Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Пояснительная записка к курсовой работе по дисциплине «Программирование»

Тема: Тест ПДД

.

Выполнил: Проверил:

Студент гр. 853502 Воробьев А. А.

Меньшиков В. В.

Минск 2018

Содержание

[**ВВЕДЕНИЕ** 2](#_Toc9885795)

[**1. ПОСТАНОВКА ЗАДАЧИ** 4](#_Toc9885796)

[**2. ВЫБОР ИНСТРУМЕНТАРИЯ** 5](#_Toc9885797)

[**3. РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА** 6](#_Toc9885798)

[**3.1 Описание программы** 6](#_Toc9885799)

[**3.2 Организация хранения вопросов и ответов** 7](#_Toc9885800)

[**3.3 Реализация функций для формирования списка вопросов и ответов** 7](#_Toc9885801)

[**3.4 Реализация функции для отображения вопроса** 9](#_Toc9885802)

[**4. ДЕМОНСТРАЦИЯ РАБОТЫ** 10](#_Toc9885803)

[**4.1 Запуск программы** 10](#_Toc9885804)

[**4.2 Тест** 11](#_Toc9885805)

[**4.3 Результат теста** 12](#_Toc9885806)

[**ЗАКЛЮЧЕНИЕ** 13](#_Toc9885807)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 14](#_Toc9885808)

[**ПРИЛОЖЕНИЕ** 15](#_Toc9885809)

# **ВВЕДЕНИЕ**

В настоящий момент в мире почти у каждого человека есть автомобиль. Со временем ПДД (Правила Дорожного Движения) забываются водителями. Из-за чего происходит много аварий, в которых люди получают ранения либо вообще погибают. Ведь от этих знаний зависит не только его безопасность, но и безопасность окружающих. Для предотвращения такой ситуации необходимо не забывать Правила Дорожного Движения. Также возможно - разъяснить и научить водителя пользоваться своими правами, отстаивать свои интересы.

Многие из водителей, недавно сдавших экзамен в ГИБДД, еще помнят большое эмоциональное напряжение и чувство страха перед самим экзаменом. Большой эмоциональный стресс и ожидание наихудшего варианта событий - спутники учеников автошкол перед экзаменами.

Для облегчения жизни перед экзаменами в ГИБДД и полного усвоения пройденного в автошколе материала рекомендуется пройти тест ПДД. Данные тесты дают возможность проверить свои знания в области правил дорожного движения (ПДД) и выявить все пробелы в знаниях. Тест повторяет вопросы, которые утверждены в качестве экзаменационных для проверки знаний ПДД в ГИБДД.

Тест ПДД будет полезен не только ученикам автошкол, но и опытным водителям, которые сдавали экзамен очень давно и некоторые моменты уже забыли. Данное тестирование сможет указать на ошибки водителя и предостеречь от их выполнения в будущем. Водитель должен помнить, что его ждут неоплаченные штрафы ГИБДД в случае нарушения ПДД во время управления транспортным средством, поэтому лучше заранее подготовиться и проверить свой уровень знаний для предотвращения подобных нарушений.

Популярность тестов ПДД объясняется несколькими причинами:

1. возможность не только бесплатного многократного тестирования, но и обучения здесь же, без приобретения специальной литературы, которая стоит в настоящее время достаточно дорого;

2. бесплатность и доступность к тестам.

3. высокая скорость прохождения тестов и получение оценки результата.

Решая билеты ГИБДД, вы тестируете свои знания ПДД, что принесет вам пользу:

тест ответами на вопросы билетов и получением оценки не заканчивается - вы можете просмотреть сделанные ошибки, проанализировать их и определиться с тем, на что при изучении Правил дорожного движения необходимо потратить дополнительное время.

Также будь то рядовой автолюбитель, автоинструктор или инспектор дорожного движения (ГИБДД или ГАИ), все действуют на основании разрешений или запретов, прописанных в ПДД. Правильное понимание и соблюдение дорожных знаков и основных положений Правил дорожного движения принять правильное решение в критической или непредвиденной ситуации на дороге. Нужно помнить, что знание ПДД никогда не помешает в жизни. А на автомобильных улицах единственным законом являются Правила дорожного движения. Для чего и будет служить моя программа.

# **1. ПОСТАНОВКА ЗАДАЧИ**

Цель: создание теста ПДД. Программа должна содержать в себе несколько вариантов билетов, которые используются для прохождения экзамена в ГИБДД.

Данный проект реализует следующие цели:

- При прохождении теста можно допустить не более одной ошибки для успешного прохождения экзамена.

- Не отвлекающий и простой интерфейс.

- Сравнение данных пользователем ответов с верным вариантом ответа, в ходе тестирования.

-Возможность просмотра правильных ответов.

# **2. ВЫБОР ИНСТРУМЕНТАРИЯ**

C**++** Builder — программный продукт, инструмент быстрой разработки приложений(RAD), интегрированная среда программирования (IDE), система, используемая программистами для разработки программного обеспечения на языке программирования C++.

C++ Builder объединяет в себе комплекс объектных библиотек (STL, VCL, CLX, MFC и др.), компилятор, отладчик, редактор кода и многие другие компоненты. Цикл разработки аналогичен Delphi. Большинство компонентов, разработанных в Delphi, можно использовать и в C++ Builder без модификации, но обратное утверждение неверно.

C++ Builder содержит инструменты, которые при помощи drag-and-drop действительно делают разработку визуальной, упрощает программирование благодаря встроенному WYSIWYG — редактору интерфейса и пр.

Библиоте́ка визуа́льных компоне́нтов (англ. Visual Component Library, VCL) — объектно-ориентированная библиотека для разработки программного обеспечения, разработанная компанией Borland (на данный момент поддерживается Embarcadero) для поддержки принципов визуального программирования. VCL входит в комплект поставки Delphi, C++ Builder и Embarcadero RAD Studio и является, по сути, частью среды разработки, хотя разработка приложений в этих средах возможна и без использования VCL. VCL предоставляет огромное количество готовых к использованию компонентов для работы в самых разных областях программирования, таких, например, как интерфейс пользователя (экранные формы и элементы управления — т. н. «контролы», «контроли»), работа с базами данных, взаимодействие с операционной системой, программирование сетевых приложений и прочее.

# **3. РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА**

## **3.1 Описание программы**

При каждом запуске теста случайным образом выбирается 10 вопросов из фиксированного набора вопросов.

Каждый вопрос содержит текстовую и графическую информацию.

При ответе на каждый вопрос пользователю предлагается выбрать один верный ответ из 4х вариантов.

При прохождении теста можно допустить не более одной ошибки для успешного прохождения теста.

Сравнение данных пользователем ответов с верным вариантом ответа после тестирования.

## **3.2 Организация хранения вопросов и ответов**

Для хранения вопроса и вариантов ответов к нему используется класс, состоящий из 4 полей:

1) Поле “question” типа UnicodeString, содержащее текст вопроса.

2) Массив “answers” типа UnicodeString, состоящий из 4 элементов, в котором записываются 4 варианта ответа к вопросу.

3) Поле “image” типа UnicodeString, содержащее имя изображения для вопроса.

4) Поле “correct” типа int, в котором храниться номер правильного ответа, находящегося в массиве ответов “answers”.

class Question

{

UnicodeString question;

UnicodeString answers[4];

UnicodeString image;

int correct;

};

Листинг 1. Класс для хранения вопроса

## **3.3 Реализация функций для формирования списка вопросов и ответов**

В этой функции формируется список вопросов и ответов. Вопросы хранятся в виде массива классов. Имена изображений, вопросы и ответы хранятся в текстовом файле. Из файла они записываются в объект класса TStringList. Затем в цикле имена изображений и ответы записываются в поля классов.

void List::load()

{

TStringList \*stringList = new TStringList;

Question tempQuestion;

stringList->LoadFromFile("Questions.txt");

int i = 0;

while (i < stringList->Count)

{

tempQuestion.SetQuestion(stringList->Strings[i]);

i++;

tempQuestion.SetImage(stringList->Strings[i]);

i++;

for (int j = 0; j < 4; j++, i++)

{

tempQuestion.SetAnswer(stringList->Strings[i], j);

}

tempQuestion.SetCorrect(stringList->Strings[i]);

i++;

push(tempQuestion);

}

}

Листинг 2. Функция для формирования списка вопросов

Далее случайно выбирается 10 различных вопросов из общего списка, которые будут предлагаться для решения.

void List::fill(List list)

{

int k;

int arr[20];

bool iscopy = 0;

randomize();

for (int i = 0; i < 10; i++)

{

k = rand() % 100;

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

{

if (k == arr[j])

{

i--;

iscopy = 1;

break;

}

}

if (iscopy != 1)

{

arr[i] = k;

push(list[k]);

}

iscopy = 0;

}

}

Листинг 3. Функция для выбора 10 вопросов

## **3.4 Реализация функции для отображения вопроса**

Данная функция предназначена для отображения текущего вопроса на экране.

void Question::Show()

{

Form2->Image->Picture->Bitmap->FreeImage();

Form2->Label3->Caption = "";

Form2->Label4->Caption = "";

Form2->RadioButton3->Visible = false;

Form2->RadioButton4->Visible = false;

Form2->RadioButton1->Checked=false;

Form2->RadioButton2->Checked=false;

Form2->RadioButton3->Checked=false;

Form2->RadioButton4->Checked=false;

Form2->SetColor(clBlack);

if(image != "none")

{

Form2->Image->Picture->LoadFromFile(image);

Form2->SetLeft(463);

}

else

{

Form2->SetLeft(8);

}

Form2->LabelQuestion->Caption = question;

Form2->Label1->Caption = answers[0];

Form2->Label2->Caption = answers[1];

if(answers[2] != "none")

{

Form2->RadioButton3->Visible = true;

Form2->Label3->Caption = answers[2];

}

if(answers[3] != "none")

{

Form2->RadioButton4->Visible = true;

Form2->Label4->Caption = answers[3];

}

}

Листинг 4. Функция для отображения вопроса

# **4. ДЕМОНСТРАЦИЯ РАБОТЫ**

## **4.1 Запуск программы**

При запуске программы открывается главное меню (Рисунок 1). В нем предлагается начать прохождение теста или выйти из программы.

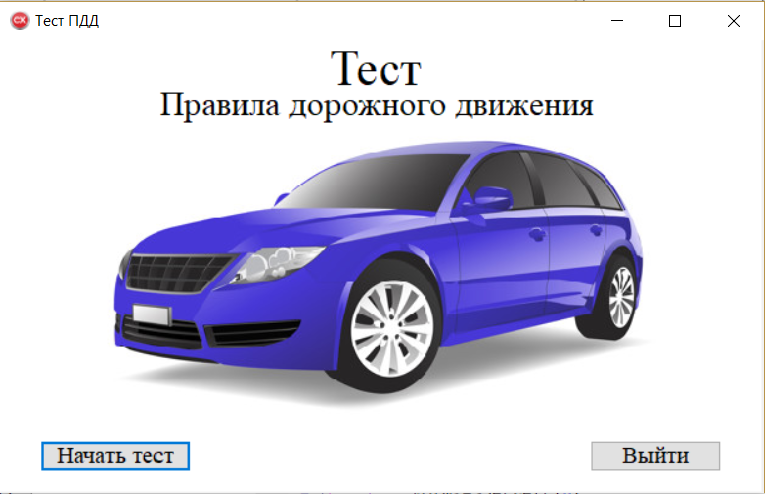


Рисунок 1. Главное окно программы.

## 

## **4.2 Тест**

После нажатия на кнопку “Начать Тест” происходит переход к Тестовому окну (Рисунок 2).

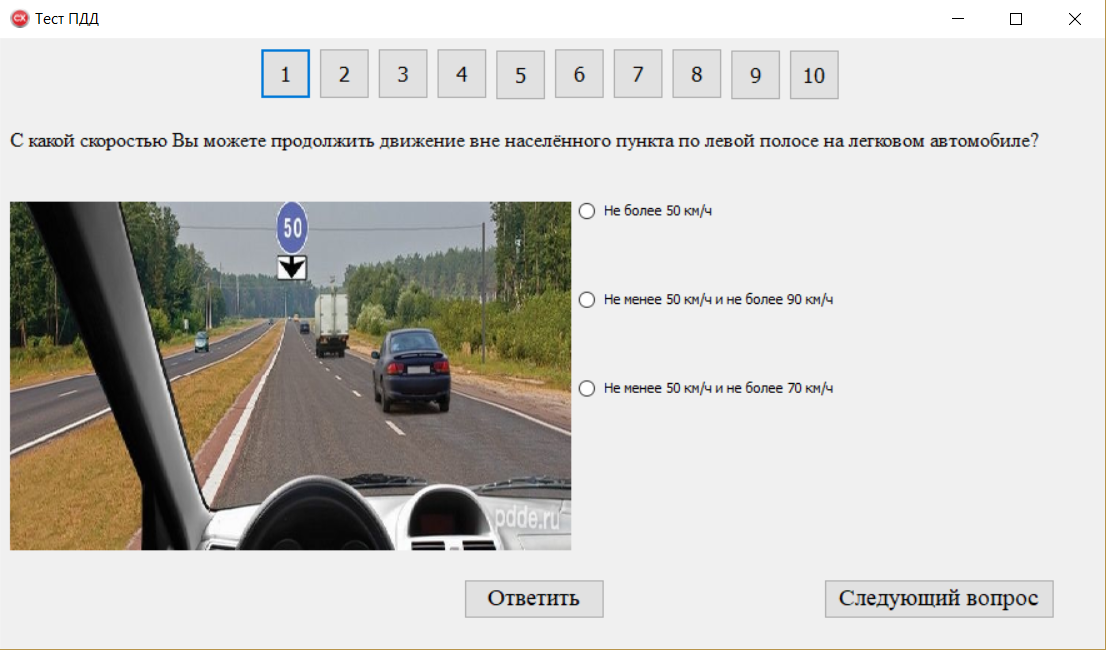


Рисунок 2. Тестовое окно.

В Тестовом окне сверху по центру находятся кнопки для выбора вопроса. Сверху выводится вопрос, слева изображения для него. Справа предлагается 4-ре варианта ответов, один из которых правильный. Выбор ответа осуществляется нажатием на элемент “RadioButton”. Для подтверждения ответа и перехода к следующему вопросу надо нажать на кнопку “Ответить”. Можно пропустить вопрос и перейти к следующему с помощью кнопки “Следующий вопрос”. Когда пользователь ответил на все вопросы и готов завершить тест, нужно нажать на появившуюся кнопку “Завершить тест”.

## **4.3 Результат теста**

После нажатия кнопки “Завершить тест” в “Тестовом окне”, откроется окно с результатом теста (Рисунок 3). В нём будет выведено сообщение “Вы сдали тест”, если пользователь допустил не более одной ошибок, или “Вы не сдали тест”, если пользователь допустил более одной ошибок.

Из этого окна можно вернуться в “Главное окно” при нажатии на кнопку “Главное меню”. Или выйти из программы, нажав на кнопку “Выйти”. А также просмотреть правильные варианты ответа, нажав на кнопку “Результаты” (Рисунок 4).

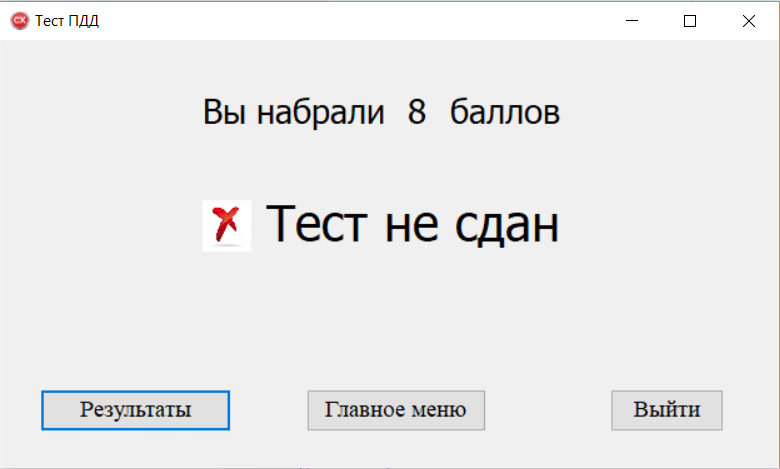


Рисунок 3. Окно с результатом теста.

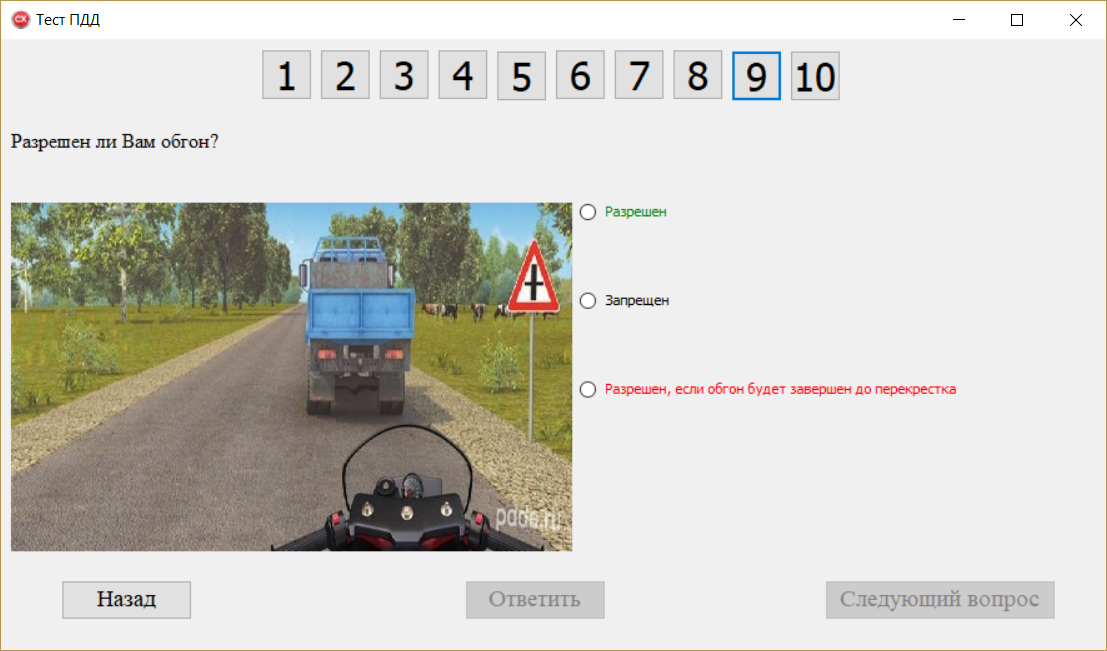


Рисунок 4. Окно с правильными вариантами ответа.

# **ЗАКЛЮЧЕНИЕ**

В результате работы над курсовой работой была разработана программа, которая имеет удобный и интуитивно понятный пользовательский интерфейс, тестирование пользователя по десяти вопросам указанной темы, просмотр результатов тестирования.

Характерными плюсами данной программы являются приятное оформления, простота навигации по программе. Программа не требует больших ресурсов компьютера для полноценной функциональности. Также положительной чертой программы является и то, что она может работать как под операционной системой Windows XP и выше.

В качестве языка разработки был выбран язык программирования высокого уровня С++. В качестве среды разработки была выбрана C++ Builder 10.2.3.

Приложение состоит из нескольких форм:

Стартовая форма - является стартовым меню программы.

Тест форма - основная форма; на ней проводится тестирование пользователя.

Форма отображения результатов тестирования - конечная форма программы, на которой пользователю предоставляется возможность просмотра результатов.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ЖВАКИНА, А. В. Разработка Windows-приложений в среде визуального программирования: Электронное учебное издание (Пособие). / А. В. Жвакина. – Минск: ВА РБ, 2016. –342с.

2. ПДД.BY [Электронный ресурс] – ПДД.BY: <http://pdd.by/>.

3. pdde.ru [Электронный ресурс] – pdde.ru: <https://pdde.ru/>.

4. H-L-L.RU [Электронный ресурс] – Программа для тестирования (тест): <http://h-l-l.ru/publ/28-1-0-114>.

# **ПРИЛОЖЕНИЕ**

**MainMenu.cpp**

//---------------------------------------------------------------------------

#include <vcl.h>

#include <jpeg.hpp>

#include "MainMenu.h"

#include "Test.h"

#include "Results.h"

#pragma hdrstop

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

list.load();

ImageBG->Picture->LoadFromFile("Background.jpg");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonExitClick(TObject \*Sender)

{

Form1->Close();

Form2->Close();

Form3->Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonStartClick(TObject \*Sender)

{

Form1->Visible = false;

Form2->Visible = true;

Form2->ButtonNext->Enabled = true;

Form2->ButtonAnswer->Enabled = true;

Form2->ButtonBack->Visible = false;

Form2->ButtonNext->Caption = "Следующий вопрос";

for (int i = 0; i < 10; i++)

{

Form2->SetSize(i, 12);

}

Form2->SetColor(clBlack);

Form2->isComplete = false;

for (int i = 0; i < 10; i++)

{

Form2->answers[i] = 0;

}

listTest.clear();

listTest.fill(Form1->list);

Form2->number = 0;

listTest[Form2->number].Show();

}

//---------------------------------------------------------------------------

**Test.cpp**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Test.h"

#include "MainMenu.h"

#include "Results.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

//---------------------------------------------------------------------------

void TForm2::SetSize(int number, int size)

{

switch(number)

{

case 0:

Button1->Font->Size = size;

break;

case 1:

Button2->Font->Size = size;

break;

case 2:

Button3->Font->Size = size;

break;

case 3:

Button4->Font->Size = size;

break;

case 4:

Button5->Font->Size = size;

break;

case 5:

Button6->Font->Size = size;

break;

case 6:

Button7->Font->Size = size;

break;

case 7:

Button8->Font->Size = size;

break;

case 8:

Button9->Font->Size = size;

break;

case 9:

Button10->Font->Size = size;

break;

}

}

//---------------------------------------------------------------------------

void TForm2::SetColor(TColor color)

{

Form2->Label1->Font->Color = color;

Form2->Label2->Font->Color = color;

Form2->Label3->Font->Color = color;

Form2->Label4->Font->Color = color;

}

//---------------------------------------------------------------------------

void TForm2::SetColor(int i)

{

switch(Form2->answers[i])

{

case 1:

Form2->Label1->Font->Color = clRed;

break;

case 2:

Form2->Label2->Font->Color = clRed;

break;

case 3:

Form2->Label3->Font->Color = clRed;

break;

case 4:

Form2->Label4->Font->Color = clRed;

break;

}

switch(Form1->listTest[i].GetCorrect())

{

case 1:

Form2->Label1->Font->Color = clGreen;

break;

case 2:

Form2->Label2->Font->Color = clGreen;

break;

case 3:

Form2->Label3->Font->Color = clGreen;

break;

case 4:

Form2->Label4->Font->Color = clGreen;

break;

}

}

//---------------------------------------------------------------------------

void TForm2::SetLeft(int left)

{

Form2->RadioButton1->Left = left;

Form2->RadioButton2->Left = left;

Form2->RadioButton3->Left = left;

Form2->RadioButton4->Left = left;

Form2->Label1->Left = left + 21;

Form2->Label2->Left = left + 21;

Form2->Label3->Left = left + 21;

Form2->Label4->Left = left + 21;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::ButtonNextClick(TObject \*Sender)

{

number++;

if (number == 9)

ButtonNext->Caption = "Закончить тест";

else

ButtonNext->Caption = "Следующий вопрос";

if (number == 10)

{

Form2->Visible = false;

Form3->Visible = true;

isComplete = true;

int correctCounter = 0;

for (int i = 0; i < 10; i++)

{

if (answers[i] == Form1->listTest[i].GetCorrect())

correctCounter++;

}

Form3->LabelScore->Caption = IntToStr(correctCounter);

if (correctCounter > 8)

{

Form3->LabelPass->Caption = "Тест сдан";

Form3->Image->Picture->LoadFromFile("yes.jpg");

}

else

{

Form3->LabelPass->Caption = "Тест не сдан";

Form3->Image->Picture->LoadFromFile("no.jpg");

}

number = 0;

}

else

Form1->listTest[number].Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button1Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 0;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button2Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 1;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button3Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 2;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button4Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 3;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button5Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 4;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button6Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 5;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button7Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 6;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button8Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 7;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button9Click(TObject \*Sender)

{

ButtonNext->Caption = "Следующий вопрос";

number = 8;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button10Click(TObject \*Sender)

{

ButtonNext->Caption = "Закончить тест";

number = 9;

Form1->listTest[number].Show();

if (isComplete == true)

SetColor(number);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::ButtonAnswerClick(TObject \*Sender)

{

int answer = 0;

if(RadioButton1->Checked==true)

answer = 1;

if(RadioButton2->Checked==true)

answer = 2;

if(RadioButton3->Checked==true)

answer = 3;

if(RadioButton4->Checked==true)

answer = 4;

if (answer != 0)

{

SetSize(number, 24);

answers[number] = answer;

if (number < 9)

ButtonNextClick(Sender);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Label1Click(TObject \*Sender)

{

RadioButton1->Checked = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Label2Click(TObject \*Sender)

{

RadioButton2->Checked = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Label3Click(TObject \*Sender)

{

if (RadioButton3->Visible == true)

RadioButton3->Checked = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Label4Click(TObject \*Sender)

{

if(RadioButton4->Visible == true)

RadioButton4->Checked = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::ButtonBackClick(TObject \*Sender)

{

Form2->Visible = false;

Form3->Visible = true;

}

//---------------------------------------------------------------------------

**Results.cpp**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Test.h"

#include "MainMenu.h"

#include "Results.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm3 \*Form3;

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::ButtonExitClick(TObject \*Sender)

{

Form1->Close();

Form2->Close();

Form3->Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::ButtonMenuClick(TObject \*Sender)

{

Form3->Visible = false;

Form1->Visible = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::ButtonResultClick(TObject \*Sender)

{

Form3->Visible = false;

Form2->Visible = true;

Form2->ButtonNext->Enabled = false;

Form2->ButtonAnswer->Enabled = false;

Form2->ButtonBack->Visible = true;

Form2->SetColor(9);

}

//---------------------------------------------------------------------------