**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

Кафедра информационных систем

КУРСОВАЯ РАБОТА

по дисциплине «Информатика и программирование»

«Разработка приложения по указанию транспортных услуг»

Исполнитель:

студентка Хлевная Е. А.

группа ПИ 185-1

Руководитель:

старший преподаватель,

Павлова Е. А.

Тюмень – 2019

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc11702194)

[1. ПОСТАНОВКА ЗАДАЧ 4](#_Toc11702195)

[2. ОПИСАНИЕ АЛГОРИТМОВ И КЛАССОВ, МЕТОДОВ 6](#_Toc11702196)

[2.1. Чтение и запись данных 6](#_Toc11702197)

[2.2. Добавление новых данных 7](#_Toc11702198)

[2.3. Изменение данных 8](#_Toc11702199)

[2.4. Удаление данных 8](#_Toc11702200)

[2.5. Сохранение данных 9](#_Toc11702201)

[3. ОПИСАНИЕ АЛГОРИТМОВ ВЫЧИСЛЕНИЯ 10](#_Toc11702202)

[3.1. Вычисление самого частого клиента 10](#_Toc11702203)

[3.2. Подсчёт показателей: самая прибыльная аренда, сумма всех аренд, средняя цена аренды. 10](#_Toc11702204)

[3.3. Подсчёт показателя: средне количество суток аренды. 11](#_Toc11702205)

[3.4. Максимальное время аренды 12](#_Toc11702206)

[3.5. Самый прибыльный клиент 12](#_Toc11702207)

[4. РУКОВОДСТВО ПРИЛОЖЕНИЯ 14](#_Toc11702208)

[ЗАКЛЮЧЕНИЕ 21](#_Toc11702209)

# ВВЕДЕНИЕ

Реализация приложения для автоматизации учёта автомобилей будет организована в базе данных.

База данных — совокупность связанных данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования.

Один из главных принципов организации данных – построение взаимосвязей между всеми элементами, что и отличает базу данных от простого набора таблиц. Т.е. информация в таблицах реляционной базы данных должна быть соответствующим образом организована. Еще один важнейший принцип — нормализация таблиц, которая сводится к устранению недостатков структуры базы данных, приводящих к различным недостаткам и нарушениям целостности данных. Недостатками структуры можно назвать, например, противоречивость данных. Проще говоря, нормализация – разбиение таблицы на две или более для исключения повторения (избыточности) информации.

В данной работе мы рассмотрим реализацию базы данных на языке программирования C# и в системе управления базами данных (СУБД) Microsoft Access.

# ПОСТАНОВКА ЗАДАЧ

*Главная задача данной разработки* – создание приложения БД на языке C#, позволяющее вести учет аренды автомобилей, а именно добавление, удаление, изменение: данных о клиентах, данных об автомобилях, данных об аренде клиентами автомобилей, получать статистику из огромного количества информации для упрощения работы с СУБД.

*Предметная область*: аренда автомобилей.

*Данные:* сведения о клиентах (фамилия клиента, имя клиента), сведения об автомобилях (марка автомобиля, посуточная цена аренды), сведения об аренде (фамилия клиента, имя клиента, марка автомобиля, количество суток аренды).

*Вычисления:* самый частый клиент, самый прибыльный клиент, самая прибыльная аренда, сумма всех аренд, средняя цена аренды, среднее количество суток аренды, максимальное время аренды.

*Коррекция:* добавление, удаление, сохранение и изменение данных о клиентах, автомобилях, аренде.

В приложении должен быть удобный просмотр данных, отфильтрованный по определенным параметрам. Например, просмотр всех автомобилей, которые арендовал определенный клиент.

*Итоговая таблица:*

1. содержит все записи, хранящие фамилии, имена клиентов, автомобили, которые они арендовали, количество суток аренды, цену арены и сумму, которую клиент заплатил, арендовав автомобиль.

*Статистические данные:*

1. содержат все вычисления.

Все необходимые данные хранятся в трёх таблицах, описание которых приведено в Таблице 1, Таблице 2, Таблице 3.

Таблица 1.

«Клиенты»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип поля | Индексированное поле |
| Код клиента | Числовой | Да, первичный ключ |
| Фамилия | Текстовый | Нет |
| Имя | Текстовый | Нет |

Таблица 2.

«Автомобили»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип поля | Индексированное поле |
| Код автомобиля | Числовой | Да, первичный ключ |
| Автомобиль | Текстовый | Нет |
| Цена аренды | Числовой | Нет |

Таблица 3.

«Аренда»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип поля | Индексированное поле |
| Код клиента | Числовой | Нет |
| Фамилия | Текстовый | Нет |
| Имя | Текстовый | Нет |
| Код автомобиля | Числовой | Нет |
| Автомобиль | Текстовый | Нет |
| Аренда | Числовой | Нет |

# ОПИСАНИЕ АЛГОРИТМОВ И КЛАССОВ, МЕТОДОВ

Помимо стандартных пространств имен, классы форм также используют: System.IO, System.Windows.Forms.

## 2.1. Чтение и запись данных

Реализация чтения и записи данных в программе описана на примере таблицы «Клиенты». Для работы с данными, хранящимися в текстовых файлах, используются класс «FileStream». В «FileStresm» метод «File.ReadAllLines» открывает текстовый файл, считывает все строки и помещает в строковый массив. (Пример 1.)

string[] w = File.ReadAllLines("arenda.txt", Encoding.GetEncoding(1251));

Пример 1. Пример использования метода «File.ReadAllLines»

Для того, чтобы прочитанный текстовый файл можно было записать в таблицу «DataGridView», для начала надо полученный строковый массив поделить множество массивов, используя при этом разделитель, чтобы программа понимала, какое слово файла в какой массив записывать. После этого, когда в массивы записаны соответствующие элементы, мы можем записать полученные данные в таблицу. (Пример 2.)

char[] d= new char[1] {'#'};

string[] kli = klients[i].Split(d, StringSplitOptions.RemoveEmptyEntries);

kod[i] = int.Parse(kli[0]);

fio[i] = kli[1];

name[i] = kli[2];

dataGridView1.Rows.Add(kod[i].ToString(), fio[i], name[i]);

Пример 2. Помещение данных в массивы и заполнение таблицы.

Форма содержит таблицу «DataGridView», содержащую 3 столбца: «Код клиента», «Фамилия», «Имя». Для столбца «Код клиента» устанавливается уникальное значение, и при добавлении нового объекта уникальное значение будет инкрементироваться на единицу. То есть фактически это стандартный столбец «id», как в большинстве баз данных

Для сохранения данных после редактирования используется метод File.WriteAllLines класса FileStream. Данный метод принимает в параметры путь к файлу, в который будут сохранены данные, кодировку и массив данных, которые будут записаны в файл. (Пример 3).

string[] kli = new string[l];

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

{

kli[i] = kod[i].ToString() + "#" + fio[i] + "#" + name[i];

}

File.WriteAllLines("klients.txt", kli, Encoding.GetEncoding(1251));

Пример 3. Сохранение данных в текстовый файл.

Данные таблицы «Клиенты» хранятся в файле «klients.txt». На Примере 4 приведена структура файла после сохранения данных.

1#Копытов#Максим

2#Лазарева#Таисия

3#Ясашин#Аскар

4#Бубликова#Таисия

5#Копылов#Сергей

6#Антонов#Гордей

………………………

20#Самойлов#Александр

Пример 4. Структура файла «klients.txt».

## 2.2. Добавление новых данных

В таблице «DataGridView» было создан метод «dataGridView1\_SelectionChanged», который отслеживает текущую строку, выделенную в таблице. Благодаря этому методу, в три текстовых поля, расположенных ниже таблицы, помещаются текущие значения таблицы, чтобы в дальнейшем работать с ними.

Реализация добавления новых данных в программе описана на примере таблицы «Клиенты». Для начала мы нажимаем на кнопку «Новый», метод которого button2\_Click. Этот метод инкрементирует код последнего элемента массива кодов клиентов, после чего помещает этот код в текстовое поле. А текстовые поля для ввода фамилии и имени он делает пустыми.

Метод button3\_Click обрабатывает нажатие кнопки «Добавить». После нажатии кнопки увеличивается размерность трёх массив на единицу (массивы кода, фамилии, имени). Код клиента, фамилия и имя принимают значение из соответствующих текстовых полей. После этого таблица перезаписывается. Код метода приведён ниже (Пример 5).

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

{

textBox2.Text = "";

textBox3.Text = "";

int kodd = kod[l - 1] + 1;

textBox1.Text = kodd.ToString();

}

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

{

l++;

Array.Resize(ref kod, l);

Array.Resize(ref fio, l);

Array.Resize(ref name, l);

kod[l - 1] = int.Parse(textBox1.Text);

fio[l - 1] = textBox2.Text;

name[l - 1] = textBox3.Text;

NewLoad();

}

Пример 5. Метод добавления данных.

## 2.3. Изменение данных

Реализация изменения данных в программе описана на примере таблицы «Клиенты». Метод button1\_Click обрабатывает нажатие кнопки «Изменить». После нажатии кнопки, фамилия и имя клиента принимают значения соответствующих текстовых полей. После этого таблица перезаписывается. Код метода приведён ниже (Пример 6).

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

{

fio[ts] = textBox2.Text;

name[ts] = textBox3.Text;

NewLoad();

}

Пример 6. Метод изменения данных

## 2.4. Удаление данных

Реализация удаления данных в программе описана на примере таблицы «Клиенты». Метод button4\_Click обрабатывает нажатие кнопки «Удалить». После нажатии кнопки мы входим в цикл, начиная с текущей строки, и элементы ниже мы переставляем на один элемент вперёд. После этого размерность трёх массивов уменьшается. После этого, мы считываем все строки файла «arenda.txt» для того, чтобы удалить данные о клиенте и в этом файле. После этого таблица перезаписывается. Код метода приведён ниже (Пример 7).

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

{

int help = int.Parse(dataGridView1.Rows[ts].Cells[0].Value.ToString());

for (int i = ts; i < l - 1; i++)

{

kod[i] = kod[i + 1];

fio[i] = fio[i + 1];

name[i] = name[i + 1];

}

l--;

Array.Resize(ref kod, l);

Array.Resize(ref fio, l);

Array.Resize(ref name, l);

string[] w = File.ReadAllLines("arenda.txt", Encoding.GetEncoding(1251));

l1 = w.Length;

Array.Resize(ref kod\_kl, l1);

Array.Resize(ref kod\_au, l1);

Array.Resize(ref arenda, l1);

for (int i = 0; i < l1; i++)

{

string[] w1 = w[i].Split(d, StringSplitOptions.RemoveEmptyEntries);

kod\_kl[i] = int.Parse(w1[0]);

kod\_au[i] = int.Parse(w1[1]);

arenda[i] = int.Parse(w1[2]);

}

for (int i = 0; i < l1; i++)

{

if (kod\_kl[i] == help)

{

MessageBox.Show("Удаление данных");

for (int j = i; j < l1 - 1; j++)

{

kod\_kl[j] = kod\_kl[j + 1];

kod\_au[j] = kod\_au[j + 1];

arenda[j] = arenda[j + 1];

}

l1--;

Array.Resize(ref kod\_kl, l1);

Array.Resize(ref kod\_au, l1);

Array.Resize(ref arenda, l1);

}

}

NewLoad();

}

Пример 7. Метод удаления данных.

## 2.5. Сохранение данных

Реализация сохранения данных в программе описана на примере таблицы «Клиенты». Метод button5\_Click обрабатывает нажатие кнопки «Сохранить». Для сохранения данных после редактирования используется метод File.WriteAllLines класса FileStream. Данный метод был описан выше (см. Пример 3).

# ОПИСАНИЕ АЛГОРИТМОВ ВЫЧИСЛЕНИЯ

На форме «Form7» реализованы методы для вычисления различных данных, что, собственно, и отличает автоматическую БД от ручной. Преимущество данной БД в том, что пользователю не надо смотреть и искать нужную информацию среди огромного числа данных, при этом подсчитываю значения вручную. Программа сделает это автоматически.

## 3.1. Вычисление самого частого клиента

Для реализации подсчёта используется метод «Loading()», который считывает информацию из трёх текстовых файлов: «klients.txt», «auto.txt», «arenda.txt». Данные из каждого текстового файла помещаются в массивы. (см. Пример 2).

Для подсчёта самого частого клиента я ввожу две новые переменные, которые хранят сам элемент и индекс элемента. После прохождения в цикле по массиву кодов клиентов я выбираю тот элемент, который чаще всего встречается в текстовом файле «arenda.txt». Код метода приведён ниже (Пример 8).

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

{

Loading();

var count = 0;

var index = -1;

for (var i = 0; i < kod\_kl.Length; ++i)

{

var k = 1;

for (var j = i + 1; j < kod\_kl.Length; ++j)

if (kod\_kl[i] == kod\_kl[j]) k++;

if (k <= count) continue;

count = k;

index = i;

}

textBox1.Text = fio[index] + " " + name[index];

}

Пример 8. Метод нахождения самого частого клиента.

## 3.2. Подсчёт показателей: самая прибыльная аренда, сумма всех аренд, средняя цена аренды.

Для реализации подсчётов показателей, я считываю три текстовых файла, в котором хранятся все данные: «klients.txt», «auto.txt», «arenda.txt». После чего происходит подсчёт показателей. Самая прибыльная аренда рассчитывается как максимальный элемент массива, в котором хранится итоговое значение аренды определённого автомобиля. Сумма всех аренд рассчитывается так: я складываю все элементы массива. А средняя цена аренды рассчитывается как среднее арифметическое суммы всех аренд. Код метода приведён ниже (Пример 9).

for (int i = 0; i < l3; i++)

{

string[] v3 = arend[i].Split(d, StringSplitOptions.RemoveEmptyEntries);

sut[i] = int.Parse(v3[5]);

for (int j = 0; j < l1; j++)

{

for (int k = 0; k < l2; k++)

{

if (int.Parse(v3[0]) == kod\_kl[j] && int.Parse(v3[3]) == kod\_au[k])

{

for(int m=0; m<l3; m++)

{

prib[m] = cena[k] \* sut[i];

sum += prib[m];

if (prib[m] > max)

{

max = prib[m];

}

}

}

}

}

}

textBox2.Text = max.ToString();

textBox5.Text = ((sum / l3)/l3).ToString();

textBox4.Text = (sum / l3).ToString();

Пример 9. Метод подсчёта самой прибыльной аренды, суммы всех аренд, средней цена аренды.

## 3.3. Подсчёт показателя: среднее количество суток аренды.

Для реализации подсчёта используется метод «Loading()», который считывает информацию из трёх текстовых файлов: «klients.txt», «auto.txt», «arenda.txt». Данные из каждого текстового файла помещаются в массивы. (см. Пример 2).

После чего происходит подсчёт показателей. А средняя количество суток аренды рассчитывается как среднее арифметическое суммы всех суток. Код метода приведён ниже (Пример 10).

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

{

int sum = 0;

int res = 0;

for (int j = 0; j < sut.Length; j++)

{

sum = sum + sut[j];

}

res = sum / sut.Length;

textBox3.Text = res.ToString();

}

Пример 10. Метод подсчёта среднего арифметического суммы аренды.

## 

## 3.4. Максимальное время аренды

Для реализации подсчёта используется метод «Loading()», который считывает информацию из трёх текстовых файлов: «klients.txt», «auto.txt», «arenda.txt». Данные из каждого текстового файла помещаются в массивы. (см. Пример 2).

Для полсчёта показателя максимальных суток аренды я прохожу по массиву суток, и выбираю максимальный элемент. Код метода приведён ниже (Пример 11).

private void button7\_Click(object sender, EventArgs e)

{

Loading();

var max = 0;

for (var i = 0; i < sut.Length; i++)

{

if (sut[i] > max)

{

max = sut[i];

}

}

textBox6.Text = max.ToString();

}

Пример 11. Метод нахождения максимального времени аренды.

## 3.5. Самый прибыльный клиент

Для реализации подсчётов показателей, я считываю три текстовых файла, в котором хранятся все данные: «klients.txt», «auto.txt», «arenda.txt». После чего происходит подсчёт показателей. Для подсчёта самого прибыльного клиента создаётся массив, который для каждого клиента подсчитывает сумму всех его аренд. В итоге я нахожу максимум, и вывожу клиента. Код метода приведён ниже (Пример 12).

int sum = 0;

int max = 0;

int index = 0;

for (int hh=0; hh < sum\_ar.Length; hh++)

{

if(sum\_ar[hh] > max)

{

max = sum\_ar[hh];

index = hh;

}

}

textBox7.Text = fio[index] + " " + name[index];

Пример 12. Метод нахождения самого прибыльного клиента.

# РУКОВОДСТВО ПРИЛОЖЕНИЯ

Окно «Главное меню»

При запуске приложения открывается окно, которое является «главным меню» базы данных. В данном окне (рисунок 1) пользователь может открыть окна: с таблицей клиентов, с таблицей автомобилей, с таблицей аренды, итоговой таблицей, а также окно для просмотра автомобилей, которые арендовал клиент, окно со статистической информацией.



Рисунок 1 – «Главное меню»

Окно «Клиенты»

При нажатии кнопки «Клиенты» открывается окно с таблицей клиентов (рисунок 2).

Для добавления нового клиента в таблицу необходимо нажать кнопку «Новый», а затем, после введения соответствующей информации в текстовые поля, нужно нажать на кнопку «Добавить».

Для изменения данных клиента пользователю необходимо нажать на строку соответствующего студента, изменить в текстовом поле значения «Фамилия», «Имя» и нажать кнопку «Изменить».

Для удаления данных о клиенте пользователю необходимо нажать на строку соответствующего студента и нажать кнопку «Удалить».

Для сохранения данных таблицы необходимо нажать на кнопку «Сохранить».

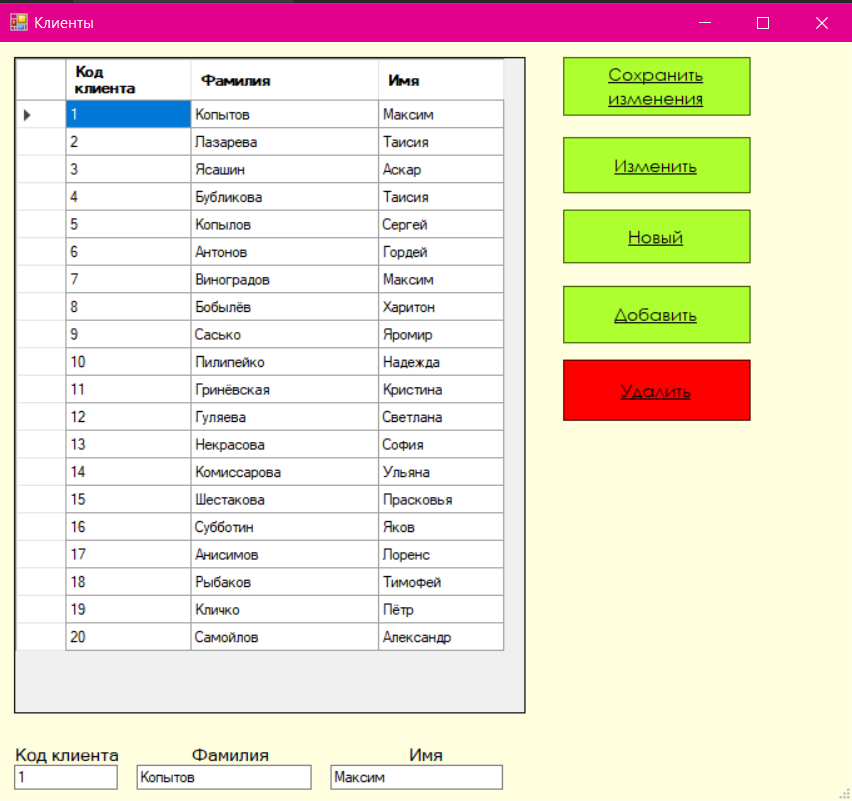


Рисунок 2 – окно «Клиенты»

Окно «Автотранспорт»

При нажатии кнопки «Автотранспорт» открывается окно с таблицей автомобилей (рисунок 3).

Для добавления нового автомобиля в таблицу необходимо нажать кнопку «Новый», а затем, после введения соответствующей информации в текстовые поля, нужно нажать на кнопку «Добавить».

Для изменения данных об автомобиле пользователю необходимо нажать на строку соответствующего автомобиля, изменить в текстовом поле значения «Марка», «Цена аренды» и нажать кнопку «Изменить».

Для удаления данных об автомобиле пользователю необходимо нажать на строку с информацией об автомобиле и нажать кнопку «Удалить».

Для сохранения данных таблицы необходимо нажать на кнопку «Сохранить».

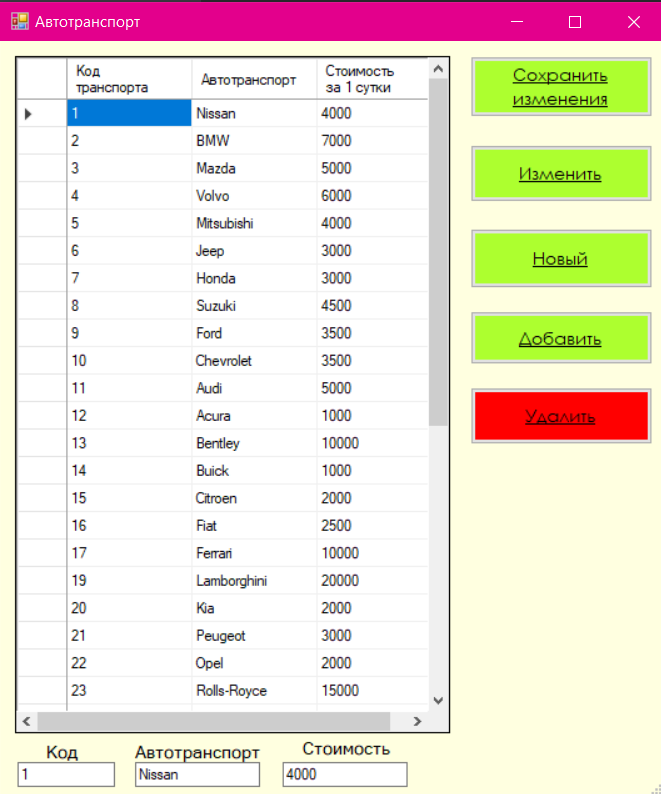


Рисунок 3 – окно «Автотранспорт»

Окно «Аренда»

При нажатии кнопки «Аренда» открывается окно с таблицей аренды автомобилей (рисунок 4).

Для добавления информации об аренде клиентом автомобиля в таблицу необходимо выбрать клиента из combobox, автомобиль из combobox, ввести количество суток, на которые клиент хочет арендовать автомобиль, а затем, после введения соответствующей информации в текстовые поля, нужно нажать на кнопку «Добавить/Изменить». Если клиент не арендовал такой автомобиль, то информация появится в таблице. Если же клиент арендовал такой автомобиль, то изменяется количество суток аренды.

Для удаления данных об автомобиле пользователю необходимо выбрать клиента, автомобиль из combobox, а затем нажать на кнопку «Удалить».

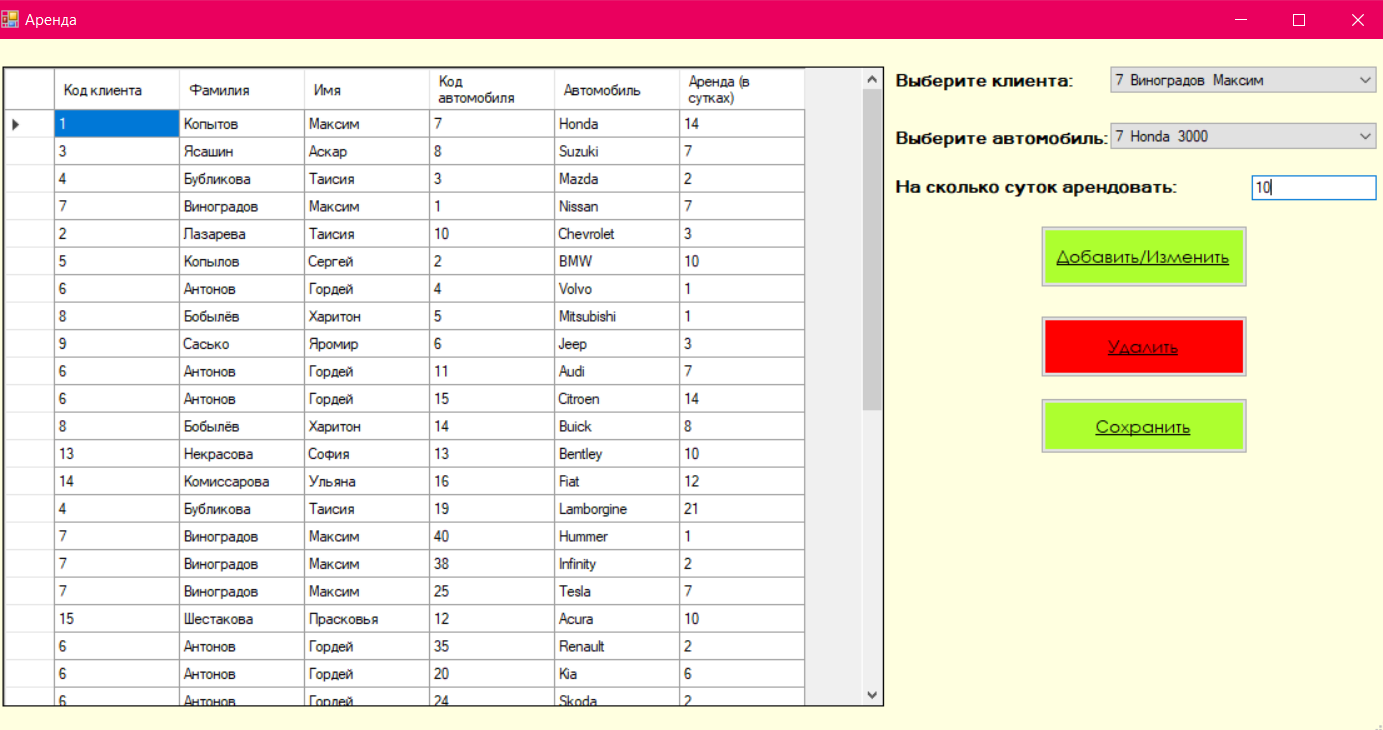
Для сохранения данных таблицы необходимо нажать на кнопку «Сохранить».

Рисунок 4 – окно «Аренда»

Окно «Обобщённая информация»

При нажатии на кнопку «Обобщённая информация» открывается окно, в котором пользователь может просмотреть всю информацию об аренде автомобилей (рисунок 5).

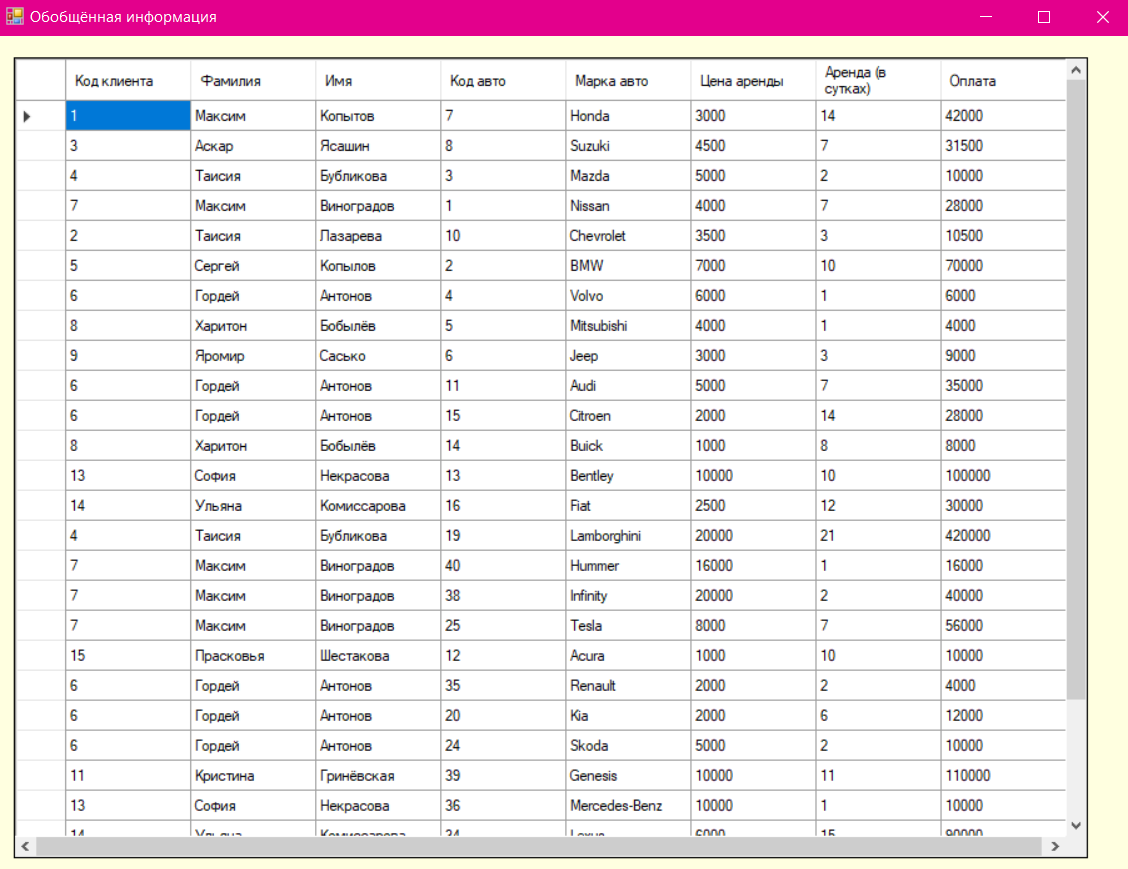


Рисунок 5 – окно «Обобщённая информация»

Окно «Информация о клиенте»

При нажатии на кнопку «Информация о клиенте» открывается окно, в котором можно просмотреть все автомобили, которые арендовал клиент, выбрав в combobox интересующего Вас клиента (Рисунок 6).

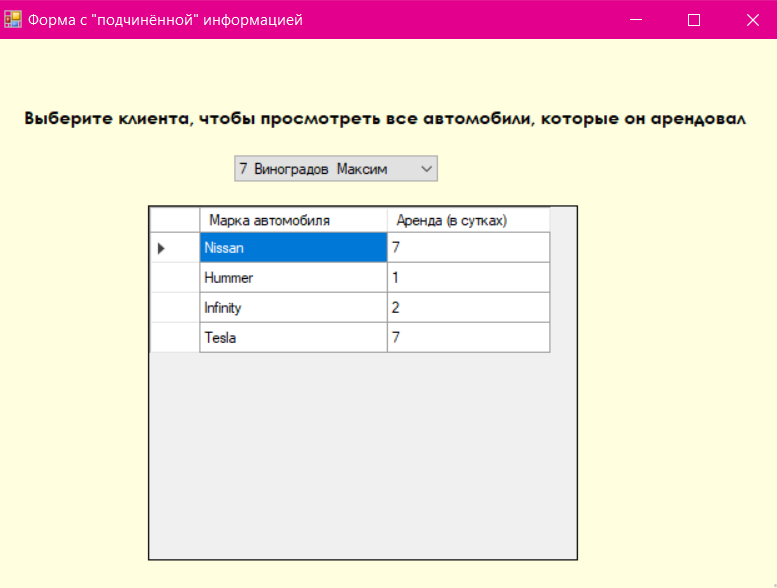


Рисунок 6 – окно «Информация о клиенте»

Окно «Статистические данные»

При нажатии на кнопку «Статистические данные» открывается окно с информацией (Рисунок 7).

При нажатии на кнопку «Самый частый клиент» в текстовом поле справа появляется соответствующая информация, которые выводит фамилию и имя клиента.

При нажатии на кнопку «Самая прибыльная аренда, сумма всех аренд, средняя цена аренды» в трёх текстовых полях справа появляется соответствующая информация, которая выводит вычисления.

При нажатии на кнопку «Среднее количество суток аренды», «Максимальное время аренды» в текстовых полях справа выводится соответствующая информация.

При нажатии на кнопку «Самый прибыльный клиент»

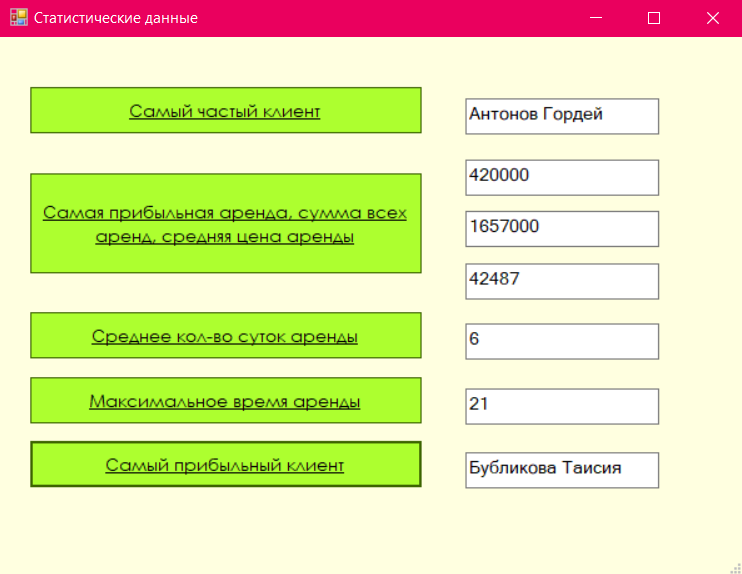


Рисунок 7 – окно «Статистическая информация»

# ЗАКЛЮЧЕНИЕ

Главной задачей данной курсовой работы являлась разработка приложения по учету аренды автомобилей.

Главным результатом проведенной работы является функционирующее приложение, которое выполняет требуемый круг задач по ведению учета аренды автомобилей клиентами.

Разработанная база данных предоставляет следующие возможности:

* добавление клиентов в базу данных;
* добавление автомобилей в базу данных;
* занесение информации об аренде клиентом автомобиля в базу данных;
* просмотр информации об аренде всех автомобилей клиентами;
* просмотр отчета, содержащего информацию о клиенте, какие автомобиле он арендовал;
* просмотр вычислений, показывающих такие сведения, как: самый частый клиент, самая прибыльная аренда, сумма всех аренд, средняя сумма аренды, самый прибыльный клиент, среднее количество суток аренды, максимальное время аренды.

Программа интуитивно проста и понятна для любого пользователя, для ее использования не требуется специального обучения. Программный продукт был написан на языке C# и в системе управления базами данных (СУБД) Microsoft Access.