Министерство образования и науки РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Глазовский государственный педагогический институт имени В.Г. Короленко»

**ОТЧЕТ**

**по учебной практике**

**ПМ 01 Разработка программных модулей программного обеспечения для компьютерных систем**

**Специальность 09.02.03 Программирование в компьютерных системах**

|  |
| --- |
| Выполнил(а): |
| студент(ка) ОП СПО группа 21 |
| Целоусова Анастасия |
|  |
| Руководитель практики: ст. преподаватель кафедры математики и информатики Касаткин К.А. |

г. Глазов 2018

Оглавление

[Введение 3](#_Toc514929146)

[§1 Цель разработки 4](#_Toc514929147)

[§2 Обзор аналогов 5](#_Toc514929148)

[§3 Постановка задачи и требования, предъявляемые к программе 6](#_Toc514929149)

[§4 Описание алгоритма работы 7](#_Toc514929150)

[§5 Выбор и обоснование выбора средства разработки, исходный код программы 10](#_Toc514929151)

[§6 Тестирование программы 14](#_Toc514929152)

[§7 Пояснительная записка 16](#_Toc514929153)

# Введение

**Генеалогическое древо** – схематичное представление [родственных связей](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D1%81%D0%B2%D1%8F%D0%B7%D0%B8), [родословной росписи](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B4%D0%BE%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%B0%D1%8F_%D1%80%D0%BE%D1%81%D0%BF%D0%B8%D1%81%D1%8C) в виде условно-символического «[дерева](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE)», у «корней» которого указывается [родоначальник](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B4%D0%BE%D0%BD%D0%B0%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%BA), на «стволе» – представители основной (по старшинству) линии рода, а на «ветвях» – различных линиях родословия, известные его [потомки](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BC%D0%BE%D0%BA) – «листья» (настоящий пример иллюстрирует древо «нисходящего родословия», каковые являются наиболее распространёнными); но зачастую, если она, роспись, не стилизована в виде дерева реального, что было очень распространено в прошлом, схема представляет родословное древо в перевёрнутом виде, когда родоначальник располагается в верхней части таблицы.

# §1 Цель разработки

1. Упорядочивание информации. Возможность быстро найти всех членов семьи.
2. Обмен информацией с другими людьми.
3. Надежность хранения данных.

# §2 Обзор аналогов

**Бесплатные:**  
1. [PAF](http://www.familysearch.org/paf) – делается церковью мормонов. Позволяет управлять источниками;

2. [GenoPro](http://www.genopro.com/) – строит генограммы вместо деревьев – изображения;

3. [GenealogyJ](http://genj.sourceforge.net/) – GEDCOM ориентированная. Т.е. даже добавляя событие вы видите, что таги GEDCOM'а. Интересна тем, что позволяет писать к ней свои плагины;  
4. [FamilyTreeBuilder](http://www.myheritage.com/family-tree-builder) – программа сайта [MyHeritage](http://www.myheritage.com/), налучшая из бесплатных, с набором возможностей и загрузкой дерева на сайт.  
5. [Gramps](http://www.gramps-project.org/wiki/index.php?title=Main_Page) – изначально написанная на Линукс, но портированная уже на Windows.

**Платные:**

1. [RootsMagic](http://www.rootsmagic.com/) – Позволяет добавлять фото не только к людям но и к семьям. В последних версиях можно строить деревья;

2. [Ages](http://www.daubnet.com/english/ages.html) – развитое управление источниками, фотографиями, местами, профессиями. Много типов деревьев.

# §3 Постановка задачи и требования, предъявляемые к программе

Задачей является создание генеалогического дерева семьи. Требуется ввести паспортные данные членов некоторого родового клана; ссылки на детей (или на родителей). Необходимо протестировать программу .

# §4 Описание алгоритма работы

Подключился?

Подключение к базе данных

Подключение к БД: ошибка!

НЕТ

Рис.1 Блок-схема для кнопки «Соединение»

На рис.1 изображена блок-схема для соединения с базой данных, если соединение не устанавливается, то вывод сообщение: «Подключение к БД: ошибка!»

Подключение к базе данных

Транзакция: ошибка!

НЕТ

Подключился?

Рис. 2 Блок-схема для кнопки «Транзакция»

На рис.2 изображена блок-схема для транзакции с базой данных, если соединение не устанавливается, то вывод сообщение: «Транзакция: ошибка!»

НЕТ

Ввод переменной str\_sql

Запись в переменную str\_sql запроса

SQL-запрос: ОШИБКА!

Подключился запрос?

Вывод информации из базы данных

Рис. 3 Блок-схема для кнопки «SQL»

На рис. 3 изображена блок-схема для кнопки «SQL», которая при записи в переменную str\_sql запроса, выводит информацию из базы данных, в случае отсутствия подключения запроса, выводится сообщение: «SQL-запрос: ошибка!»

Отключение от базы данных

Отключение запросов

Рис. 4 Блок-схема для кнопки «Отключение»

На рис. 4 изображена блок-схема для кнопки «Отключение», которая при нажатии происходит отключение запросов, а затем отключение от базы данных.

# §5 Выбор и обоснование выбора средства разработки, исходный код программы

Приложение создавалось в среде разработки Lazarus, база данных была создана в HeidiSQL. Именно эти программы были выбраны для создания проекта, так как изучались основы работы в Lazarus и Heidi.

Исходный код программы:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, db, mysql51conn, sqldb, FileUtil, Forms, Controls,

Graphics, Dialogs, DBGrids, StdCtrls, ComCtrls, DbCtrls;

type

{ TForm1 }

TForm1 = class(TForm)

Button5: TButton;

Button6: TButton;

Button7: TButton;

Button8: TButton;

DataSource1: TDataSource;

DBGrid2: TDBGrid;

DBNavigator1: TDBNavigator;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

MySQL51Connection1: TMySQL51Connection;

PageControl1: TPageControl;

SQLQuery1: TSQLQuery;

SQLTransaction1: TSQLTransaction;

TabSheet1: TTabSheet;

TabSheet2: TTabSheet;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure DBGrid2SelectEditor(Sender: TObject; Column: TColumn;

var Editor: TWinControl);

private

public

end;

var

Form1: TForm1;

str\_sql : string;

implementation

{$R \*.lfm}

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);

begin

try

MySQL51Connection1.Connected:=true;

except

ShowMessage('Подключение к БД: ОШИБКА!');

exit;

end;

end;

procedure TForm1.Button2Click(Sender: TObject);

begin

try

SQLTransaction1.Active:=true;

except

ShowMessage('Транзакция: ОШИБКА!');

exit;

end;

end;

procedure TForm1.Button3Click(Sender: TObject);

begin

try

SQLQuery1.Active := false;

SQLQuery1.SQL.Clear;

str\_sql := 'SET character\_set\_client='+#39+'utf8'+#39+',character\_set\_connection='+#39+'cp1251'+#39+',character\_set\_results='+#39+'utf8'+#39+';';

SQLQuery1.sql.add(str\_sql);

SQLQuery1.ExecSQL;

SQLQuery1.SQL.Clear;

SQLQuery1.sql.add('SELECT fggyy.ID, fggyy.Name, fggyy.`Data`, fggyy.Pasport, rodstvo.name AS `Rodstvo` FROM fggyy JOIN rodstvo ON fggyy.Rodstvo = rodstvo.ID;');

SQLQuery1.Open;

except

ShowMessage('SQL-запрос: ОШИБКА!');

exit;

end;

end;

procedure TForm1.Button4Click(Sender: TObject);

begin

SQLQuery1.Active:=false;

SQLTransaction1.Commit;

MySQL51Connection1.Connected:=false;

end;

procedure TForm1.DBGrid2SelectEditor(Sender: TObject; Column: TColumn;

var Editor: TWinControl);

begin

Edit1.Text := DBGrid2.Columns.Items[1].Field.Value;

Edit2.Text := DBGrid2.Columns.Items[2].Field.Value;

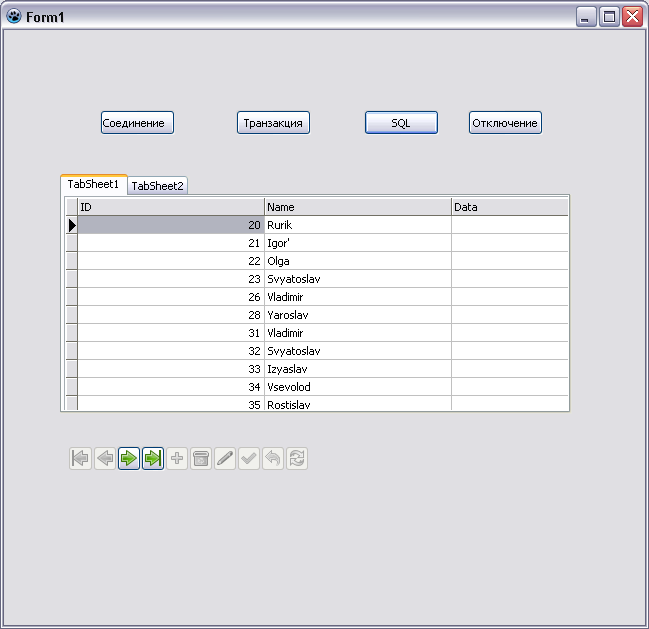
Edit3.Text := DBGrid2.Columns.Items[3].Field.Value;

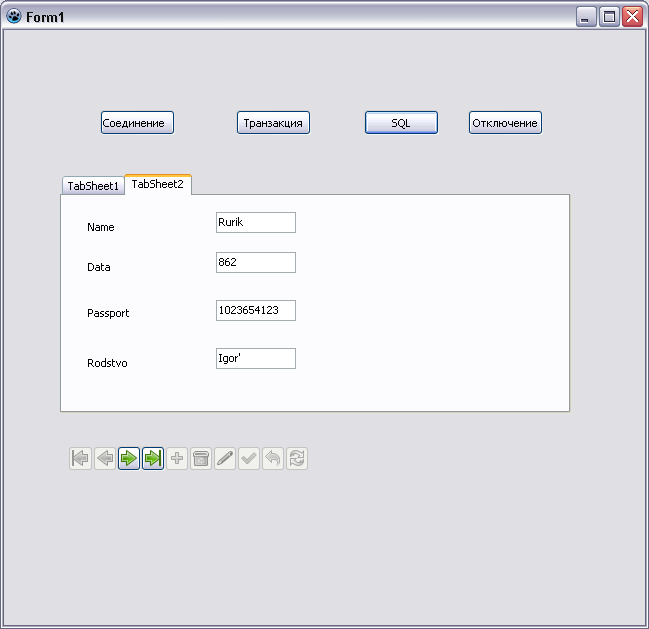
Edit4.Text := DBGrid2.Columns.Items[4].Field.Value;

end;

end.

# §6 Тестирование программы





Для тестирования были проведены все функции программы:

1. При нажатии на кнопку «Соединение», устанавливалась связь с базой данных.
2. При нажатии на кнопку «SQL», выводится таблица базы данных.
3. При нажатии на вторую вкладку выводится информация о конкретном человеке: имя, паспортные данные, дата рождения, степень родства.

# §7 Пояснительная записка

Введение

Название программы: Генеалогическое дерево.

Назначение и область применения: Программа «Генеалогическое дерево» применяется в сфере поиска определенного члена семьи в различных науках.

Например, если рассматривать медицину, усовершенствуя программу, будет возможно вычислить частоту патологий в одной семьи.

Если рассматривать биологические науки, то возможно наблюдать за эволюцией и т.д.