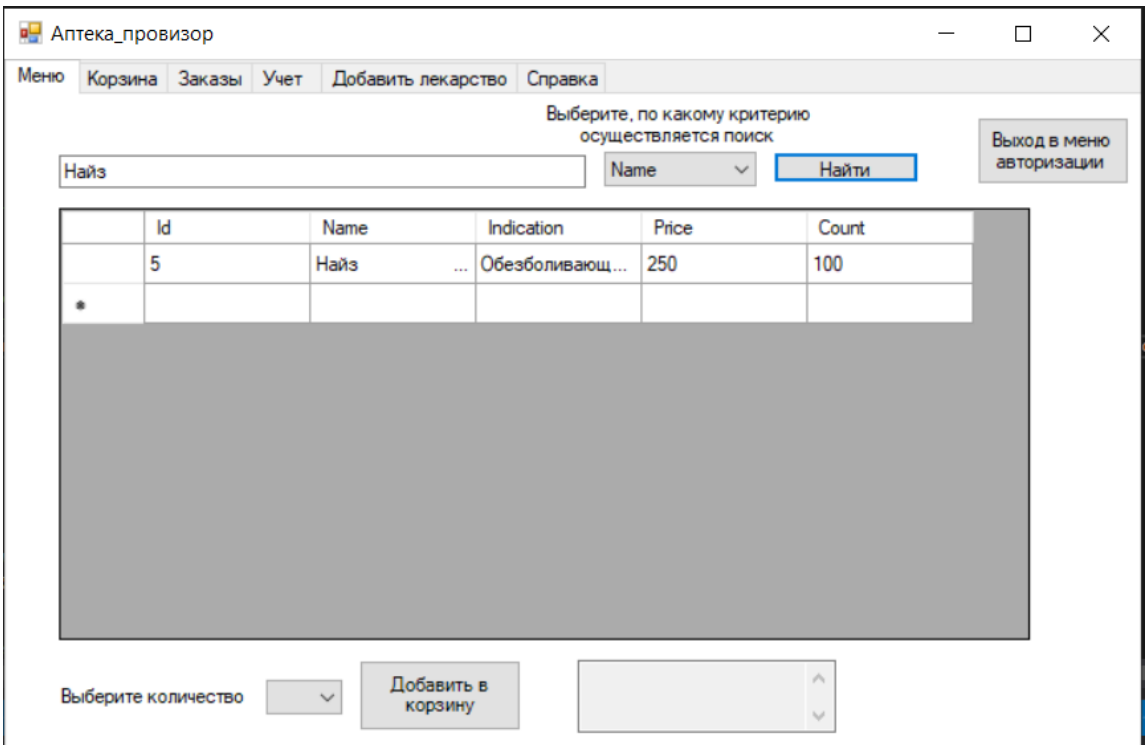

Рисунок 4.12 Поиск по названию

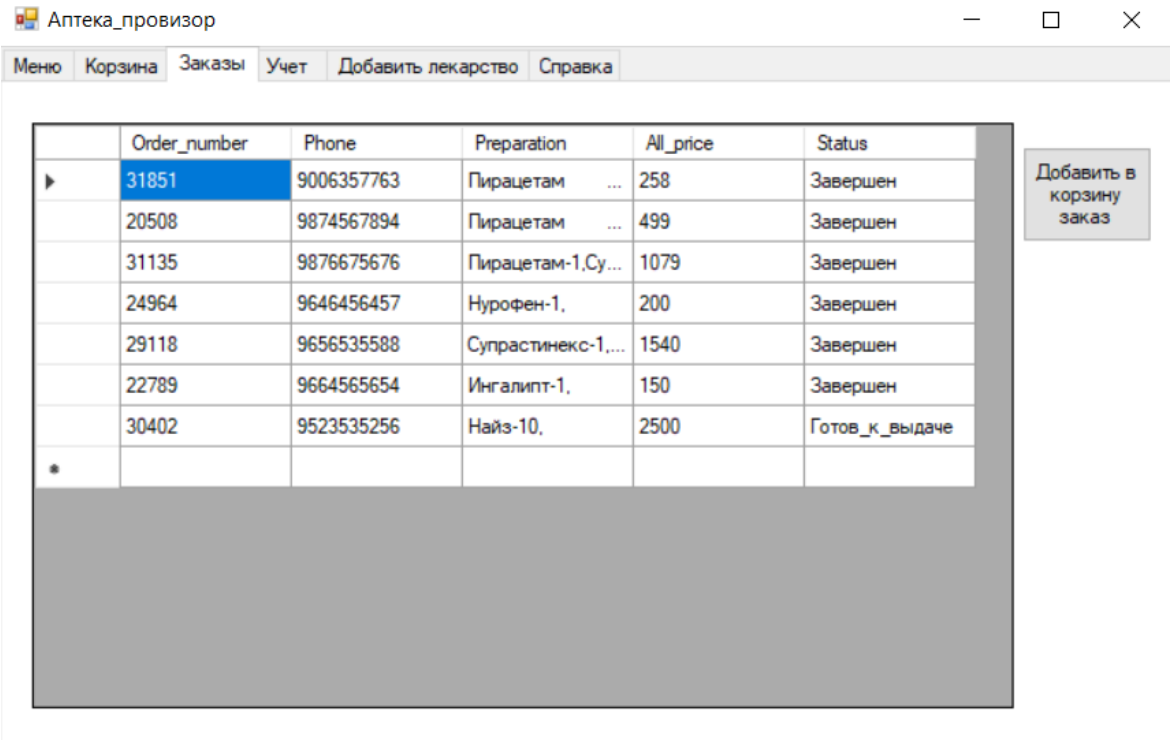

Рисунок 4.13 Вкладка "Заказы"

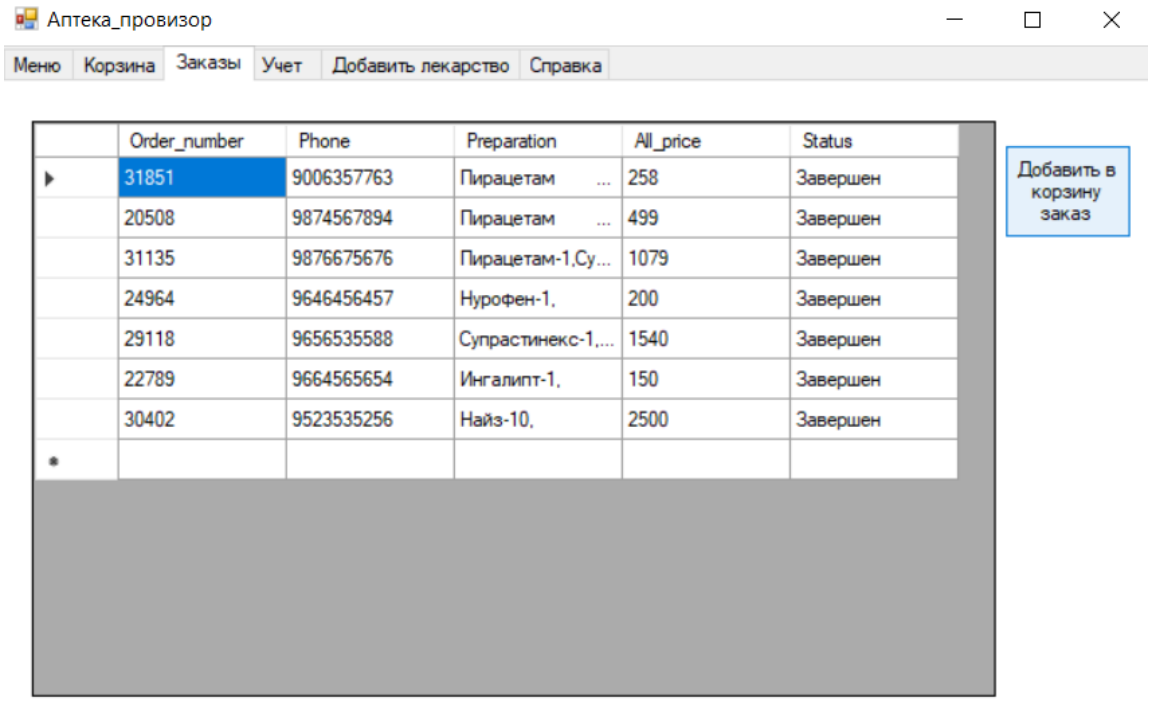

Рисунок 4.14 Выбор заказа и нажатие кнопки "Добавить в корзину заказ"

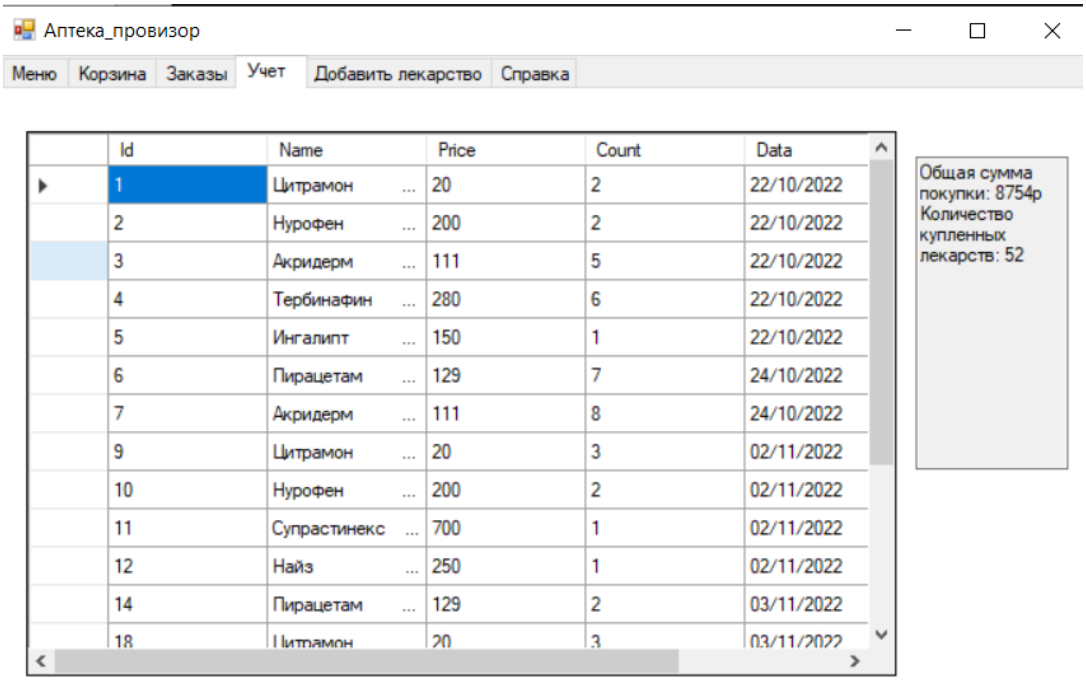

Рисунок 4.15 Выбор вкладки "Учет"

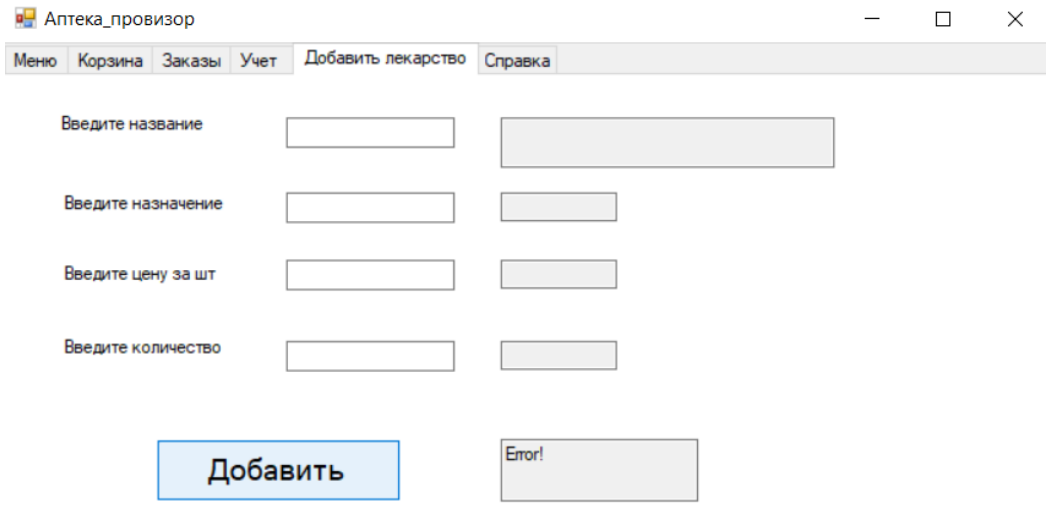

Рисунок 4.16 Вкладка "Добавить лекарство" и нажатие кнопки "Добавить" с пустыми полями

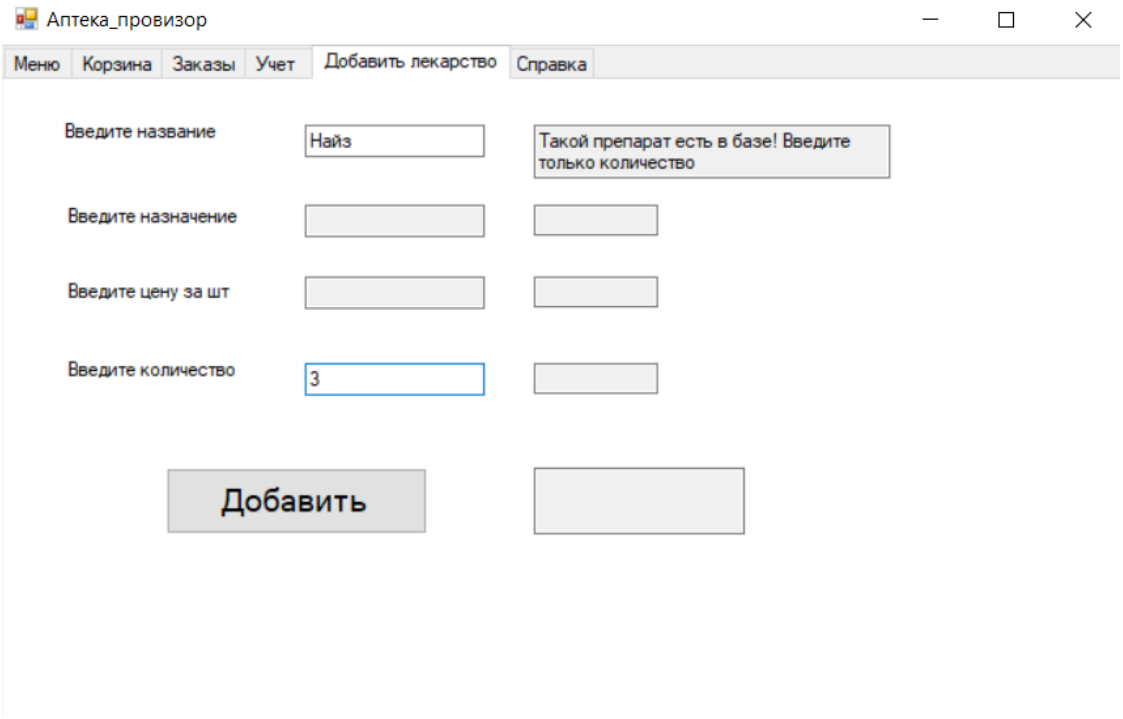

Рисунок 4.17 Добавление уже существующего лекарства

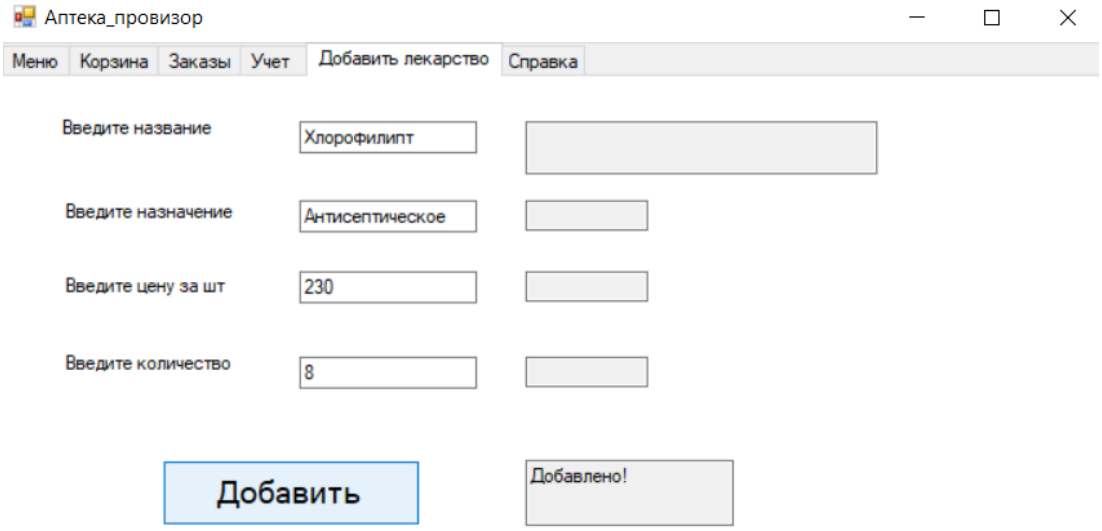

Рисунок 4.18 Добавление лекарства, введенного корректно

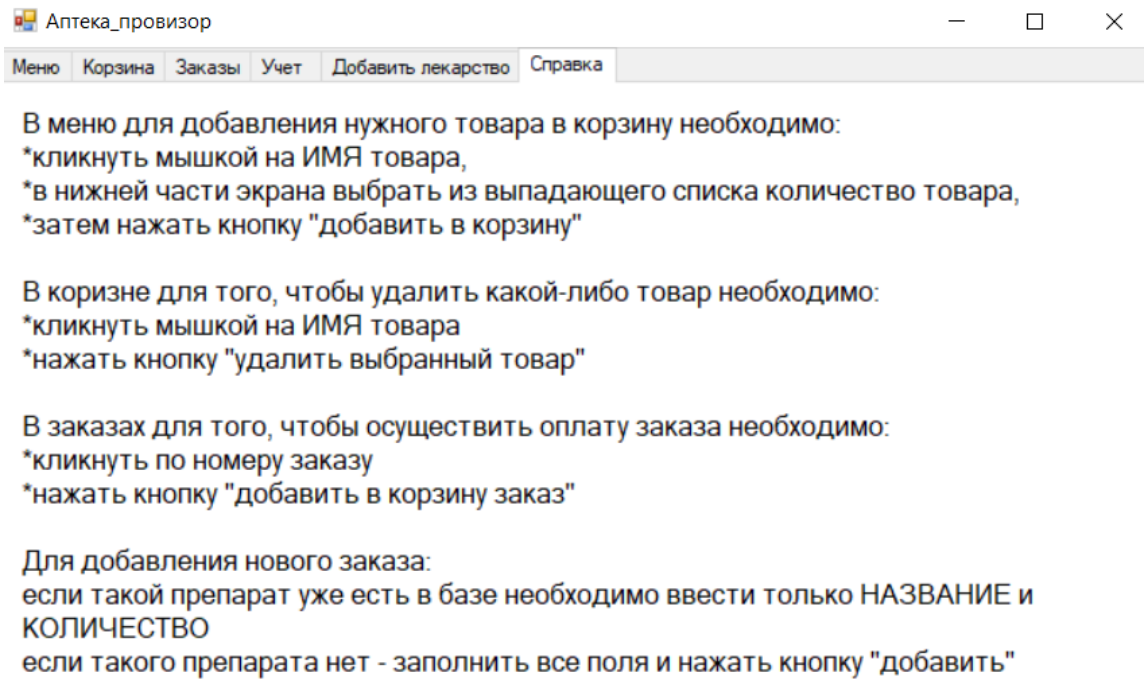

Рисунок 4.19 Вкладка "Справка"

Заключение

В ходе курсового проекта было разработано приложение, позволяющее протестировать взаимодействие объектов классов, спроектированных и реализованных для обеспечения работы аптеки. Также приложение соответствует сформулированному техническому заданию.

Во время тестирования приложения, путем нахождения его плюсов и минусов, была оценена его работоспособность.

Результатом выполнения курсового проекта стало:

- закрепление знаний, полученных в ходе изучения дисциплины «Объектно- ориентированное программирование»;
- приобретение навыков практического программирования с использованием методов объектно-ориентированного программирования;
- подготовка к выполнению выпускной квалификационной работы.

Приложение

Preparation.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class preparation
    {
        int id;
        string name;
        string indication;
        float price;
        int count;

        public preparation(int id, string name, string indication, float price, int
count)
        {
            this.id = id;
            this.name = name;
            this.indication = indication;
            this.price = price;
            this.count = count;
        }

        public int Id { get => id; set => id = value; }
        public string Name { get => name; set => name = value; }
        public string Indication { get => indication; set => indication = value; }
        public float Price { get => price; set => price = value; }
        public int Count { get => count; set => count = value; }

    }
}
```

Order.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class order
    {
        int number_order;
        string phone;
        string preparations;
        float all_price;
        string status;
        public order(int number_order, string phone, string preparations, float
all_price, string status)
        {
            this.number_order = number_order;
            this.phone = phone;
            this.preparations = preparations;
            this.all_price = all_price;
            this.status = status;
        }
```

```
        public int Number_order { get => number_order; set => number_order = value; }
        public float All_Price { get => all_price; set => all_price = value; }
        public string Phone { get => phone; set => phone = value; }
        public string Preparations { get => preparations; set => preparations = value; }
        public string Status { get => status; set => status = value; }
    }
}
```

Analitic.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class Analitic
    {
        int id;
        string name;
        double price;
        int count;
        string date;
        double all_price;
        public Analitic(int id, string name, double price, int count, string date, double
all_price)
        {
            this.id = id;
            this.name = name;
            this.price = price;
            this.count = count;
            this.date = date;
            this.all_price = all_price;
        }
        public int Id { get => id; set => id = value; }
        public string Name { get => name; set => name = value; }
        public double Price { get => price; set => price = value; }
        public int Count { get => count; set => count = value; }
        public string Date { get => date; set => date = value; }
        public double All_price { get => all_price; set => all_price = value; }
    }
}
```

Basket.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class basket
    {
        int id;
        string name;
        float price;
        int count;
        float all_price;

        public basket(int id, string name,float price, int count, float all_price = 0)
        {
```

```
            this.id = id;
            this.name = name;
            this.price = price;
            this.count = count;
            this.all_price = all_price;
        }
        public int Id { get => id; set => id = value; }
        public float All_Price { get => all_price; set => all_price = value; }
        public string Name { get => name; set => name = value; }
        public float Price { get => price; set => price = value; }
        public int Count { get => count; set => count = value; }
    }
}
```

Preparation_list.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data;
using System.Data.SqlClient;

namespace курсовая_ООП
{
    class preparations_list
    {
        public List<preparation> assorty = new List<preparation>();
        public preparations_list()
        {
            List<preparation> assorty = new List<preparation>();
        }

        public void add (preparation preparat)
        {
            assorty.Add(preparat);
        }
        public void add(int id, string name, string indication, float price, int count)
        {
            preparation preparat = new preparation(id, name, indication, price, count);
            assorty.Add(preparat);
        }
        public void clear ()
        {
            assorty.Clear();
        }
        public int get_count_preparation()
        {
            return assorty.Count();
        }
        public int getcount(int id)
        {
            foreach (preparation preparat in assorty)
            {
                if (preparat.Id == id)
                {
                    return preparat.Count;
                }
            }
            return 0;
        }
        public bool getname(string name)
        {
            foreach (preparation preparat in assorty)
```

```
            {
                if (preparat.Name.Contains(name))
                {
                    return true;
                }
            }
            return false;
        }
    }
}
```

Basket_list.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class Basket_list
    {
        public List<basket> bas = new List<basket>();
        public Basket_list()
        {
            List<basket> bas = new List<basket>();
        }

        public void add(basket basket)
        {
            bas.Add(basket);
        }
        public void add(int id, string name, float price, int count, float all_price)
        {
            basket basket = new basket(id, name, price, count, all_price);
            bas.Add(basket);
        }
        public int get_count_basket()
        {
            return bas.Count();
        }
        public void clear()
        {
            bas.Clear();
        }
        public bool get_name_basket(string name)
        {
            foreach (basket basket in bas)
            {
                if (basket.Name.Trim() == name.Trim())
                {
                    return true;
                }
            }
            return false;
        }
    }
}
```

Order_list.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class order_list
    {
        public List<order> list = new List<order>();
        public order_list()
        {
            List<order> list = new List<order>();
        }
        public void add(int number_order, string phone, string preparations, float
all_price, string status)
        {
            order order = new order(number_order,phone,preparations,all_price,status);
            list.Add(order);
        }
        public int get_count_list_in_orders()
        {
            return list.Count();
        }
        public bool get_number(string number)
        {
            foreach (order order in list)
            {
                if (order.Phone == number)
                {
                    return true;
                }
            }
            return false;
        }
    }
}
```

Analitic_list.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace курсовая_ООП
{
    class analitic_list
    {
        public List<Analitic> list = new List<Analitic>();
        public analitic_list()
        {
            List<Analitic> list = new List<Analitic>();
        }
        public void add(int id, string name, double price, int count, string date, double
all_price)
        {
            Analitic analitic = new Analitic(id, name, price, count, date, all_price);
            list.Add(analitic);
        }

        public bool get_name_analitic(string name)
        {
            foreach (Analitic analitic in list)
            {
```

```
                if (analitic.Name.Trim() == name.Trim())
                {
                    return true;
                }
            }
            return false;
        }
        public int get_count_list_in_analitic()
        {
            return list.Count();
        }
        public void clear()
        {
            list.Clear();
        }
        public bool get_now_date_in_list(string name, string date)
        {
            foreach (Analitic analitic in list)
            {
                if (analitic.Name.Trim() == name.Trim() && analitic.Date==date)
                {
                    return true;
                }
            }
            return false;
        }
    }
}
```

Form_menu.cs

```
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 курсовая_ООП
{
    public partial class form_menu : Form
    {
        public form_menu()
        {
            InitializeComponent();
        }

        private void Button1_Click(object sender, EventArgs e)
        {
            //Form1 form1 = new Form1(this);
            Form2 form2 = new Form2(this);
            form2.Show();
            //form1.Show();
            this.Hide();
        }

        private void Button2_Click(object sender, EventArgs e)
        {
            form_client form = new form_client(this);
            form.Show();
            this.Hide();
```

```
        }

        private void Button3_Click(object sender, EventArgs e)
        {
            help_Form2 form2 = new help_Form2(this);
            form2.Show();
            this.Hide();
        }
    }
}
```

Form_administration.cs

```
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.SqlClient;

namespace курсовая_ООП
{
   public partial class Form1 : Form
   {
      public string selectedState;
      public string selectedCount;
      public string count1;
      public string name1;
      public string price1;
      public string id1;
      public string all_price;
      public string status;
      public string order_number="";
      preparations_list preparat = new preparations_list();
      Basket_list bas = new Basket_list();
      analitic_list list = new analitic_list();
      order_list order = new order_list();
      check check = new check();
      public float sum = 0; //общая стоимость для корзины
      public double sum1; //общая стоимость для аналитики
      public int count = 0;
      SqlConnection cnn = new SqlConnection(@"Data Source = (LocalDB)\MSSQLLocalDB;
AttachDbFilename =
C:\Users\Polina\Desktop\курсач_ооп\application_for_pharmacy\курсовая_ООП\preparations.mdf;
Integrated Security = True");
      SqlDataAdapter da = null;
      DataTable table = null;
      form_menu menu;
      public Form1(form_menu m)
      {
```

```
            InitializeComponent();
            menu = m;
            search_criteria_comboBox1.Items.AddRange(new string[] { "Name", "Indication", "Price" });
            search_criteria_comboBox1.SelectedIndexChanged += ComboBox1_SelectedIndexChanged;
        }
        private void ComboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            selectedState = search_criteria_comboBox1.SelectedItem.ToString();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            //cnn = new SqlConnection(@"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename =
C:\Users\Polina\Desktop\курсач ооп\курсовая_ООП\курсовая_ООП\preparations.mdf; Integrated
Security = True");
            cnn.Open();
            //SqlCommand cmd = new SqlCommand();
            DataSet ds = new DataSet();
            //cmd.CommandText = "select * from preparations";
            da = new SqlDataAdapter("select * from preparations", cnn);
            table = new DataTable();
            da.Fill(table);
            da.Fill(ds, "preparations");
            dataGridView1.DataSource = table;
            //cnn.Close();
            DataRow row = ds.Tables[0].NewRow();
            for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
            {
                row = ds.Tables[0].Rows[i];
                preparat.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToString(row[2]),
Convert.ToInt32(row[3]), Convert.ToInt32(row[4]));
                if (preparat.assorty[i].Count == 0)
                    dataGridView1.Rows[i].DefaultCellStyle.ForeColor = Color.Red;

            }
            ds = new DataSet();
            da = new SqlDataAdapter("select Id, Name, Price, Count, FORMAT(Date, 'dd/MM/yyyy', 'en-
US') as Data, All_price FROM Analitic", cnn);
            table = new DataTable();
            da.Fill(table);
            da.Fill(ds, "Analitic");
            row = ds.Tables[0].NewRow();
            for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
            {
                row = ds.Tables[0].Rows[k];
                list.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToDouble(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]), Convert.ToDouble(row[5]));
            }
            sum1 = 0;
            count = 0;
            for (int j = 0; j < list.get_count_list_in_analitic(); j++)
            {
                sum1 += list.list[j].All_price;
                count += list.list[j].Count;
```

```
            textBox4.Text = "Общая сумма покупки: " + sum1 + "р";
            textBox4.Text += "\nКоличество купленных лекарств: " + count;
        }
        dataGridView4.DataSource = table;
        da = new SqlDataAdapter("select * from Basket", cnn);
        ds = new DataSet();
        table = new DataTable();
        da.Fill(ds, "Basket");
        da.Fill(table);
        row = ds.Tables[0].NewRow();
        for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
        {
            row = ds.Tables[0].Rows[k];
            bas.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToInt32(row[2]),
Convert.ToInt32(row[3]), Convert.ToInt32(row[4]));

        }
        dataGridView2.DataSource = table;
        ds = new DataSet();
        da = new SqlDataAdapter("SELECT Order_number, Phone, Preparation, All_price, Status FROM
Orders", cnn);
        table = new DataTable();
        da.Fill(table);
        da.Fill(ds, "Orders");
        row = ds.Tables[0].NewRow();
        for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
        {
            row = ds.Tables[0].Rows[k];
            order.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToString(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]));

        }
        dataGridView3.DataSource = table;
    }

    private void Find_Click(object sender, EventArgs e)
    {
        find.Text = "";
        for (int i = 0; i < dataGridView1.Rows.Count - 1; i++)
        {
            dataGridView1.Rows[i].Visible = true;
        }
    }

    private void Find_button1_Click(object sender, EventArgs e)
    {
        for (int c = 0; c < dataGridView1.Columns.Count; c++)
        {
            if (dataGridView1.Columns[c].HeaderText == selectedState)
            {
                for (int i = 0; i < dataGridView1.Rows.Count - 1; i++)
                {
                    dataGridView1.CurrentCell = null;
```

```
                dataGridView1.Rows[i].Visible = false;

                if (dataGridView1[c, i].Value.ToString().ToLower().Contains(find.Text.ToLower()))
                {
                    dataGridView1.Rows[i].Visible = true;
                }
            }
        }
    }

    private void DataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
        textBox1.Text = "";
        count_comboBox2.Items.Clear();
        count1 = dataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString();
        name1 = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();
        price1 = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();
        id1 = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();

        for (int i = 1; i <= int.Parse(count1); i++)
        {
            count_comboBox2.Items.Add(i.ToString());
            count_comboBox2.SelectedIndexChanged += ComboBox2_SelectedIndexChanged;
        }
    }
    private void ComboBox2_SelectedIndexChanged(object sender, EventArgs e)
    {
        selectedCount = count_comboBox2.SelectedItem.ToString();
    }

    private void DataGridView2_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
        id1= dataGridView2.Rows[e.RowIndex].Cells[0].Value.ToString();
        count1 = dataGridView2.Rows[e.RowIndex].Cells[3].Value.ToString();
        price1= dataGridView2.Rows[e.RowIndex].Cells[2].Value.ToString();
        name1 = dataGridView2.Rows[e.RowIndex].Cells[1].Value.ToString();
        all_price = dataGridView2.Rows[e.RowIndex].Cells[4].Value.ToString();
    }
    //кнопка "удалить выбранный товавр"
    private void Button6_Click(object sender, EventArgs e)
    {
        int id=int.Parse(id1);
        int count = int.Parse(count1);
        float price = float.Parse(price1);
        float all_price1 = float.Parse(all_price);
        for (int i=0;i<preparat.get_count_preparation();i++)
        {
            if(preparat.assorty[i].Id==id)
            {
                preparat.assorty[i].Count += count;
                da = new SqlDataAdapter("UPDATE preparations SET count =" +
(preparat.assorty[i].Count) + " WHERE id = " + id + "; select * from preparations", cnn);
```

```
                table = new DataTable();
                da.Fill(table);
                dataGridView1.DataSource = table;
                //preparat.assorty[j].Count -= c;
                break;
                //sum -= (price * count);
            }
        }
        for (int i = 0; i < bas.get_count_basket(); i++)
        {
            if (bas.bas[i].Id == id)
            {
                bas.bas.RemoveAt(i);
                break;
            }
        }
        sum -= (price * count);
        textBox2.Text = "Общая сумма покупки: " + sum + "р";
        basket basket = new basket(id, name1,price, count, all_price1);
        //bas.remove(basket);
        //bas.bas.Remove(basket);
        da = new SqlDataAdapter("DELETE FROM Basket where Id="+id+";select * from basket", cnn);
        table = new DataTable();
        da.Fill(table);
        dataGridView2.DataSource = table;
    }

    //кнопка "Добавить в корзину"
    private void Button5_Click(object sender, EventArgs e)
    {
        if (count_comboBox2.SelectedItem == null)
            textBox1.Text = "Не выбран ни один товар или его количество!";
        else
        {
            float s = float.Parse(price1);
            int c = int.Parse(selectedCount);
            int iden = int.Parse(id1);
            if (bas.get_count_basket() != 0 && bas.get_name_basket(name1.Trim()) == true)
            {
                for (int i = 0; i < bas.get_count_basket(); i++)
                {
                    /*if (bas.bas[i].Name == name && preparat.getcount(iden) - bas.bas[i].Count == 0)
                        dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;*/
                    if (bas.bas[i].Name.Trim() == name1.Trim() && bas.bas[i].Count + c <=
preparat.getcount(iden))
                    {
                        bas.bas[i].Count += c;
                        bas.bas[i].All_Price += c * s;
                        sum += (s * c);
                        da = new SqlDataAdapter("UPDATE Basket SET count =" + bas.bas[i].Count + "
WHERE id = " + bas.bas[i].Id + "; select * from Basket", cnn);
                        table = new DataTable();
                        da.Fill(table);
```

```
                    dataGridView2.DataSource = table;
                    //dataGridView1.DataSource = table;
                    textBox1.Text = "Добавлено";
                }
                else if (bas.bas[i].Name.Trim() == name1.Trim() && bas.bas[i].Count + c >
preparat.getcount(iden))
                {
                    textBox1.Text = "Данный товар закончился или максимальное количество уже
лежит в корзине!";
                    //dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;
                }
            }
        }
        else
        {
            bas.add(iden, name1, s, c, (c * s));
            //bask.init();
            //data.add_basket(name, s, c);
            sum += (s * c);
            da = new SqlDataAdapter("INSERT INTO Basket (id, Name,Price,Count,All_Price)
VALUES (" + iden + ", N'" + name1 + "'," + s + "," + c + "," + (c * s) + "); select * from Basket", cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView2.DataSource = table;
            textBox1.Text = "Добавлено";
            /*for (int i = 0; i < bas.get_count_basket(); i++)
            {
                if (preparat.getcount(iden) - bas.bas[i].Count == 0)
                    dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;
            }*/
        }
        for (int j = 0; j < preparat.get_count_preparation(); j++)
        {
            if (preparat.assorty[j].Id == iden)
            {
                da = new SqlDataAdapter("UPDATE preparations SET count =" +
(preparat.assorty[j].Count - c) + " WHERE id = " + iden + "; select * from preparations", cnn);
                table = new DataTable();
                da.Fill(table);
                dataGridView1.DataSource = table;
                preparat.assorty[j].Count -= c;
                if (preparat.assorty[j].Count == 0)
                    dataGridView1.Rows[j].DefaultCellStyle.ForeColor = Color.Red;
                break;
            }
        }
    }
    textBox2.Text = "Общая сумма покупки: " + sum + "р";
}
private string now_date()
{
    string date = "";
    int day = DateTime.Now.Day;
```

```
            int month = DateTime.Now.Month;
            int year = DateTime.Now.Year;
            if (day < 10)
            {
               date += "0" + day;
            }
            else
            {
               date += day;
            }
            date += "/";
            if (month < 10)
            {
               date += "0" + month;
            }
            else
            {
               date += month;
            }
            date += "/";
            date += year;
            return date;
        }

        //кнопка "купить"
        private void Button4_Click(object sender, EventArgs e)
        {
            //analitic_list list = new analitic_list();
            //int p = dataGridView2.Columns.Count;
            //dataGridView1.Rows.Clear();
            if (bas.get_count_basket()==0)
            {
               MessageBox.Show("Не выбран ни один товар");
               return;
            }
            sum = 0;
            sum1 = 0;
            count = 0;
            list.clear();
            DataSet ds = new DataSet();
            da = new SqlDataAdapter("select Id, Name, Price, Count, FORMAT(Date, 'dd/MM/yyyy', 'en-
US') as Data, All_price FROM Analitic", cnn);
            da.Fill(ds, "Analitic");
            DataRow row = ds.Tables[0].NewRow();
            for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
            {
               row = ds.Tables[0].Rows[k];
               list.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToDouble(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]), Convert.ToDouble(row[5]));

            }
            for (int j = 0; j < list.get_count_list_in_analitic(); j++)
            {
```

```
            sum1 += list.list[j].All_price;
            textBox4.Text = "Общая сумма покупки: " + sum1 + "р";
            textBox4.Text += "\nКоличество купленных лекарств: " + count;
        }
        textBox2.Text = "Общая сумма покупки: " + sum + "р";
        for (int i = 0; i < dataGridView2.Rows.Count - 1; i++)
        {
            if (list.get_count_list_in_analitic() != 0 && list.get_name_analitic(bas.bas[i].Name) == true)
            {
                for (int q = 0; q < list.get_count_list_in_analitic(); q++)
                {
                    if (list.list[q].Name.Trim() == bas.bas[i].Name.Trim() && list.list[q].Date == now_date())
                    {
                        SqlDataAdapter d;
                        list.list[q].Count += bas.bas[i].Count;
                        list.list[q].All_price += bas.bas[i].Count * bas.bas[i].Price;
                        d = new SqlDataAdapter("UPDATE Analitic SET Count =" + list.list[q].Count +
",All_Price=" + list.list[q].All_price + "  WHERE Id = " + list.list[q].Id + "; select Id, Name, Price, Count,
FORMAT(Date, 'dd/MM/yyyy', 'en-US') as Data, All_price from Analitic;", cnn);
                        table = new DataTable();
                        d.Fill(table);
                        dataGridView4.DataSource = table;
                        textBox4.Text = "Общая сумма покупки: " + (sum1 + bas.bas[i].Count *
bas.bas[i].Price) + "р";
                        textBox4.Text += "\nКоличество купленных лекарств: " + (count+bas.bas[i].Count);
                        break;
                    }
                    else if (list.list[q].Name.Trim() == bas.bas[i].Name.Trim() &&
!list.get_now_date_in_list(bas.bas[i].Name, now_date()))
                    {
                        //DataSet ds = new DataSet();
                        SqlDataAdapter daa;
                        daa = new SqlDataAdapter("INSERT INTO Analitic (Name,Price,Count,All_Price)
VALUES (N'" + bas.bas[i].Name + "'," + bas.bas[i].Price + "," + bas.bas[i].Count + "," +
bas.bas[i].All_Price + "); select Id, Name, Price, Count, FORMAT(Date, 'dd/MM/yyyy', 'en-US') as Data
, All_price from Analitic;", cnn); //Name=" + bas.bas[i].Name+", Price="+bas.bas[i].Price+",
Count="+bas.bas[i].Count+ ", Date=CURDATE(), All_Price="+bas.bas[i].All_Price, cnn);
                        table = new DataTable();
                        daa.Fill(table);
                        dataGridView4.DataSource = table;
                        textBox4.Text = "Общая сумма покупки: " + (sum1 + bas.bas[i].All_Price) + "р";
                        textBox4.Text += "\nКоличество купленных лекарств: " + (count+bas.bas[i].Count);
                        break;
                        /*da.Fill(ds, "Analitic");
                        DataRow row = ds.Tables[0].NewRow();
                        for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
                        {
                            row = ds.Tables[0].Rows[k];
                            list.add(Convert.ToString(row[1]), Convert.ToDouble(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]), Convert.ToDouble(row[5]));

                        }*/
                    }
```

```
                }
            }
            else
            {
                //DataSet ds = new DataSet();
                SqlDataAdapter daa;
                daa = new SqlDataAdapter("INSERT INTO Analitic (Name,Price,Count,All_Price)
VALUES (N'" + bas.bas[i].Name + "'," + bas.bas[i].Price + "," + bas.bas[i].Count + "," +
bas.bas[i].All_Price + "); select Id, Name, Price, Count, FORMAT(Date, 'dd/MM/yyyy', 'en-US') as Data,
All_price from Analitic;", cnn); //Name=" + bas.bas[i].Name+", Price="+bas.bas[i].Price+",
Count="+bas.bas[i].Count+ ", Date=CURDATE(), All_Price="+bas.bas[i].All_Price, cnn);
                table = new DataTable();
                daa.Fill(table);
                dataGridView4.DataSource = table;
                textBox4.Text = "Общая сумма покупки: " +( sum1+ bas.bas[i].All_Price )+ "p";
                textBox4.Text += "\nКоличество купленных лекарств: " + (count+bas.bas[i].Count);
                /*da.Fill(ds, "Analitic");
                DataRow row = ds.Tables[0].NewRow();
                for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
                {
                    row = ds.Tables[0].Rows[k];
                    list.add(Convert.ToString(row[1]), Convert.ToDouble(row[2]), Convert.ToInt32(row[3]),
Convert.ToString(row[4]), Convert.ToDouble(row[5]));

                }*/
                //break;
            }
        }
        /*double sum1 = 0;
        for (int j = 0; j < list.get_count_list_in_analitic(); j++)
        {
            sum1 += list.list[j].All_price;
            textBox1.Text = "Общая сумма покупки: " + sum1 + "p";
        }*/
        da = new SqlDataAdapter("DELETE FROM Basket", cnn);
        table = new DataTable();
        da.Fill(table);
        dataGridView2.DataSource = table;
        //dataGridView2.Rows.Clear();
        bas.clear();
        MessageBox.Show("Покупка совершена успешно!");
    }

    //добавление лекарства
    private void Button1_Click(object sender, EventArgs e)
    {
        //check check = new check();
        if (check.checkdigit(textBox11.Text) && textBox11.ReadOnly == false &&
check.StringCheck(textBox5.Text))
        {
            if (preparat.getname(textBox5.Text)) //если в таблице есть такой препарат
            {
                textBox6.Text = "Такой препарат есть в базе! Введите только количество";
```

```
                for (int i = 0; i < preparat.get_count_preparation(); i++)
                {
                    if (preparat.assorty[i].Name.Contains(textBox5.Text))
                    {
                        da = new SqlDataAdapter("UPDATE preparations SET count =" +
(preparat.assorty[i].Count + int.Parse(textBox11.Text)) + " WHERE id = " + preparat.assorty[i].Id + ";
select * from preparations", cnn);
                        table = new DataTable();
                        da.Fill(table);
                        dataGridView1.DataSource = table;
                        preparat.assorty[i].Count += int.Parse(textBox11.Text);
                        textBox13.Text = "Добавлено!";
                    }
                }
            }
            else if (check.checkdigit(textBox9.Text) && check.StringCheck(textBox7.Text))
            {
                da = new SqlDataAdapter("insert into preparations (Name,Indication,Price, Count) Values
(N'" + textBox5.Text + "', N'" + textBox7.Text + "'," + textBox9.Text + "," + textBox11.Text + "); select
* from preparations", cnn);
                table = new DataTable();
                da.Fill(table);
                dataGridView1.DataSource = table;
                preparat.add(preparat.get_count_preparation(), textBox5.Text, textBox7.Text,
float.Parse(textBox9.Text), int.Parse(textBox11.Text));
                textBox13.Text = "Добавлено!";
                //break;
            }
            else
            {
                textBox13.Text = "Error!"; //textBox12.Text = "Error!";
            }
        }
        else
        {
            textBox13.Text = "Error!"; //textBox12.Text = "Error!";
        }

    }
    private void TextBox5_Click(object sender, EventArgs e)
    {
        textBox5.Clear(); textBox7.Clear(); textBox9.Clear(); textBox11.Clear();
        textBox6.Clear(); textBox8.Clear(); textBox10.Clear();textBox12.Clear();
        textBox13.Clear();
        textBox7.ReadOnly = false; textBox9.ReadOnly = false; textBox11.ReadOnly = false;
    }

    private void TextBox7_Click(object sender, EventArgs e)
    {
        //textBox7.ReadOnly = false; textBox9.ReadOnly = false;
        check check = new check();
        if (check.StringCheck(textBox5.Text))
        {
```

```
        if (preparat.getname(textBox5.Text)) //если в таблице есть такой препарат
        {
            textBox7.ReadOnly = true; textBox9.ReadOnly = true;
            textBox6.Text = "Такой препарат есть в базе! Введите только количество";
        }
    }
    else
    {
        textBox6.Text = "Error!";
        textBox11.ReadOnly = true; textBox9.ReadOnly = true; textBox7.ReadOnly = true;
    }
}
private void TextBox9_Click(object sender, EventArgs e)
{
    //textBox11.ReadOnly = false;
    check check = new check();
    if (!check.StringCheck(textBox7.Text) && textBox7.ReadOnly == false)
    {
        textBox11.ReadOnly = true;
        textBox8.Text = "Error!";
    }
}

private void TextBox11_TextChanged(object sender, EventArgs e)
{
    check check = new check();
    if (!check.checkdigit(textBox9.Text) && textBox9.ReadOnly==false)
    {
        textBox10.Text = "Error!";
    }
}
//кнопка выход в меню авторизации
private void Button2_Click(object sender, EventArgs e)
{
    da = new SqlDataAdapter("DELETE FROM Basket", cnn);
    table = new DataTable();
    da.Fill(table);
    this.Close();
    menu.Show();
}

private void DataGridView3_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    //id1 = dataGridView2.Rows[e.RowIndex].Cells[0].Value.ToString();
    //count1 = dataGridView3.Rows[e.RowIndex].Cells[3].Value.ToString();
    //price1 = dataGridView3.Rows[e.RowIndex].Cells[2].Value.ToString();
    name1 = dataGridView3.Rows[e.RowIndex].Cells[2].Value.ToString();
    all_price = dataGridView3.Rows[e.RowIndex].Cells[3].Value.ToString();
    status = dataGridView3.Rows[e.RowIndex].Cells[4].Value.ToString();
    order_number = dataGridView3.Rows[e.RowIndex].Cells[0].Value.ToString();
}

private int count_preparat_in_order(string name)
```

```
        {
            int count = 0;
            for(int i=0;i<name.Length;i++)
            {
                if (name[i] == ',')
                    count++;
            }
            return count;
        }

        //кнопка добавить ЗАКАЗ в корзину
        private void Button3_Click(object sender, EventArgs e)
        {
            int flag = 0;
            if (order.get_count_list_in_orders() == 0)
            { MessageBox.Show("Заказов нет!"); return; }
            else if (bas.get_count_basket() != 0)
            { MessageBox.Show("Корзина не пуста, очистите ее, прежде чем добавлять туда заказ!");
return; }
            else if (order_number == "")
            {
                MessageBox.Show("Заказ не выбран!"); return;
            }
            for (int i = 0; i < order.get_count_list_in_orders(); i++)
            {
                if (order.list[i].Number_order == int.Parse(order_number) && order.list[i].Status.Trim() ==
"Завершен")
                { flag = 1; break; }
            }
            //flag = 0;
            if (flag==1)
            {
                MessageBox.Show("Этот заказ уже оплачен!");
                return;
            }
            else
            {
                int count_preparat = count_preparat_in_order(name1);
                string str = "";
                for (int i = 0; i < count_preparat; i++)
                {
                    int x = 0;
                    while (name1[x] != ',')
                    {
                        //x++;
                        str += name1[x];
                        x++;
                    }
                    int count = str[str.Length - 1] - '0';
                    string name = str.Substring(0, str.Length - 2);
                    //name = name.Trim();
                    /*for(int j=0;name.Length<30;j++)
                    {
```

```
                name = name.Insert(name.Length, " ");
            }
            /*da = new SqlDataAdapter("(select Price from preparations where Name='" + name + "')",
cnn);
            DataSet ds = new DataSet();
            //DataRow row = new DataRow;
            da.Fill(ds, "preparations");
            DataRow row = ds.Tables[0].NewRow();
            for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
            {
                row = ds.Tables[0].Rows[k];
                string stroka = Convert.ToString( row[0]);
                //order.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]),
Convert.ToString(row[2]), Convert.ToInt32(row[3]), Convert.ToString(row[4]));

            }*/
            int id = 1;
            float price = 0;
            for (int j = 0; j < preparat.get_count_preparation(); j++)
            {
                if (preparat.assorty[j].Name.Contains(name))
                {
                    id = preparat.assorty[j].Id;
                    price = preparat.assorty[j].Price;
                    break;
                }

            }
            //da.Fill(table);
            da = new SqlDataAdapter(//"select Id, Price from preparations where Name="+name+ ";" +
                "INSERT INTO Basket (Id, Name,Price,Count,All_Price)  " +
                "VALUES (" + id + ",N'" + name + "'," + price + "," + count + "," + (price * count) + ");
select * from Basket", cnn);

            bas.add(id, name, price, count, (count * price));
            //bask.init();
            //data.add_basket(name, s, c);
            //sum += (s * c);
            //da = new SqlDataAdapter("INSERT INTO Basket (id, Name,Price,Count,All_Price)
VALUES (" + iden + ", N'" + name1 + "'," + s + "," + c + "," + (c * s) + "); select * from Basket", cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView2.DataSource = table;
            //textBox1.Text = "Добавлено";
            /*for (int i = 0; i < bas.get_count_basket(); i++)
            {
                if (preparat.getcount(iden) - bas.bas[i].Count == 0)
                    dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;
            }*/
            name1 = name1.Substring(str.Length + 1);
            str = "";
        }
        textBox2.Text = "Общая сумма покупки: " + all_price + "р";
```

```
            da = new SqlDataAdapter("update Orders set Status=N'Завершен' where Order_number=" +
order_number + "; select Order_number, Phone, Preparation, All_price, Status FROM Orders", cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView3.DataSource = table;
            order_number = "";
        }
    }
  }
}
```

Form_secure.cs

```
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 курсовая_ООП
{
    public partial class Form2 : Form
    {
        form_menu menu;
        public Form2(form_menu m)
        {
            menu = m;
            InitializeComponent();
        }

        private void Button1_Click(object sender, EventArgs e)
        {
            if (textBox1.Text != "1234")
                MessageBox.Show("Неверный пароль!");
            else
            {
                Form1 form = new Form1(menu);
                this.Close();
                form.Show();
            }
        }
    }
}
```

Form_client.cs

```
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.SqlClient;

namespace курсовая_ООП
{
    public partial class form_client : Form
    {
        public string selectedState;
        public string selectedCount;
        form_menu menu;
        public string count1;
        public string name1;
        public string price1;
        public string id1;
        public string all_price;
        preparations_list preparat = new preparations_list();
        Basket_list bas = new Basket_list();
        //analitic_list list = new analitic_list();
        order_list order = new order_list();
        public float summa = 0; //общая стоимость для корзины
        public double sum1; //общая стоимость для аналитики
        SqlConnection cnn = new SqlConnection(@"Data Source = (LocalDB)\MSSQLLocalDB;
AttachDbFilename =
C:\Users\Polina\Desktop\курсач_ооп\application_for_pharmacy\курсовая_ООП\preparations.mdf;
Integrated Security = True");
        SqlDataAdapter da = null;
        DataTable table = null;
        check check = new check();
        public form_client(form_menu m)
        {
            menu = m;
            InitializeComponent();
            search_criteria_comboBox1.Items.AddRange(new string[] { "По имени", "По назначению", "По
цене" });
            search_criteria_comboBox1.SelectedIndexChanged += ComboBox1_SelectedIndexChanged;
        }
        private void ComboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            selectedState = search_criteria_comboBox1.SelectedItem.ToString();
        }

        private void Form_client_Load(object sender, EventArgs e)
        {
            DataSet ds = new DataSet();
            //cmd.CommandText = "select * from preparations";
            da = new SqlDataAdapter("select * from preparations", cnn);
            table = new DataTable();
            da.Fill(table);
            da.Fill(ds, "preparations");
            dataGridView1.DataSource = table;
            //cnn.Close();
            DataRow row = ds.Tables[0].NewRow();
            for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
            {
```

```
            row = ds.Tables[0].Rows[i];
            preparat.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToString(row[2]),
Convert.ToInt32(row[3]), Convert.ToInt32(row[4]));
            if (preparat.assorty[i].Count==0)
               dataGridView1.Rows[i].DefaultCellStyle.ForeColor = Color.Red;

         }
         ds = new DataSet();
        /* da = new SqlDataAdapter("select Id, Name, Price, Count, FORMAT(Date, 'dd/MM/yyyy', 'en-
US') , All_price FROM Analitic", cnn);
         table = new DataTable();
         da.Fill(table);
         da.Fill(ds, "Analitic");
         row = ds.Tables[0].NewRow();
         for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
         {
            row = ds.Tables[0].Rows[k];
            list.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToDouble(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]), Convert.ToDouble(row[5]));
         }
         sum1 = 0;
         for (int j = 0; j < list.get_count_list_in_analitic(); j++)
         {
            sum1 += list.list[j].All_price;
            textBox4.Text = "Общая сумма покупки: " + sum1 + "р";
         }
         dataGridView4.DataSource = table;*/
         da = new SqlDataAdapter("select * from Basket", cnn);
         table = new DataTable();
         da.Fill(table);
         dataGridView2.DataSource = table;
         ds = new DataSet();
         da = new SqlDataAdapter("SELECT Order_number, Phone, Preparation, All_price, Status FROM
Orders", cnn);
         da.Fill(ds, "Orders");
         row = ds.Tables[0].NewRow();
         for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
         {
            row = ds.Tables[0].Rows[k];
            order.add(Convert.ToInt32(row[0]), Convert.ToString(row[1]), Convert.ToString(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]));

         }
      }

      private void Find_Click(object sender, EventArgs e)
      {
         find.Text = "";
         for (int i = 0; i < dataGridView1.Rows.Count - 1; i++)
         {
            dataGridView1.Rows[i].Visible = true;
         }
      }
```

```
private void DataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
   textBox1.Text = "";
   count_comboBox2.Items.Clear();
   count1 = dataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString();
   name1 = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();
   price1 = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();
   id1 = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();

   for (int i = 1; i <= int.Parse(count1); i++)
   {
      count_comboBox2.Items.Add(i.ToString());
      count_comboBox2.SelectedIndexChanged += ComboBox2_SelectedIndexChanged;
   }
}

private void ComboBox2_SelectedIndexChanged(object sender, EventArgs e)
{
   selectedCount = count_comboBox2.SelectedItem.ToString();
}

private void Find_button1_Click(object sender, EventArgs e)
{
   for (int c = 0; c < dataGridView1.Columns.Count; c++)
   {
      if (dataGridView1.Columns[c].HeaderText == selectedState)
      {
         for (int i = 0; i < dataGridView1.Rows.Count - 1; i++)
         {
            dataGridView1.CurrentCell = null;
            dataGridView1.Rows[i].Visible = false;

            if (dataGridView1[c, i].Value.ToString().ToLower().Contains(find.Text.ToLower()))
            {
               dataGridView1.Rows[i].Visible = true;
            }
         }
      }
   }
}

private void DataGridView2_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
   id1 = dataGridView2.Rows[e.RowIndex].Cells[0].Value.ToString();
   count1 = dataGridView2.Rows[e.RowIndex].Cells[3].Value.ToString();
   price1 = dataGridView2.Rows[e.RowIndex].Cells[2].Value.ToString();
   name1 = dataGridView2.Rows[e.RowIndex].Cells[1].Value.ToString();
   all_price = dataGridView2.Rows[e.RowIndex].Cells[4].Value.ToString();
}

private void Button6_Click(object sender, EventArgs e)
{
```

```
            int id = int.Parse(id1);
            int count = int.Parse(count1);
            float price = float.Parse(price1);
            float all_price1 = float.Parse(all_price);
            for (int i = 0; i < preparat.get_count_preparation(); i++)
            {
                if (preparat.assorty[i].Id == id)
                {
                    preparat.assorty[i].Count += count;
                    da = new SqlDataAdapter("UPDATE preparations SET count =" +
(preparat.assorty[i].Count) + " WHERE id = " + id + "; select * from preparations", cnn);
                    table = new DataTable();
                    da.Fill(table);
                    dataGridView1.DataSource = table;
                    //preparat.assorty[j].Count -= c;
                    break;
                    //sum -= (price * count);
                }
            }
            for (int i = 0; i < bas.get_count_basket(); i++)
            {
                if (bas.bas[i].Id == id)
                {
                    bas.bas.RemoveAt(i);
                    break;
                }
            }
            summa -= (price * count);
            textBox2.Text = "Общая сумма покупки: " + summa + "р";
            basket basket = new basket(id, name1, price, count, all_price1);
            //bas.remove(basket);
            //bas.bas.Remove(basket);
            da = new SqlDataAdapter("DELETE FROM Basket where Id=" + id + ";select * from basket",
cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView2.DataSource = table;
        }

        private void Button5_Click(object sender, EventArgs e)
        {
            if (count_comboBox2.SelectedItem == null)
                textBox1.Text = "Не выбран ни один товар или его количество!";
            else
            {
                float s = float.Parse(price1);
                int c = int.Parse(selectedCount);
                int iden = int.Parse(id1);
                if (bas.get_count_basket() != 0 && bas.get_name_basket(name1.Trim()) == true)
                {
                    for (int i = 0; i < bas.get_count_basket(); i++)
                    {
                        /*if (bas.bas[i].Name == name && preparat.getcount(iden) - bas.bas[i].Count == 0)
```

```
                    dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;*/
                if (bas.bas[i].Name == name1 && bas.bas[i].Count + c <= preparat.getcount(iden))
                {
                    bas.bas[i].Count += c;
                    bas.bas[i].All_Price += c * s;
                    summa += (s * c);
                    da = new SqlDataAdapter("UPDATE Basket SET count =" + bas.bas[i].Count + "
WHERE id = " + bas.bas[i].Id + "; select * from Basket", cnn);
                    table = new DataTable();
                    da.Fill(table);
                    dataGridView2.DataSource = table;
                    //dataGridView1.DataSource = table;
                    textBox1.Text = "Добавлено";
                }
                else if (bas.bas[i].Name.Trim() == name1.Trim() && bas.bas[i].Count + c >
preparat.getcount(iden))
                {
                    textBox1.Text = "Данный товар закончился или максимальное количество уже
лежит в корзине!";
                    //dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;
                }
            }
        }
        else
        {
            bas.add(iden, name1, s, c, (c * s));
            //bask.init();
            //data.add_basket(name, s, c);
            summa += (s * c);
            da = new SqlDataAdapter("INSERT INTO Basket (id, Name,Price,Count,All_Price)
VALUES (" + iden + ", N'" + name1 + "'," + s + "," + c + "," + (c * s) + "); select * from Basket", cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView2.DataSource = table;
            textBox1.Text = "Добавлено";
            /*for (int i = 0; i < bas.get_count_basket(); i++)
            {
                if (preparat.getcount(iden) - bas.bas[i].Count == 0)
                    dataGridView1.Rows[iden - 1].DefaultCellStyle.ForeColor = Color.Red;
            }*/
        }
        for (int j = 0; j < preparat.get_count_preparation(); j++)
        {
            if (preparat.assorty[j].Id == iden)
            {
                da = new SqlDataAdapter("UPDATE preparations SET count =" +
(preparat.assorty[j].Count - c) + " WHERE id = " + iden + "; select * from preparations", cnn);
                table = new DataTable();
                da.Fill(table);
                dataGridView1.DataSource = table;
                preparat.assorty[j].Count -= c;
                if (preparat.assorty[j].Count == 0)
                    dataGridView1.Rows[j].DefaultCellStyle.ForeColor = Color.Red;
```

```
                    break;
                }
            }
        }
        textBox2.Text = "Общая сумма покупки: " + summa + "р";
    }

    private void Button4_Click(object sender, EventArgs e)
    {
        //sum = 0;
        //sum1 = 0;
        //order.clear();
        //DataSet ds = new DataSet();
        //da = new SqlDataAdapter("SELECT Order_number, Phone, Preparation, All_price, Status
FROM Orders", cnn);
        //da.Fill(ds, "Orders");
        //DataRow row = ds.Tables[0].NewRow();
        //for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
        //{
        //    row = ds.Tables[0].Rows[k];
        //    order.add(Convert.ToInt32(row[0]), Convert.ToInt32(row[1]), Convert.ToString(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]));

        //}
        /*for (int j = 0; j < order.get_count_list_in_orders(); j++)
        {
            sum1 += list.list[j].All_price;
            textBox4.Text = "Общая сумма покупки: " + sum1 + "р";
        }
        textBox2.Text = "Общая сумма покупки: " + sum + "р";*/
        //for (int i = 0; i < dataGridView2.Rows.Count - 1; i++)
        //{
        if (textBox3.Text != "" && check.checkNumber(textBox3.Text))
        {
            if (order.get_count_list_in_orders() != 0 && order.get_number(textBox3.Text) == true)
            {
                for (int q = 0; q < order.get_count_list_in_orders(); q++)
                {
                    if (order.list[q].Phone == textBox3.Text && order.list[q].Status != "Завершен")
                    {
                        /*SqlDataAdapter d;
                        list.list[q].Count += bas.bas[i].Count;
                        list.list[q].All_price += bas.bas[i].Count * bas.bas[i].Price;
                        d = new SqlDataAdapter("UPDATE Analitic SET Count =" + list.list[q].Count +
",All_Price=" + list.list[q].All_price + "  WHERE Id = " + list.list[q].Id + "; select Id, Name, Price, Count,
FORMAT(Date, 'dd/MM/yyyy', 'en-US') , All_price from Analitic;", cnn);
                        table = new DataTable();
                        d.Fill(table);
                        dataGridView4.DataSource = table;
                        textBox4.Text = "Общая сумма покупки: " + (sum1 + bas.bas[i].Count *
bas.bas[i].Price) + "р";*/
                        MessageBox.Show("По данному номеру телефона уже есть не завершенный
заказ!");
```

```
                    break;
                }
                else if (order.list[q].Phone == textBox3.Text && order.list[q].Status == "Завершен")
                {
                    //DataSet ds = new DataSet();
                    //Создание объекта для генерации чисел
                    Random rnd = new Random();
                    //Получить очередное (в данном случае - первое) случайное число
                    int value = rnd.Next(10000, 32000);
                    string str = "";
                    for (int i = 0; i < dataGridView2.Rows.Count - 1; i++)
                    {
                        str += bas.bas[i].Name + "-" + bas.bas[i].Count + ",";

                    }
                    SqlDataAdapter daa;
                    daa = new SqlDataAdapter("INSERT INTO Orders (Order_number, Phone, Preparation,
All_price, Status)  VALUES (" + value + "," + textBox3.Text + ",N'" + str + "'," + summa +
",N'Готов_к_выдаче'); select Order_number, Phone, Preparation, All_price, Status FROM Orders;",
cnn); //Name=" + bas.bas[i].Name+", Price="+bas.bas[i].Price+", Count="+bas.bas[i].Count+ ",
Date=CURDATE(), All_Price="+bas.bas[i].All_Price, cnn);
                    table = new DataTable();
                    daa.Fill(table);
                    //dataGridView2.DataSource = table;
                    order.add(value, textBox3.Text, str, summa, "Готов_к_выдаче");
                    MessageBox.Show("Заказ вас ожидает в аптеке");
                    //break
                    //textBox4.Text = "Общая сумма покупки: " + (sum1 + bas.bas[i].All_Price) + "p";
                    /*da.Fill(ds, "Analitic");
                    DataRow row = ds.Tables[0].NewRow();
                    for (int k = 0; k < ds.Tables[0].Rows.Count; k++)
                    {
                        row = ds.Tables[0].Rows[k];
                        list.add(Convert.ToString(row[1]), Convert.ToDouble(row[2]),
Convert.ToInt32(row[3]), Convert.ToString(row[4]), Convert.ToDouble(row[5]));

                    }*/
                }
            }
        }
        else
        {
            Random rnd = new Random();
            //Получить очередное (в данном случае - первое) случайное число
            int value = rnd.Next(10000, 32000);
            string str = "";
            for (int i = 0; i < dataGridView2.Rows.Count - 1; i++)
            {
                //bas.bas[i].Name=bas.bas[i].Name.Trim();
                str += bas.bas[i].Name.Trim() + "-" + bas.bas[i].Count + ",";

            }
            SqlDataAdapter daa;
```

```
            daa = new SqlDataAdapter("INSERT INTO Orders (Order_number, Phone, Preparation,
All_price, Status)  VALUES (" + value + "," + textBox3.Text + ",N'" + str + "'," + summa +
",N'Готов_к_выдаче'); select Order_number, Phone, Preparation, All_price, Status FROM Orders;",
cnn); //Name=" + bas.bas[i].Name+", Price="+bas.bas[i].Price+", Count="+bas.bas[i].Count+ ",
Date=CURDATE(), All_Price="+bas.bas[i].All_Price, cnn);
                table = new DataTable();
                daa.Fill(table);
                order.add(value, textBox3.Text, str, summa, "Готов к выдаче");
                //dataGridView2.DataSource = table;
                MessageBox.Show("Заказ вас ожидает в аптеке");
                //break;
            }
            //}
            /*double sum1 = 0;
            for (int j = 0; j < list.get_count_list_in_analitic(); j++)
            {
                sum1 += list.list[j].All_price;
                textBox1.Text = "Общая сумма покупки: " + sum1 + "р";
            }*/
            da = new SqlDataAdapter("DELETE FROM Basket", cnn);
            table = new DataTable();
            da.Fill(table);
            dataGridView2.DataSource = table;
            //dataGridView2.Rows.Clear();
            bas.clear();
            //MessageBox.Show("Заказ вас ожидает в аптеке");
        }
        else
            textBox4.Text = "Error!";
    }

    private void TextBox3_Click(object sender, EventArgs e)
    {
        textBox4.Clear();
        textBox3.Clear();
    }

    private void Button2_Click(object sender, EventArgs e)
    {
        da = new SqlDataAdapter("DELETE FROM Basket", cnn);
        table = new DataTable();
        da.Fill(table);
        this.Close();
        menu.Show();
    }
  }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;

namespace курсовая_ООП
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new form_menu());
        }
    }
}
```