СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc89934815)

[1 Постановка задачи 4](#_Toc89934816)

[1.1 Организационная сущность задачи 4](#_Toc89934817)

[1.2 Функциональная модель 4](#_Toc89934818)

[1.3 Входные данные 5](#_Toc89934819)

[1.4 Выходные данные 5](#_Toc89934820)

[2 Вычислительная система 6](#_Toc89934821)

[2.1 Используемые технические средства 6](#_Toc89934822)

[2.2 Инструменты разработки 6](#_Toc89934823)

[3 Проектирование задачи 7](#_Toc89934824)

[3.1 Организация данных 7](#_Toc89934825)

[3.2 Концептуальный прототип 7](#_Toc89934826)

[3.3 Функции: логическая и физическая организация 8](#_Toc89934827)

[4 Описание программного средства 10](#_Toc89934828)

[4.1 Общие сведения 10](#_Toc89934829)

[4.2 Функциональное назначение 10](#_Toc89934830)

[4.3 Организация хранения данных 10](#_Toc89934831)

[5 Методика испытаний 12](#_Toc89934832)

[5.1 Технические требования 12](#_Toc89934833)

[5.2 Демонстрационный пример использования 12](#_Toc89934834)

[6 Применение 13](#_Toc89934835)

[6.1 Назначение программы 13](#_Toc89934836)

[6.2 Справочная система 13](#_Toc89934837)

[ЗАКЛЮЧЕНИЕ 14](#_Toc89934838)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 15](#_Toc89934839)

[ПРИЛОЖЕНИЕ А 16](#_Toc89934840)

[ПРИЛОЖЕНИЕ Б 28](#_Toc89934841)

ВВЕДЕНИЕ

Актуальность данной темы автоматизации формирование подписей достаточно велика так, как при постоянной работе человек ошибается то автоматизация позволит увеличить эффективность и уменьшить количество ошибок.

# 1 Постановка задачи

## 1.1 Организационная сущность задачи

Целью данного курсового проекта является создание программного приложения на тему «Автоматическое составление форм». Чтобы создать данную программу, необходимо исследовать предметную область.

Предметная область данной задачи будет включать в себя форму для записи клиента.

Данная программа будет выполнять следующие функции:

* автоматизировать процесс заполнения форм;

В настоящее время подобные программные приложения используются на некоторых сайтах. Дело в том, что мелкие агентства не имеют достаточно средств для покупки сайта или программиста, а также число покупателей в них не столь велико, чтобы нуждаться в установке подобных программ.

## 1.2 Функциональная модель

Для разрабатываемого программного продукта входной информацией будет служить ввод критериев.

Данный проект выполняет следующие основные функции:

* добавление заказа;
* автоматизация заполнения формы;
* удаление заказа;
* сохранение отчета в файл;

Функциональная модель представлена на рисунке 1.1.

Статистика

Измененные данные

Туристический оператор (сделки, бронирование, загруженность направлений)

Сведения о работе оператора

Информация, вводимая оператором

Оператор

Рисунок 1.1 – Функциональная модель

## 1.3 Входные данные

Входными данными будет являться информация, выбранная пользователем.

Пользователю будет предоставлена возможность добавления заявки в базу, для этого ему потребуется ввести данные о виде проживания.

Входные данные для базы тур представлены в таблице 1.1.

Таблица 1.1 – Входные данные базы формы заявки

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип поля | Размер поля | Назначение поля |
| Куда | string | 64 | Куда |
| Откуда | string | 64 | Откуда |
| Вылет с | string | 64 | Время вылета с |
| По | string | 64 | Время вылета по |
| Ночей | int | 30 | Кол-во ночей |
| Отель | string | 10 | Название отеля |
| Взрослых | int | 30 | Кол-во Взрослых |
| Детей | int | 30 | Кол-во Детей |

## 1.4 Выходные данные

К выходным данным можно отнести информацию, которая предоставляется пользователю в результате действий, произведенных в программе.

# 2 Вычислительная система

## 2.1 Используемые технические средства

Программное приложение не будет требовать инсталляции. Для того чтобы воспользоваться программой, необходимо скопировать каталог с программой на жесткий диск. Чтобы вынести ярлык программы на рабочий стол, необходимо нажать правой кнопкой мыши на «Vector» и выбрать «Отправить» на «Рабочий стол (создать ярлык)».

## 2.2 Инструменты разработки

Данное программное приложение будет разрабатываться в операционной системе Microsoft Windows 10. Программа будет разрабатываться в среде Microsoft VisualStudio 2017.

Microsoft VisualStudio 2017 – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

# 3 Проектирование задачи

## Организация данных

Исходя из исследования предметной области было принято решение о хранении данных в виде структур, ведь такой вид хранения предоставляет нам доступ к нужному полю структуры из любой точки кода. Были выделены следующие структуры данных:

* Tours.cs;

Все данные будут хранится в виде базы данных.

## 3.2 Концептуальный прототип

Концептуальный прототип состоит из описания внешнего пользовательского интерфейса – системы меню, диалоговых окон и элементов управления. Программный продукт будет состоять из шести модулей. Каждому модулю будет соответствовать свой пункт меню.

Меню:

* Форма с полями.

Производится создание нового объекта структуры Tour, ввод исходных данных пользователя таких как ID тура, город вылета и выбор пункта назначения, количества ночей, интервала стоимости, типа размещения. Далее производится обработка введенных данных, анализ и вывод результатов поиска в виде списка доступных отелей. После введенные поля сохраняются в структуру Tour, а сам объект записывается в структуру Tours. Пример заполнения тура представлен на рисунке 3.1.

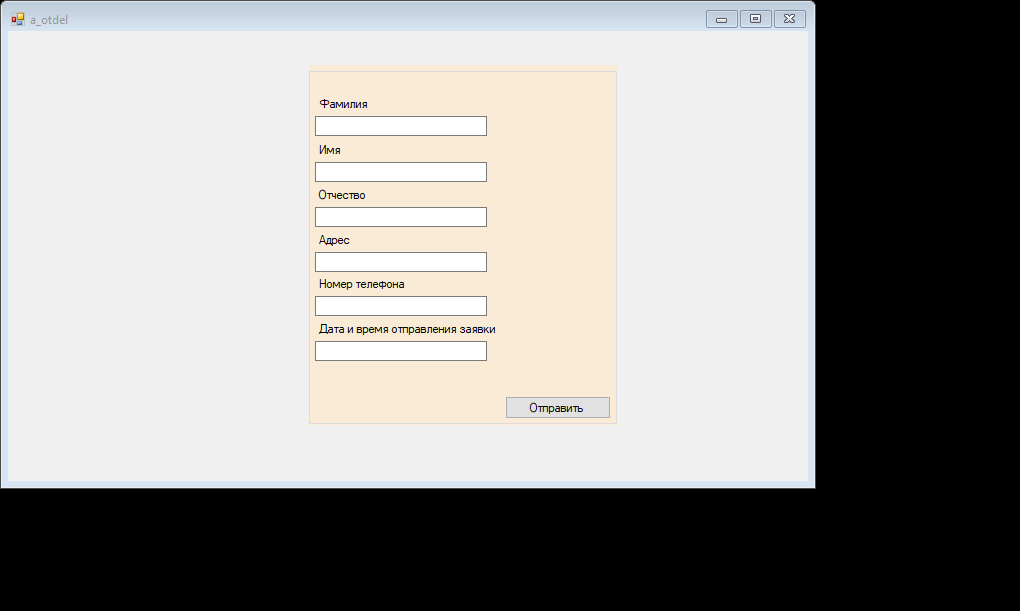


Рисунок 3.1 – Форма заказа.

## 3.3 Функции: логическая и физическая организация

Функция добавления записей в базу данных закреплена за пунктом меню номер 1 «Добавление записи». Код функции добавления представлен ниже:

using System;

using System.Data.OleDb;

using System.Windows.Forms;

namespace ЛР3

{

public partial class a\_otdel : Form

{

public static string connectString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=laba.mdb;";

private OleDbConnection myConnetcion;

public a\_otdel()

{

InitializeComponent();

myConnetcion = new OleDbConnection(connectString);

myConnetcion.Open();

}

private void a\_otdel\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

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

{

try

{

string query = "INSERT INTO klient (w\_name, w\_surname, w\_tname, w\_adress, w\_phonenumber) VALUES ('sname', 'Name', 'Tname', 'Adress', 'Phone')";

string sname = this.textBox1.Text;

string name = this.textBox2.Text;

string tname = this.textBox3.Text;

string adress = this.textBox4.Text;

string phone = this.textBox5.Text;

string time = Convert.ToString(DateTime.Now);

query = query.Replace("Name", name);

query = query.Replace("sname", sname);

query = query.Replace("Tname", tname);

query = query.Replace("Adress", adress);

query = query.Replace("Phone", phone);

OleDbCommand command = new OleDbCommand(query, myConnetcion);

command.ExecuteNonQuery();

}

catch { MessageBox.Show("error"); }

}

}

}

Фрагмент кода функций представлен в приложении А.

# 4 Описание программного средства

## 4.1 Общие сведения

Программное приложение на тему «Автоматическое составление форм». предназначена для облегчения процесса бронирования или заключения сделок.

Программа не требует инсталляции, необходимо ее только скопировать на компьютер.

Вызов и загрузка программы осуществляется после запуска исполняемого файла Vector.exe. Размер этого файла составляет 16 КБ.

Проект состоит из 3 модулей, листинг которых представлен в Приложении А.

## 4.2 Функциональное назначение

Программное приложение предназначено для автоматизации работы оператора.

Программа обладает интерфейсом понятным на интуитивном уровне как пользователю, не обладающему глубокими знаниями компьютерных технологий и навыками работы с персональным компьютером, так и опытному пользователю.

Для работы с программным приложением не требуется установка дополнительных компонентов. Это обусловлено тем, что данное программное приложение является простым в использовании.

Благодаря денной программе оператору будет проще оформить клиенту заказ, при этом сэкономив время.

## 4.3 Организация хранения данных

Для организации хранения данных в виде файлов при создании приложения потребовалась следующая структура (дерево) каталогов и файлов, представленная на рисунке 4.1.

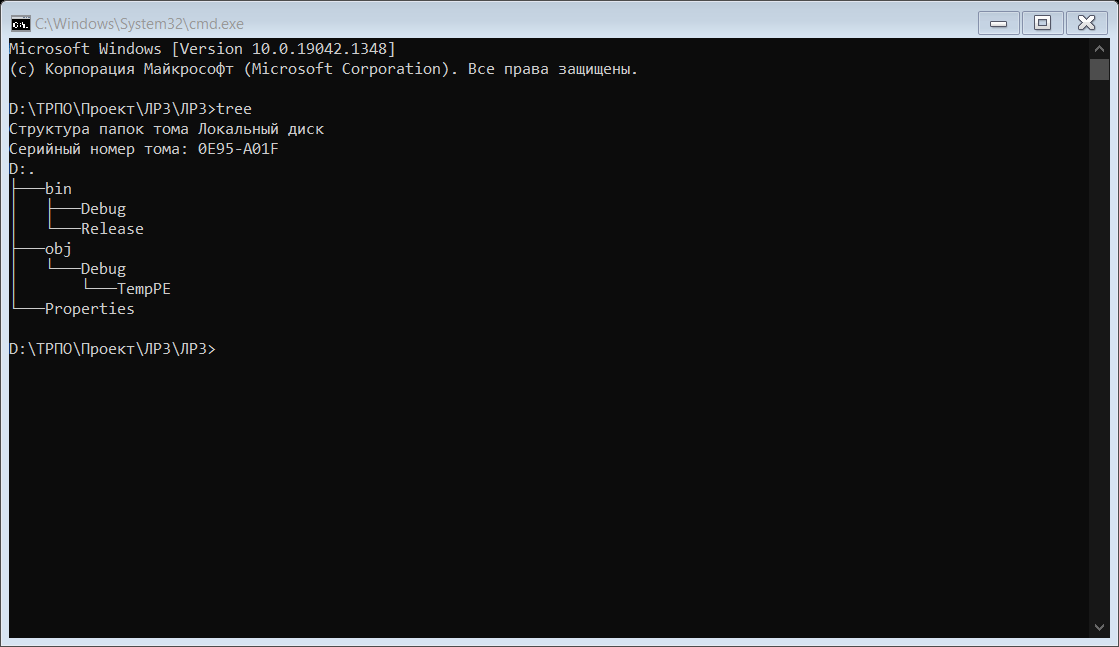


Рисунок 4.1 – Структура размещения каталогов и файлов приложения

# 5 Методика испытаний

## 5.1 Технические требования

Для оптимальной работы программы необходимы следующие минимальные требования:

* процессор: Pentium IV и выше;
* объем оперативной памяти: 512 Мбайт и выше;
* свободного места на жестком диске: 100 Мбайт;
* операционная система: Windows 7 и выше;
* наличие .NET Framework 4.4 версии и выше;
* наличие манипулятора «мышь»;
* клавиатура IBM PC любой модификации.

## 5.2 Демонстрационный пример использования

Программное средство должно функционировать с определенными характеристиками для заданных областей данных. Отладкой называется процесс, позволяющий получить программу, функционирующую с требуемыми характеристиками в заданной области входных данных. В результате отладки программа должна соответствовать определенной фиксированной совокупности правил и показателей качества.

Для выявления логических ошибок необходимо провести полное тестирование программного приложения, всех его функций, реализованных в программе.

При запуске программы появляется первое окно с возможностью «Tour Agency» с возможностью ввода данных о туре. Вид главного окна представлен на рисунке 5.1.



Рисунок 5.1 – Вид главного меню «Tour Agency»

# 6 Применение

## 6.1 Назначение программы

Разрабатываемое программное приложение предназначено для облегчения работы оператора, ускорения процесса заполнения форм и бронирования.

Удобный и приятный интерфейс программы не вызовет затруднений в использовании даже у не опытного пользователя. Однако при необходимости предусмотрена справка, подробно описывающая основные функции программного приложения.

## 6.2 Справочная система

Программное средство имеет интуитивно понятный и легкий в освоении интерфейс, а также несложную логическую организацию.

Так как есть возможность того, что пользователь не сможет понять, как нужно работать с программой в меню предусмотрен пункт «Справка». При нажатии этого пункта меню появиться справочная информация.

# ЗАКЛЮЧЕНИЕ

Задание на курсовое проектирование заключалось в написании программы на тему «Автоматическое составление форм». Реализовать поставленную задачу удалось при помощи среды программирования Microsoft VisualStudio 2019.

В процессе разработки было приложено максимум усилий для создания качественного и легкого в использовании программного продукта, который отвечал бы заданным требованиям.

Продукт содержит понятные и конкретные названия пунктов меню. Все это позволяет быстро и легко вникнуть в суть программы, разобраться в ее работе, а также быстро и эффективно выполнить необходимые действия. Это значит, что даже неопытный пользователь может применять в работе данное приложение.

Программа выполняет следующие функции:

* автоматизирует процесс бронирования;

Программа реализована полностью в соответствии с алгоритмом задачи, отлажена и протестирована.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Шиманович, Е.Л – С/С# в примерах и задачах. – Мн.: Новое знание, 2016.
2. METANIT.COM [Электронный ресурс]. – Электронные данные – Режим доступа: <https://metanit.com/cpp>.
3. ГОСТ 2.105-95. ЕСКД. Общие требования к текстовым документам
4. ГОСТ 19.401-2000. ЕСПД. Текст программы
5. ГОСТ 19.402-2000 ЕСПД. Описание программы
6. ГОСТ 19.301-2000 ЕСПД. Программа и методика испытаний

# ПРИЛОЖЕНИЕ А

(справочное)

Текст программы

**Form1.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.IO;

namespace WindowsFormsApp1

{

public partial class MainMenu : Form

{

public MainMenu()

{

InitializeComponent();

}

private void button11\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "" && textBox4.Text != "" && textBox5.Text != "" && textBox10.Text != "")

{

StreamWriter f = new StreamWriter("tour.txt");

f.WriteLine(textBox1.Text);

f.WriteLine(textBox2.Text);

f.WriteLine(textBox3.Text);

f.WriteLine(textBox8.Text);

f.WriteLine(Convert.ToInt16(textBox4.Text));

f.WriteLine(textBox9.Text);

f.WriteLine(Convert.ToInt16(textBox5.Text));

f.WriteLine(Convert.ToInt16(textBox10.Text));

if (trackBar1.Value == 0)

f.WriteLine(1000);

else

f.WriteLine(trackBar1.Value \* 1000);

f.WriteLine(trackBar2.Value \* 1000);

f.Close();

this.Hide();

Tours tours = new Tours();

tours.Show();

}

else

{

button11.Text = "Все поля должны быть заполнены";

}

}

private void button12\_MouseEnter(object sender, EventArgs e)

{

button12.BackColor = Color.Red;

}

private void button12\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void button13\_Click(object sender, EventArgs e)

{

this.WindowState = (FormWindowState)1;

}

private void button12\_MouseLeave\_1(object sender, EventArgs e)

{

button12.BackColor = Color.Thistle;

}

private void button13\_MouseEnter(object sender, EventArgs e)

{

button13.BackColor = Color.PaleVioletRed;

}

private void button13\_MouseLeave(object sender, EventArgs e)

{

button13.BackColor = Color.Thistle;

}

}

}

**Tours.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.IO;

namespace WindowsFormsApp1

{

public partial class Tours : Form

{

struct Tour

{

public string from;

public string to;

public string flyFrom;

public string flyTo;

public int nights;

public string hotel;

public int older;

public int young;

public int minCost;

public int maxCost;

public Tour(string from, string to, string flyFrom, string flyTo, int nights, string hotel, int older, int young, int minCost, int maxCost)

{

this.from = from;

this.to = to;

this.flyFrom = flyFrom;

this.flyTo = flyTo;

this.nights = nights;

this.hotel = hotel;

this.older = older;

this.young = young;

this.minCost = minCost;

this.maxCost = maxCost;

}

}

public Tours()

{

InitializeComponent();

Random r = new Random();

StreamReader f = new StreamReader("tour.txt");

string from = f.ReadLine();

string to = f.ReadLine();

string flyFrom = f.ReadLine();

string flyTo = f.ReadLine();

int nights = Convert.ToInt16(f.ReadLine());

string hotel = f.ReadLine();

int older = Convert.ToInt16(f.ReadLine());

int young = Convert.ToInt16(f.ReadLine());

int minCost = Convert.ToInt16(f.ReadLine());

int maxCost = Convert.ToInt16(f.ReadLine());

f.Close();

Tour tour = new Tour(from, to, flyFrom, flyTo, nights, hotel, older, young, minCost, maxCost);

if (minCost >= 1000 && minCost < 3500)

{

label2.Visible = true;

textBox1.Visible = true;

textBox1.Text = "Ночей: " + tour.nights;

textBox1.Text += "\r\nОтель: " + tour.hotel;

textBox1.Text += "\r\nЦена: " + (tour.maxCost - tour.minCost + r.Next(0, 1001));

pictureBox4.Visible = true;

}

if (minCost >= 3500 && minCost < 6000)

{

label3.Visible = true;

textBox2.Visible = true;

textBox2.Text = "Ночей: " + tour.nights;

textBox2.Text += "\r\nОтель: " + tour.hotel;

textBox2.Text += "\r\nЦена: " + (tour.maxCost - tour.minCost + r.Next(1001, 3501));

pictureBox1.Visible = true;

}

if (minCost >= 6000 && minCost < 7500)

{

label4.Visible = true;

textBox3.Visible = true;

textBox3.Text = "Ночей: " + tour.nights;

textBox3.Text += "\r\nОтель: " + tour.hotel;

textBox3.Text += "\r\nЦена: " + (tour.maxCost - tour.minCost + r.Next(3501, 6001));

pictureBox3.Visible = true;

}

if (minCost >= 7500 && minCost < 10000)

{

label5.Visible = true;

textBox4.Visible = true;

textBox4.Text = "Ночей: " + tour.nights;

textBox4.Text += "\r\nОтель: " + tour.hotel;

textBox4.Text += "\r\nЦена: " + (tour.maxCost - tour.minCost + r.Next(6001, 10001));

pictureBox2.Visible = true;

}

}

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

{

button1.BackColor = Color.Red;

}

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

{

Application.Exit();

}

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

{

button1.BackColor = Color.Thistle;

}

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

{

this.WindowState = FormWindowState.Minimized;

}

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

{

button2.BackColor = Color.PaleVioletRed;

}

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

{

button2.BackColor = Color.Thistle;

}

}

}

# 

# ПРИЛОЖЕНИЕ Б

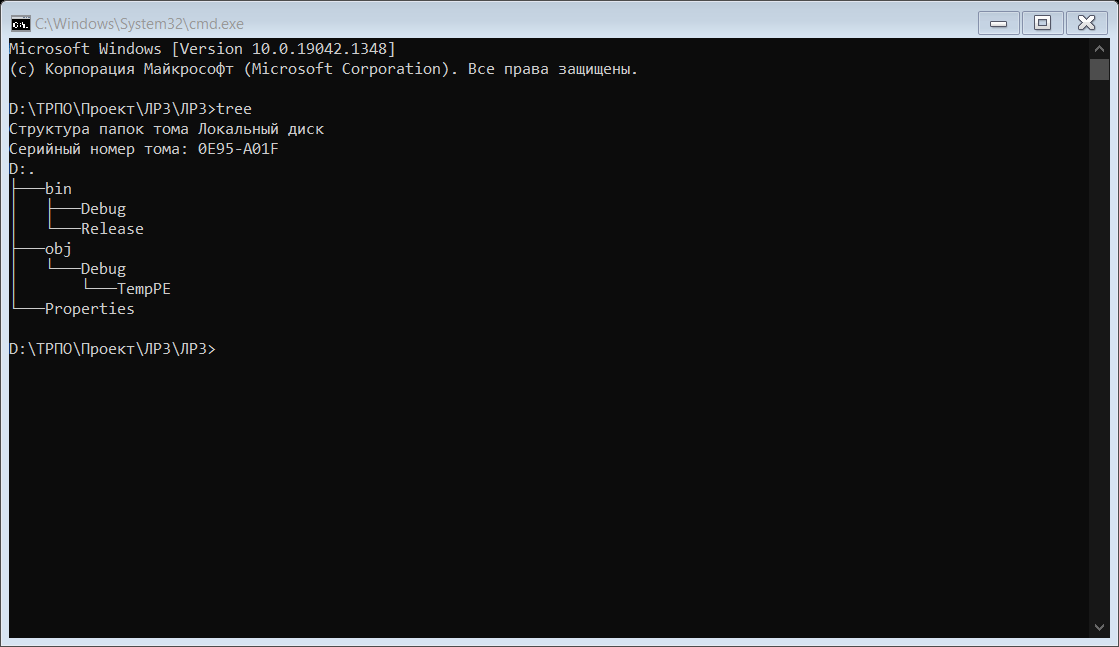


Схема данных