Министерство образования и науки, молодежи и спорта Украины

Донбасская государственная машиностроительная академия

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных и самостоятельных работ

(для студентов направления «Системный анализ»)

Утверждено
на заседании кафедры ИСПР
Протокол № 1 от 31 августа 2011г.

Краматорск 2011

УДК 681.3

Объектно-ориентированное программирование : Методические указания к выполнению лабораторных и самостоятельных работ для студентов направления «Компьютерные науки» специальности «Интеллектуальные системы принятия решений» всех форм обучения / Сост. А. Ю. Мельников. – Краматорск: ДГМА, 2011. – 48 с.

Содержат задания к работам по составлению диаграмм программ в среде Lazarus

Составитель Мельников А.Ю., канд. техн. наук, доцент

Отв. за выпуск Белевцов Л. В., докт. физ.-мат. наук, доцент

5 ЗАДАНИЯ К ЛАБОРАТОРНЫМ И САМОСТОЯТЕЛЬНЫМ РАБОТАМ

5.1 Лабораторная работа 1. Компоненты среды визуального программирования Lazarus

Цель работы. Изучить основные страницы палитры компонент среды визуального программирования Lazarus.

Задание к работе. Создать простейшее приложение в среде визуального программирования Lazarus. Форма должна содержать три взаимосвязанных компоненты различных страниц палитры компонент: две выбираются из таблицы 3 согласно номеру варианта, третью студент подбирает самостоятельно исходя из личных предпочтений. Для каждой компоненты необходимо написать как минимум по одному методу, которым эта компонента реагирует на выбранное событие, при этом хотя бы один метод должен изменять состояние другой компоненты. В отчете по работе перечислить список свойств каждой компоненты, которые подвергались изменению в процессе работы приложения.

Таблица 3 – Список компонент

Вар.	Компонента 1	Компонента 2
1	2	3
1	MainMenu	Memo
2	PopupMenu	ToggleBox
3	Button	CheckBox
4	Label	RadioButton
5	Edit	ListBox
6	Memo	Shape
7	ToggleBox	NoteBook
8	CheckBox	LabeledEdit
9	RadioButton	Button
10	ListBox	StatusBar
11	ComboBox	BitBtn
12	ScrollBar	Label
13	RadioGroup	SpeedButton
14	CheckGroup	Image
15	BitBtn	Edit
16	SpeedButton	ComboBox

Продолжение таблицы 3

1	2	3
17	Image	ScrollBar
18	Shape	ScrollBox
19	NoteBook	CheckListBox
20	LabeledEdit	RadioGroup
21	CheckListBox	CheckGroup
22	ScrollBox	ProgressBar
23	StringGrid	PopupMenu
24	TrackBar	SpinEdit
25	ProgressBar	Timer
26	StatusBar	MainMenu
27	SpinEdit	ColorBox
28	PageControl	StringGrid
29	ColorBox	PageControl
30	Timer	TrackBar

Пример выполнения работы

Задание: использовать компоненты Shape, Timer и ProgressBar.

Возможное решение: приложение будет представлять собой компьютерную игру, в процессе которой игроку нужно кликнуть мышью по случайно возникающим на экране кругам (компонента Shape) с фиксированным контролем времени (компонента Timer) и фиксированным (максимально возможным) числом кругов. За перемещение изображений по экрану отвечает другая компонента Timer, ход времени и число (долю) «пойманных» кругов показывают компоненты ProgressBar. Игра заканчивается в случае истечения отведенного на нее времени либо «поимки» игроком заданного числа кругов.

Используемые свойства компонент

Компонента	Свойства, определяемые сразу	Свойства, изменяемые в процессе работы
Shape1	Shape (stCircle), Width (40), Height (40)	Top, Left
Timer1	Interval (1000)	
Timer2	Interval (1000)	
ProgressBar1	Width (по ширине формы)	Min, Max, Position
ProgressBar2	Width (по ширине формы)	Min, Max, Position

Тест модуля приложения приведен далее, вид формы и работающего приложения – на рис. 92-94.

```
unit Unit1;
{$mode objfpc}{$H+}
interface
uses
  Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs, Menus, StdCtrls, Buttons, ExtCtrls, ComCtrls;
type
  { TForm1 }
  TForm1 = class(TForm)
    ProgressBar1: TProgressBar;
    ProgressBar2: TProgressBar;
    Shape1: TShape;
    Timer1: TTimer;
    Timer2: TTimer;
    procedure FormActivate(Sender: TObject);
     procedure Shape1MouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure Timer1Timer(Sender: TObject);
    procedure Timer2Timer(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;

var
  Form1: TForm1;

implementation
{ TForm1 }

var k:integer; // число пойманных кругов
```

```
procedure TForm1.FormActivate(Sender: TObject);
// процедура установки начальных значений
begin
    with ProgressBar1 do begin
    Min:=0;
    Max:=60; // 1 минута на игру
    Position:=0 end;
    with ProgressBar2 do begin
    Min:=0;
    Max:=5; // нужно поймать 5 кругов
    Position:=0 end;
    k:=0;
    randomize; // включаем генератор случайных чисел
    with Shape1 do begin
    Top:=random(Height-50); // размещаем первый круг
    Left:=random(Width)
    end;
end;

procedure TForm1.Shape1MouseDown(Sender: TObject; Button:
TMouseButton;  Shift: TShiftState; X, Y: Integer);
// щелчок по кругу
begin
    k:=k+1;
    with ProgressBar2 do begin
    Position:=k;
    if Position=Max then begin // конец игры
    ShowMessage('Поздравляем, вы выиграли!');
    Close
    end
    end;
    Shape1.Top:=random(Height-50-Shape1.Height);
    Shape1.Left:=random(Width-Shape1.Width)
end;
```

```
procedure TForm1.Timer1Timer(Sender: TObject);
// контроль времени
begin
    with ProgressBar1 do begin
    Position:=Position+1; // секунда прошла
    if Position=Max then begin // конец игры
    ShowMessage('Время истекло!');  Close
    end
    end
end;

procedure TForm1.Timer2Timer(Sender: TObject);
// перемещение круга
var nach:integer;
begin
    nach:=random(4); // случайное число 0..3
    case nach of
    0: begin Shape1.Left:=Shape1.Left+10; // идем направо
      if Shape1.Left+Shape1.Width >= Width then Shape1.Left:=1
      end;
    1: begin Shape1.Top:=Shape1.Top+10; // идем вниз
      if Shape1.Top+Shape1.Height >= Height-50 // ProgressBar
      then Shape1.Top:=1
      end;
    2: begin Shape1.Left:=Shape1.Left-10; // идем налево
      if Shape1.Left <= 10 then Shape1.Left:=Width-Shape1.Width-1
      end;
    3: begin Shape1.Top:=Shape1.Top-10; // идем вверх
      if Shape1.Top <= 10 then Shape1.Top:=Height-50-Shape1.Height
      end;
    end
end;
initialization
  {$I unit1.lrs}
end.
```

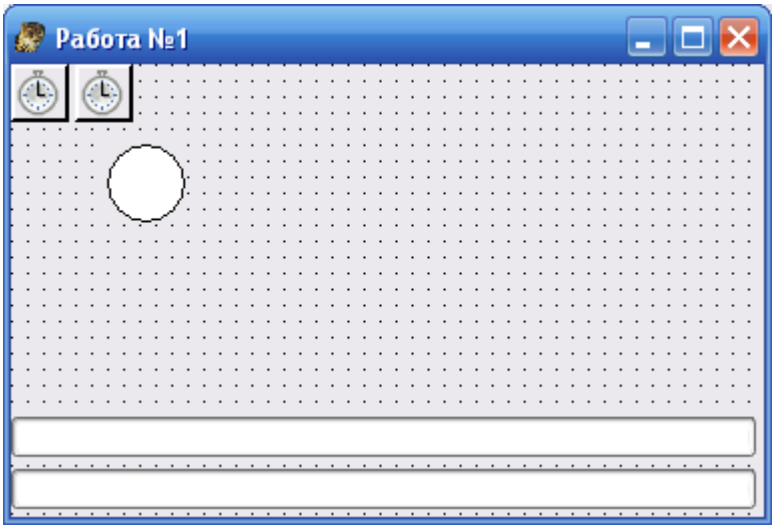


Рисунок 92 – Вид приложения на этапе дизайна

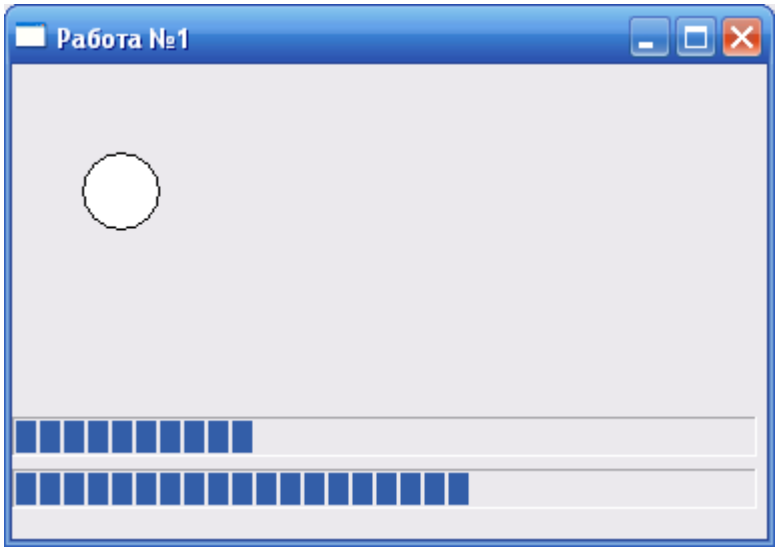
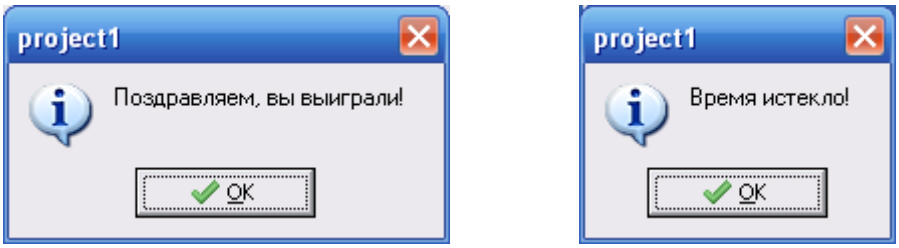


Рисунок 93 – Вид приложения в момент работы



(а) *(б)*

Рисунок 94 – Сообщения об окончании игры: успешном (а) и по времени (б)

Выводы. В ходе лабораторной работы были изучены компоненты Shape, Timer и ProgressBar, их основные свойства и предназначение; было разработано игровое приложение, в котором использовались только рассмотренные компоненты.

5.2 Лабораторная работа 2. Разработка простого приложения для Windows с помощью средства разработки Lazarus

Цель работы. Получить навыки использования различных компонент среды визуального программирования Lazarus для решения поставленной задачи.

Задание к работе. Создать приложение согласно индивидуальному заданию (табл. 4). Приложение должны содержать базовый набор управляющих элементов (главное меню, строка состояния и т.п.)

Таблица 4 – Варианты заданий

Вар.	Содержание задачи
1	**2**
1	Разработать приложение, осуществляющее перевод температуры в градусах по Цельсию в температуру по Фаренгейту или Кельвину по следующим зависимостям: Фаренгейт = 32 + (Цельсий / 5) * 9 Кельвин = 273,15 + Цельсий
2	Разработать приложение, осуществляющее ввод 6 значений и нахождение либо среднего арифметического, либо среднего геометрического значения при нажатии командной кнопки, а также выдачу результата
3	Разработать приложение, осуществляющее ввод 5 значений и нахождение математического ожидания, или среднеквадратического отклонения, или дисперсии при нажатии командной кнопки, а также выдачу результата
4	Разработать приложение, осуществляющее ввод 7 значений и нахождение минимального или максимального при нажатии кнопки, а также выдачу результата
5	Разработать приложение, осуществляющее ввод значений матрицы 3x3 (при помощи окон ввода текста, расположенных в виде матрицы) и вычисление ее детерминантов
6	Разработать приложение, осуществляющее решение квадратного уравнения по введенным пользователем коэффициентам. Окна ввода коэффициентов обязательно должны быть дополнены текстовыми пояснениями
7	Разработать тестирующую программу, которая предлагает пользователю два вопроса и по три ответа на каждый из них (только один правильный), а также выставляет оценку по результатам тестирования

Продолжение таблицы 4

1	2
8	Разработать приложение, в котором цвет формы (свойство Color) изменялся бы при каждом третьем нажатии соответствующей кнопки, при этом имелась бы возможность этот режим обработки нажатия кнопки отключать
9	Разработать приложение с окном, в котором будут расположены все компоненты страницы «Стандартные». При наведении на каждую из компонент указателя мыши должна появляться подсказка (свойство Hint при установленном свойстве ShowHint:=True) с описанием возможностей компонента. При этом должна быть предусмотрена возможность переключения режима показа подсказок (краткие / полные)
10	Разработать приложение с окном, в котором будут расположены все компоненты страницы «Дополнительные». Работа приложения должна осуществляться подобно предыдущему варианту
11	Разработать приложение с окном, в котором будут расположены все компоненты страницы «Dialogs». Работа приложения должна осуществляться подобно предыдущему варианту
12	Разработать приложение – электронный калькулятор для операций +, -, *, / над вещественными числами
13	Разработать приложение, осуществляющее возведение в степень числа. При этом пользователь должен иметь возможность ввести число, возводимое в степень, и значение степени, а программа должна осуществлять проверку вводимых символов и перевод отрицательного значения степени в положительное
14	Разработать приложение, в котором цвет формы изменяется при помощи трех полос прокрутки (компоненты ScrollBar) для каждого базового цвета – красного, зеленого и синего. Полученные с помощью полос прокрутки числовые значения должны быть преобразованы в значение цвета (при помощи функции RGB), которое присваивается свойству Color
15	Разработать приложение, которое заполняет случайными числами в случайном порядке матрицу (4x4) полей ввода текста и очищает их путем нажатия соответствующей кнопки
16	Разработать «игровой автомат», который генерирует случайным образом выигрышное число и сравнивает его с введенным пользователем числом; по результатам сравнения автомат должен назначить размер выигрыша и сообщить об этом пользователю
17	Разработать приложение – шифровальщик текстов, которое по заданному вами закону либо переставляет местами символы введенного пользователем текста, либо заменяет одни символы в исходном тексте на другие, и отображает зашифрованный вариант пользовательского текста

Продолжение таблицы 4

1	2
18	Разработать приложение – эмулятор устройства связи с возможностью набора телефонного номера с проверкой формата вводимых данных и указанием пользователю ошибок при вводе
19	Разработать справочную систему, выдающую в окно вывода текста информацию о четырех операторах языка программирования Паскаль. Выбор отображаемой информации осуществить при помощи зависимых или независимых переключателей
20	Разработать справочную систему, выдающую в окно вывода текста информацию о четырех командах интерфейса средства разработки Lazarus. Выбор отображаемой информации осуществить при помощи зависимых или независимых переключателей
21	Разработать приложение – игру под названием «Сапер», которая потребует от пользователя включения массива независимых переключателей таким образом, чтобы не включить один или несколько переключателей, символизирующих мины. Информация о близости мины при включении очередного переключателя должна выдаваться пользователю
22	Разработать приложение – электронный калькулятор, использующий тригонометрические функции
23	Разработать приложение для игры в крестики-нолики двумя игроками. Для этого расставьте CheckBox в виде матрицы 3x3 для каждого игрока и предоставьте возможность выбора текущего игрока при помощи зависимого переключателя. Программа должна выдать результат игры
24	Разработать приложение – игру «Морской бой» (упрощенная версия)
25	Разработать приложение для построения гистограмм (столбчатых диаграмм) при помощи компоненты Chart. Данные должны вводиться из текстового файла и отображаться при помощи компоненты StringGrid
26	Разработать приложение для построения гистограмм (столбчатых диаграмм) при помощи методов вывода прямоугольников на канву. Данные должны вводиться пользователем при помощи компоненты StringGrid
27	Разработать приложение для построения графика заданной функции при помощи компоненты Chart. Данные должны вводиться из текстового файла и отображаться при помощи компоненты StringGrid

Продолжение таблицы 4

1	2
28	Разработать приложение для построения графика заданной функции при помощи методов рисования линий на канве. Данные должны вводиться пользователем при помощи компоненты StringGrid
29	Разработать приложение, обеспеченное максимально возможным числом компонент для выбора свойств формы (например, компонента ColorBox позволит изменять цвет формы, в поле ввода текста можно будет ввести заголовок формы и т.д.)
30	Разработать приложение, осуществляющее поиск экстремальных значений заданной функции на заданном интервале одним из известных методов оптимизации. Предусмотреть вывод всей промежуточной информации

Пример выполнения работы

Задание: оснастить игру, разработанную в ходе предыдущей работы, нормальным пользовательским интерфейсом.

Описание решения: переместим игровое поле на отдельную панель, добавим главное меню, статусную строку и кнопки для управления, обеспечим возможность изменения параметров и их сохранения в текстовом файле, а также сохранения результатов игры на диске, установим контроль над возможностью изменения размеров окна.

Текст программы приведен ниже, экранные формы на этапе дизайна и работы приложения – на рис. 95-103.

```
unit Unit1;
{$mode objfpc}{$H+}
interface
uses
  Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs,
  Menus, StdCtrls, Buttons, ExtCtrls, ComCtrls, Grids;
type
  { TForm1 }
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    MenuItem4: TMenuItem;
```

```
MenuItem1: TMenuItem;
MenuItem2: TMenuItem;
MenuItem3: TMenuItem;
MenuItem5: TMenuItem;
MenuItem6: TMenuItem;
MenuItem7: TMenuItem;
Notebook1: TNotebook;
Page1: TPage;
Page2: TPage;
Page3: TPage;
Panel1: TPanel;
ProgressBar1: TProgressBar;
ProgressBar2: TProgressBar;
Shape1: TShape;
StatusBar1: TStatusBar;
StringGrid1: TStringGrid;
StringGrid2: TStringGrid;
Timer1: TTimer;
Timer2: TTimer;
ToolBar1: TToolBar;
ToolButton1: TToolButton;
ToolButton2: TToolButton;
ToolButton3: TToolButton;
ToolButton4: TToolButton;
procedure FormActivate(Sender: TObject);
procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);
procedure FormCreate(Sender: TObject);
procedure FormResize(Sender: TObject);
procedure MenuItem1Click(Sender: TObject);
procedure MenuItem2Click(Sender: TObject);
procedure MenuItem3Click(Sender: TObject);
procedure MenuItem5Click(Sender: TObject);
procedure MenuItem6Click(Sender: TObject);
procedure MenuItem7Click(Sender: TObject);
```

```
    procedure Shape1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Timer1Timer(Sender: TObject);
    procedure Timer2Timer(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;
var
  Form1: TForm1;

implementation
{ TForm1 }
var k:integer; // число пойманных кругов

procedure TForm1.FormCreate(Sender: TObject);
// установка начальных значений
var S:TStringList;st:string;
    i:integer;
begin
     randomize; // включаем генератор случайных чисел
     with StringGrid1 do begin
     ColWidths[1]:=60;ColWidths[0]:=260;
     Cells[0,0]:='                              Параметр';
     Cells[1,0]:='Значение';
     Cells[0,1]:='Время на одну игру (секунд)';
     Cells[1,1]:='60';
     Cells[0,2]:='Сколько кругов нужно поймать';
     Cells[1,2]:='5';
     Cells[0,3]:='Задержка передвижения круга (миллисекунд)';
     Cells[1,3]:='500';
     Cells[0,4]:='Имя игрока';
     Cells[1,4]:='Admin';
     end;
```

```
    if FileExists('params.dat') then begin
      S:=TStringList.Create;
      S.LoadFromFile('params.dat');
      for i:=1 to 4 do StringGrid1.Cells[1,i]:=S[i-1];
      S.Free
    end;
    with StringGrid2 do begin
    Cells[0,0]:='   Дата и время';
    Cells[1,0]:='         Игрок';
    Cells[2,0]:='         Очки';
    end;
    if FileExists('gamers.dat') then begin
      S:=TStringList.Create;
      S.LoadFromFile('gamers.dat');
      StringGrid2.RowCount:=S.Count+1;
      for i:=1 to S.Count do begin
        st:=S[i-1];
        StringGrid2.Cells[0,i]:=copy(st,1,pos('//',st)-1);
        delete(st,1,pos('//',st)+1);
        StringGrid2.Cells[1,i]:=copy(st,1,pos('//',st)-1);
        delete(st,1,pos('//',st)+1);
        StringGrid2.Cells[2,i]:=st
        end;
      S.Free
    end;
    StatusBar1.Panels[0].Text:='Приложение загружено'
end;

procedure TForm1.FormActivate(Sender: TObject);
// процедура установки начальных значений
begin
    with ProgressBar1 do begin
    Min:=0;
    Max:=StrToInt(StringGrid1.Cells[1,1]);
    Position:=0 end;
```

```
    with ProgressBar2 do begin
    Min:=0;
    Max:=StrToInt(StringGrid1.Cells[1,2]);
    Position:=0 end;
    Timer2.Interval:=StrToInt(StringGrid1.Cells[1,3]);
    k:=0;
    with Shape1 do begin
    Top:=random(Panel1.Height-Height); // размещаем первый круг
    Left:=random(Panel1.Width-Width)
    end;
end;

procedure TForm1.FormClose(Sender: TObject; var CloseAction: TCloseAction);
// сохраняем параметры
var S:TStringList;
   i,j:integer;
begin
    S:=TStringList.Create;
    S.Clear;
    for i:=1 to 4 do S.Add(StringGrid1.Cells[1,i]);
    S.SaveToFile('params.dat');
    S.Clear;
    for i:=1 to StringGrid2.RowCount-1 do
    S.Add(StringGrid2.Cells[0,i]+'//'+
    StringGrid2.Cells[1,i]+'//'+StringGrid2.Cells[2,i]);
    if S.Count > 0 then S.SaveToFile('gamers.dat');
    S.Free
end;

procedure TForm1.FormResize(Sender: TObject);
// изменение размеров формы
begin
    Notebook1.Left:=0;
    Notebook1.Top:=ToolBar1.Height+1;
```

```
    Notebook1.Width:=Width-1;
    Notebook1.Height:=Height-Notebook1.Top-StatusBar1.Height-21;
    ProgressBar2.Top:=Notebook1.Height-ProgressBar2.Height-26;
    ProgressBar1.Top:=ProgressBar2.Top-ProgressBar1.Height-2;
    ProgressBar2.Left:=1;ProgressBar2.Width:=Notebook1.Width-2;
    ProgressBar1.Left:=1;ProgressBar1.Width:=Notebook1.Width-2;
    Panel1.Left:=1;
    Panel1.Width:=Notebook1.Width-2;
    Panel1.Top:=1;
    Panel1.Height:=ProgressBar1.Top-Panel1.Top-1;
    StringGrid2.Left:=1;
    StringGrid2.Top:=1;
    StringGrid2.Height:=Notebook1.Height-22;
end;

procedure TForm1.MenuItem1Click(Sender: TObject);
// начать игру
begin
    FormActivate(Sender);
    Notebook1.PageIndex:=0;
    Timer1.Enabled:=True;
    Timer2.Enabled:=True;
    StatusBar1.Panels[0].Text:='Игра'
end;

procedure TForm1.MenuItem2Click(Sender: TObject);
// прервать игру
begin
    Timer1.Enabled:=False;
    Timer2.Enabled:=False;
    ProgressBar1.Position:=0;
    ProgressBar2.Position:=0;
    StatusBar1.Panels[0].Text:='Игра прервана';
    ShowMessage('Игра прервана по инициативе игрока!')
end;
```

```
procedure TForm1.MenuItem3Click(Sender: TObject);
// завершить работу приложения
begin
    Close
end;

procedure TForm1.MenuItem5Click(Sender: TObject);
// о программе
begin
    ShowMessage(
    'Игра "попади по мячу"'+#13#13+
    'Автор: студент группы ИС-15-5 Иванов Петр Сергеевич')
end;

procedure TForm1.MenuItem6Click(Sender: TObject);
// изменить параметры
begin
    Notebook1.PageIndex:=1;
end;

procedure TForm1.MenuItem7Click(Sender: TObject);
// список игроков
begin
    Notebook1.PageIndex:=2;
end;

procedure  TForm1.Shape1MouseDown(Sender:  TObject;  Button:  TMouseBut-
ton;
  Shift: TShiftState; X, Y: Integer);
// щелчок по кругу
begin
    if StatusBar1.Panels[0].Text <> 'Игра' then exit;
    k:=k+1;
    with ProgressBar2 do begin
    Position:=k;
```

```
    if Position=Max then begin // конец игры
    Timer1.Enabled:=False;
    Timer2.Enabled:=False;
    with StringGrid2 do begin
        RowCount:=RowCount+1;
        Cells[0,RowCount-1]:=DateTimeToStr(Now);
        Cells[1,RowCount-1]:=StringGrid1.Cells[1,4];
        Cells[2,RowCount-1]:=IntToStr(k)+' circles / '+
        IntToStr(ProgressBar1.Position)+' seconds'
    end;
    ShowMessage('Поздравляем, вы выиграли!');
    StatusBar1.Panels[0].Text:='Игра завершена победой игрока';
    Notebook1.PageIndex:=2;
    end
    end;
    Shape1.Top:=random(Panel1.Height-Shape1.Height);
    Shape1.Left:=random(Panel1.Width-Shape1.Width)
end;

procedure TForm1.Timer1Timer(Sender: TObject);
// контроль времени
begin
    with ProgressBar1 do begin
    Position:=Position+1; // секунда прошла
    if Position=Max then begin // конец игры
    Timer1.Enabled:=False;
    Timer2.Enabled:=False;
    ShowMessage('Время истекло!');
    StatusBar1.Panels[0].Text:='Игра завершена по истечению времени';
    end
    end
end;
```

```
procedure TForm1.Timer2Timer(Sender: TObject);
// перемещение круга
var nach:integer;
begin
    nach:=random(4); // случайное число 0..3
    case nach of
    0: begin
      Shape1.Left:=Shape1.Left+10; // идем направо
      if Shape1.Left+Shape1.Width >= Panel1.Width then Shape1.Left:=1
      end;
    1: begin
      Shape1.Top:=Shape1.Top+10; // идем вниз
      if Shape1.Top+Shape1.Height >= Panel1.Height-50 // ProgressBar
      then Shape1.Top:=1
      end;
    2: begin
      Shape1.Left:=Shape1.Left-10; // идем налево
      if Shape1.Left <= 10 then Shape1.Left:=Panel1.Width-Shape1.Width-1
      end;
    3: begin
      Shape1.Top:=Shape1.Top-10; // идем вверх
      if Shape1.Top <= 10 then Shape1.Top:=Panel1.Height-Shape1.Height
      end;
    end
end;

initialization
  {$I unit1.lrs}

end.
```

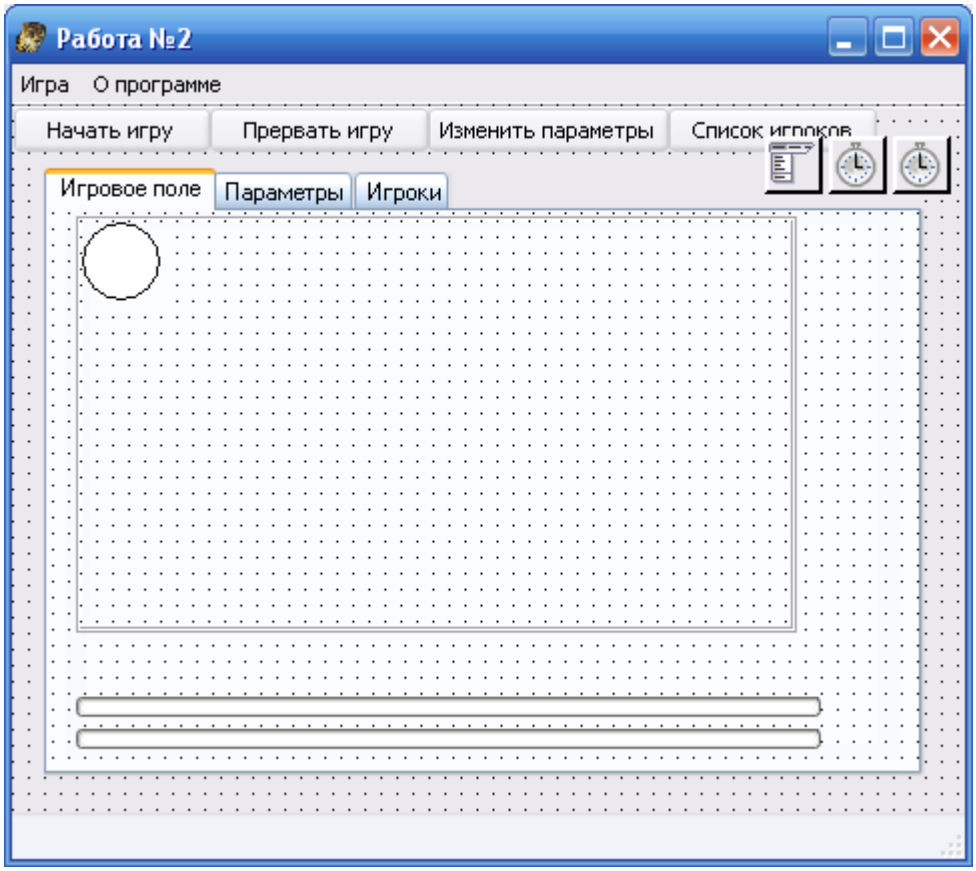


Рисунок 95 – Вид приложения на этапе дизайна («Игровое поле»)

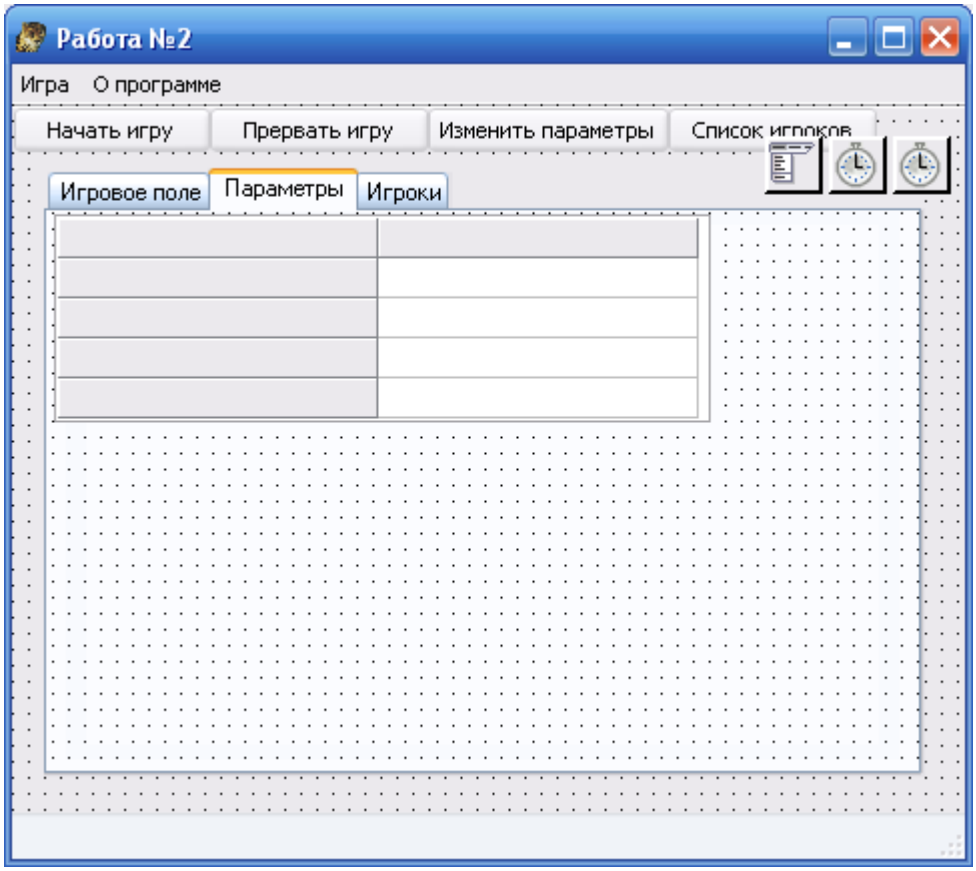


Рисунок 96 – Вид приложения на этапе дизайна («Параметры»)

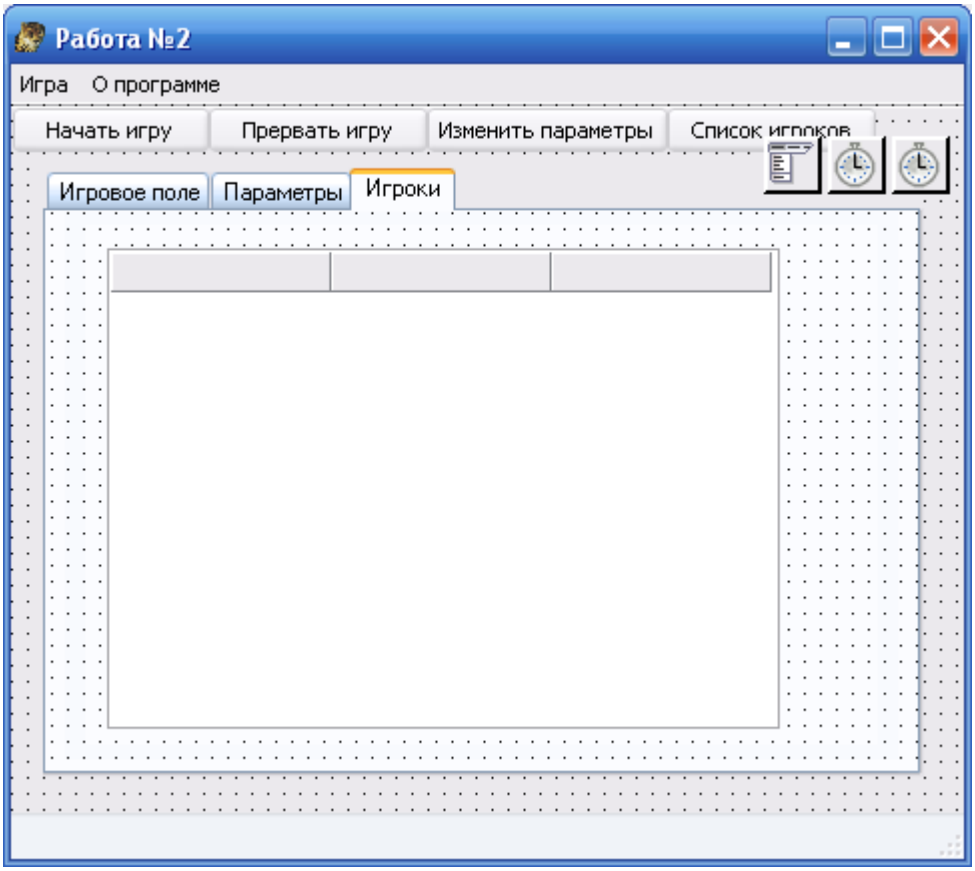


Рисунок 97 – Вид приложения на этапе дизайна («Игроки»)

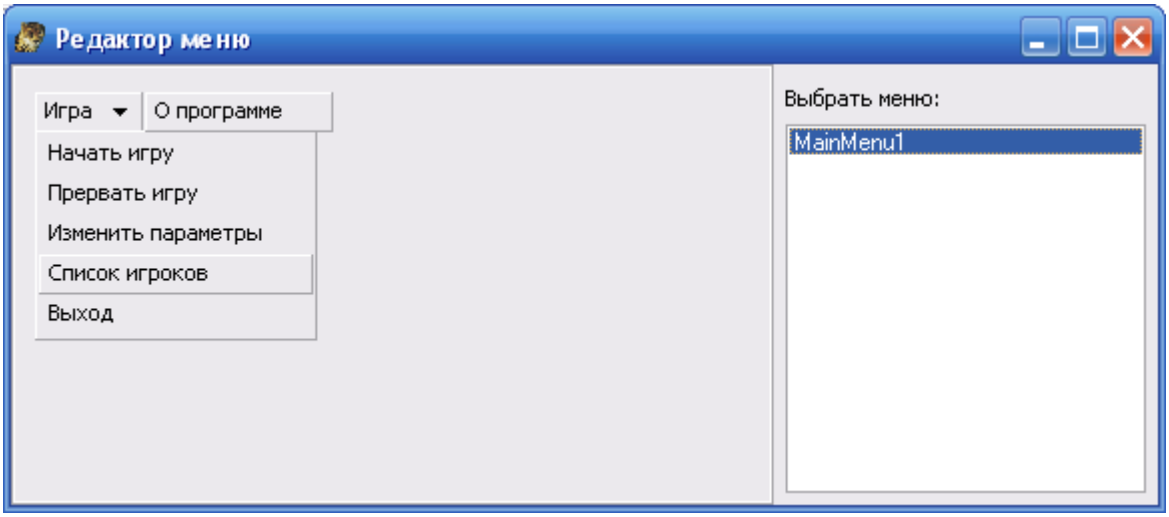


Рисунок 98 – Вид приложения на этапе дизайна (главное меню)

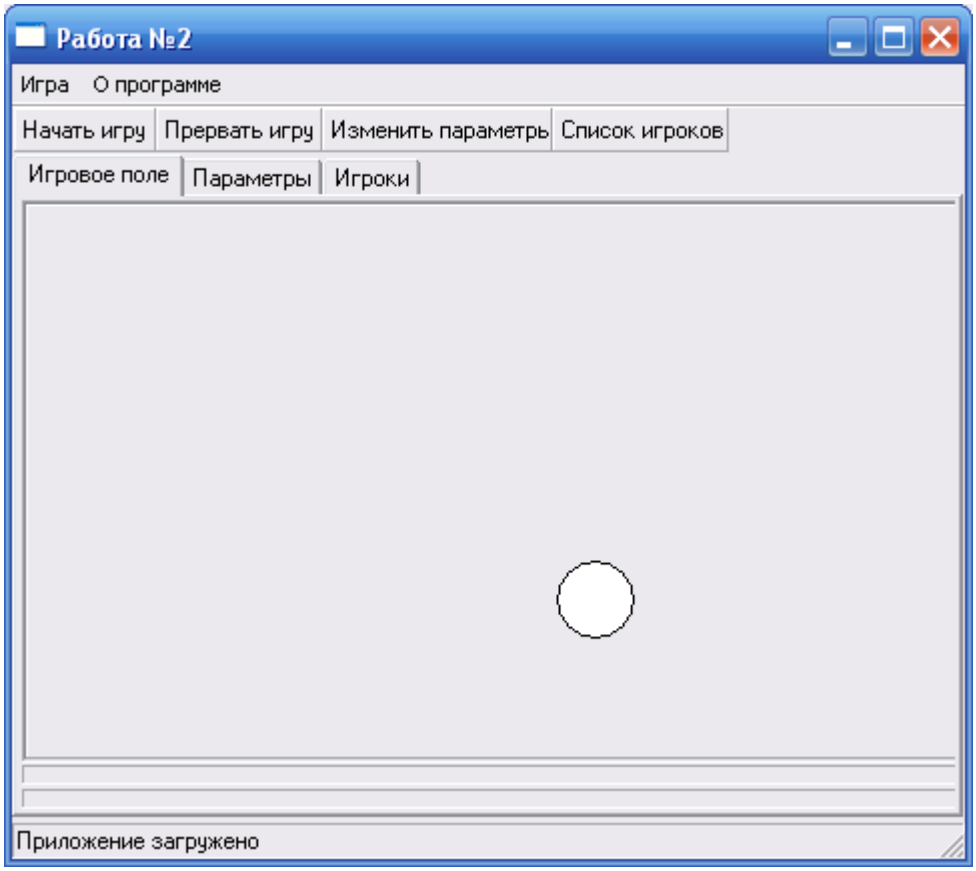


Рисунок 99 – Работа приложения: начало

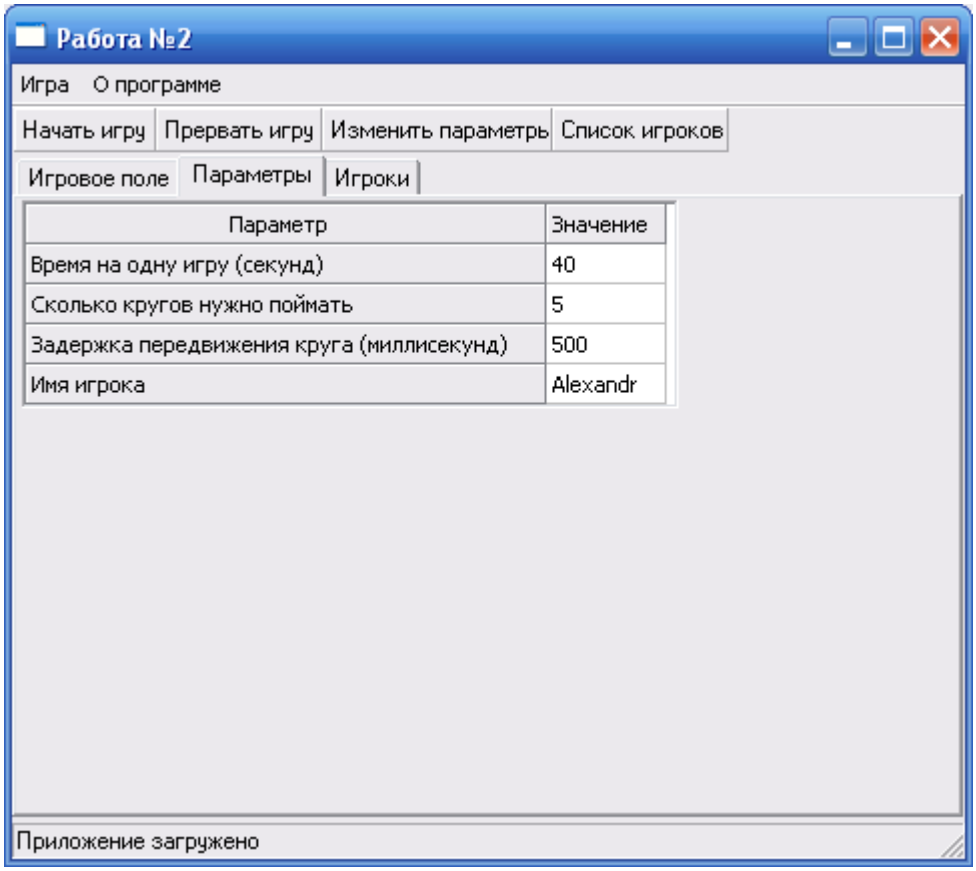


Рисунок 100 – Работа приложения: установка параметров

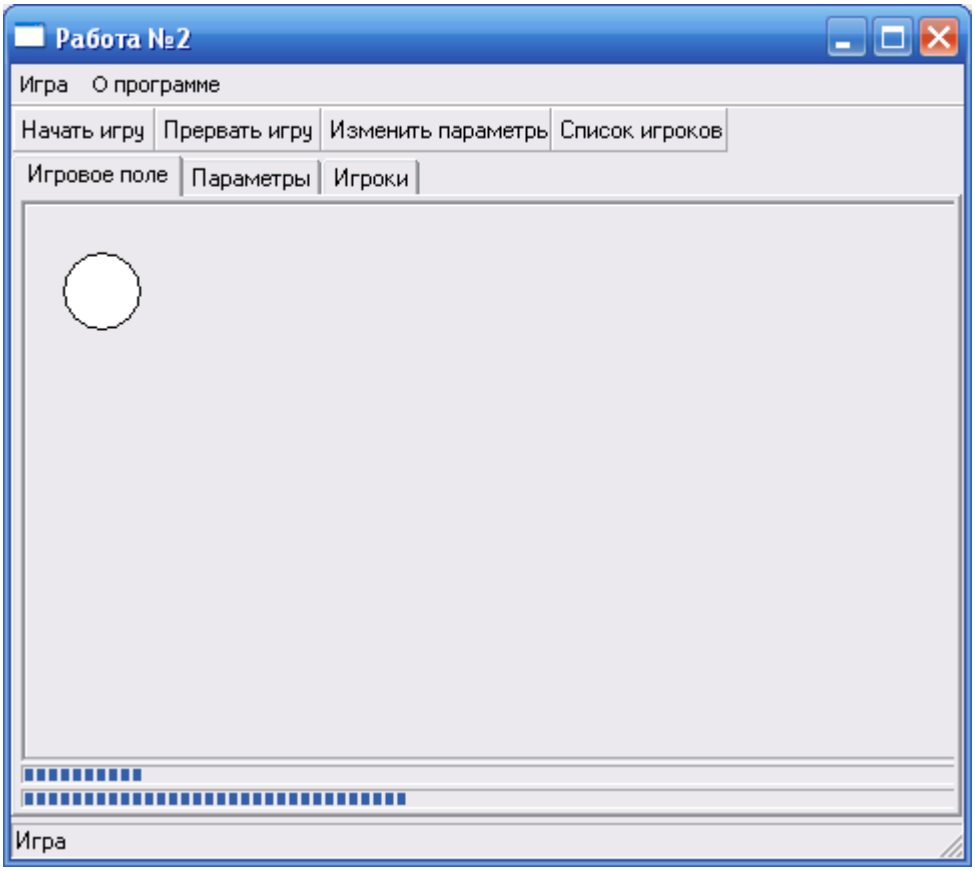


Рисунок 101 – Работа приложения: процесс игры

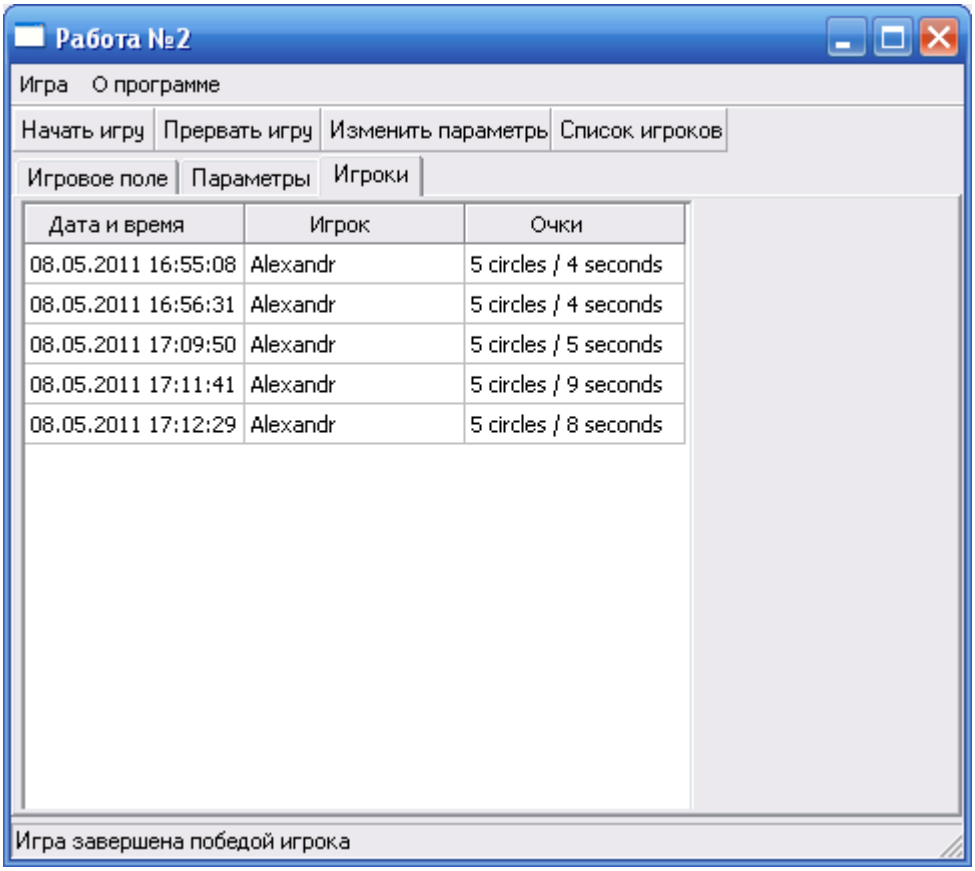


Рисунок 102 – Работа приложения: список игроков и достижений

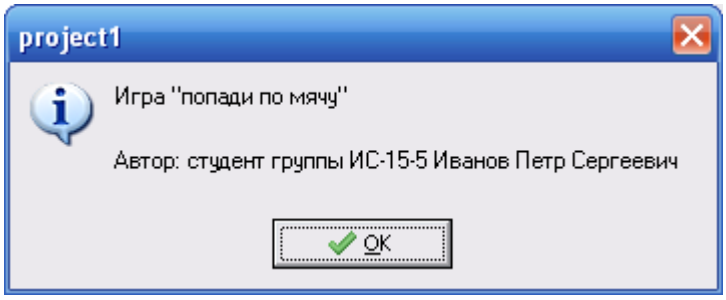


Рисунок 103 – Работа приложения: пункт меню «О программе»

Выводы. В ходе лабораторной работы было создано полнофункциональное приложение для Windows – игра «Попади по мячу»; изучены компоненты MainMenu, Notebook, Panel, ProgressBar, Shape, StatusBar, StringGrid, Timer, ToolBar, их основные свойства и предназначение; созданы методы для обработки событий FormCreate, FormActivate, FormResize, FormClose, MenuItemClick, ShapeMouseDown, TimerTimer. Приложение реализовано в виде одной формы, многостраничный интерфейс обеспечивается компонентой Notebook, состоящей из трех страниц; выбор пользователя осуществляется с помощью главного меню и панельных кнопок.

5.3 Лабораторная работа 3. Построение графиков функций в среде Lazarus

Цель работы. Получить навыки разработки приложений, позволяющих изобразить на экране таблицы и графики функций одной переменной.

Задание к работе. По указанному литературному источнику в читальном зале библиотеки взять данные о математическом описании функций (кривых), выданных в качестве индивидуального задания (табл.5). Разработать приложение, позволяющее рассчитывать таблицу значений своей функции (компонента StringGrid) и строить ее график (компонента Chart). Предусмотреть возможность корректировки пользователем интервала и шага изменения аргумента (вводятся в соответствующих текстовых полях), расчета значения функции при заданном значении аргумента (задавать как явно, так и при помощи «ползунка»), а также вывода результатов в текстовый файл.

Таблица 5 – Варианты заданий

Вар.	**Содержание задания**
1	**2**
1	Линейная функция; математическое описание, требования и ограничения, пример графиков см. на с.113 [11]
2	Квадратичная функция; математическое описание, требования и ограничения, пример графиков см. на с.113 [11]
3	Функция третьей степени; математическое описание, требования и ограничения, пример графиков см. на с.113 [11]
4	Степенная функция; математическое описание, требования и ограничения, пример графиков см. на с.114, 116 параграф 4, 117 [11]
5	Дробно-линейная функция; математическое описание, требования и ограничения, пример графиков см. на с.114 [11]
6	Нелинейная дробно-рациональная функция; математическое описание, требования и ограничения, графики см. на с.115, п.1 [11]
7	Нелинейная дробно-рациональная функция; математическое описание, требования и ограничения, пример графиков см. на с.115, п.2 [11]
8	Нелинейная дробно-рациональная функция; математическое описание, требования и ограничения, пример графиков см. на с.115, п.3 [11]
9	Полукубическая парабола; математическое описание, требования и ограничения, пример графиков см. на с.123 [11]
10	Декартов лист; математическое описание, требования и ограничения, пример графиков см. на с.123 [11]
11	Циссоида; математическое описание, требования и ограничения, пример графиков см. на с.123-124 [11]
12	Строфоида; математическое описание, требования и ограничения, пример графиков см. на с.124 [11]
13	Улитка Паскаля; математическое описание, требования и ограничения, пример графиков см. на с.124 [11]
14	Кардиоида; математическое описание, требования и ограничения, пример графиков см. на с.125 [11]
15	Эпициклоида; математическое описание, требования и ограничения, пример графиков см. на с.126 [11]
16	Показательная функция; математическое описание, требования и ограничения, графики см. на с.119-120 [11]
17	Логарифмическая функция; математическое описание, требования и ограничения, графики см. на с.119-120 [11]
18	Функция гиперболического синуса; математическое описание, требования и ограничения, пример графиков см. на с.121-122 [11]
19	Функция гиперболического косинуса; математическое описание, требования и ограничения, пример графиков см. на с.121-122 [11]

Продолжение таблицы 5

1	2
20	Функция гиперболического тангенса; математическое описание, требования и ограничения, пример графиков см. на с.122 [11]
21	Функция гиперболического котангенса; математическое описание, требования и ограничения, пример графиков см. на с.122 [11]
22	Обыкновенная циклоида; математическое описание, требования и ограничения, пример графиков см. на с.125-126 [11]
23	Архимедова спираль; математическое описание, требования и ограничения, пример графиков см. на с.128 [11]
24	Гиперболическая спираль; математическое описание, требования и ограничения, пример графиков см. на с.128 [11]
25	Укороченная циклоида; математическое описание, требования и ограничения, пример графиков см. на с.126 [11]
26	Удлиненная циклоида; математическое описание, требования и ограничения, пример графиков см. на с.126 [11]
27	Развертка (эвольвента) окружности; математическое описание, требования и ограничения, графики см. на с.128-129 [11]
28	Овалы Кассини; математическое описание, требования и ограничения, пример графиков см. на с.125 [11]
29	Лемниската; математическое описание, требования и ограничения, пример графиков см. на с.125 [11]
30	Трактриса; математическое описание, требования и ограничения, пример графиков см. на с. 129 [11]

Пример выполнения работы

Задание: рассчитать значения и построить график функции логарифмической спирали.

Описание решения: функция логарифмической спирали задается в полярной системе координарт и имеет вид $\rho = A\, e^{k\varphi}$, φ = 0..360°. Если k = 0, то кривая превращается в окружность радиусом A и центром в начале координат. Преобразование значений из полярной системы координат в декартовую осуществляется по формулам:

$$\begin{cases} x = \rho \cos\varphi \\ y = \rho \sin\varphi \end{cases}$$

Интервал и шаг функции не задается, угол изменяется от 0 до 360°.

Текст программы приведен ниже, экранные формы на этапе дизайна и работы приложения – на рис. 104-111.

```
unit Unit1;
{$mode objfpc}{$H+}
interface
uses
  Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs,  Grids, ExtCtrls, TAGraph, StdCtrls, taseries, ComCtrls;
type
  { TForm1 }
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Chart1: TChart;
    Edit1: TLabeledEdit;
    Edit2: TLabeledEdit;
    StringGrid1: TStringGrid;
    StringGrid2: TStringGrid;
    TrackBar1: TTrackBar;
  procedure Button1Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure Func(fi:real;var x,y:real);
  procedure TrackBar1Change(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;
var
  Form1: TForm1;
  S:TSerie;
  a,k:real;
```

```
implementation
procedure TForm1.Func(fi:real;var x,y:real);
var po:real;
begin
    po:=a*exp(k*fi);
    x:=po*cos(fi);
    y:=po*sin(fi);
end;
procedure TForm1.TrackBar1Change(Sender: TObject);
var i:integer;
begin
    i:=TrackBar1.Position;
    StringGrid2.Cells[0,0]:=StringGrid1.Cells[0,i+1];
    StringGrid2.Cells[1,0]:=StringGrid1.Cells[1,i+1];
    StringGrid2.Cells[2,0]:=StringGrid1.Cells[2,i+1];
end;
procedure TForm1.FormCreate(Sender: TObject);
begin
    with StringGrid1 do begin
    RowCount:=362;ColCount:=3;
    Cells[0,0]:='fi';Cells[1,0]:='X';Cells[2,0]:='Y' end;
    S:=TSerie.Create(Chart1);
    S.ShowLines:=True;
    S.ShowPoints:=False;
    S.SeriesColor:=clBlack;
    Chart1.Title.Text.Text:='Логарифмическая спираль';
    Chart1.Title.Visible:=true;
    Chart1.Series.Clear;
    Chart1.AddSerie(S);
    with TrackBar1 do begin
        Width:=Chart1.Width;
        Min:=0;Max:=360;
        Position:=0
    end
end;
```

```
procedure TForm1.Button1Click(Sender: TObject);
// рассчитать значения
var kod,fi:integer;x,y:real;
begin
    val(Edit1.Text,a,kod);
    if (a = 0) or (kod > 0) then
        begin a:=1;Edit1.Text:='1' end;
    val(Edit2.Text,k,kod);
    if kod > 0 then
        begin k:=1;Edit2.Text:='1' end;
    x:=0;y:=0;
    with StringGrid1 do
    for fi:=0 to Rowcount-2 do begin
       func(fi/180*pi,x,y);
       Cells[0,fi+1]:=IntToStr(fi);
       Cells[1,fi+1]:=FloatToStr(x);
       Cells[2,fi+1]:=FloatToStr(y)
       end;
    StringGrid1.SetFocus
end;
procedure TForm1.Button2Click(Sender: TObject);
// построить график
var i:integer;
begin
    S.Clear;
    with StringGrid1 do for i:=1 to Rowcount-1 do
    S.AddXY(StrToFloat(Cells[1,i]),
    StrToFloat(Cells[2,i]),IntToStr(i),clBlack);
    TrackBar1.Width:=Chart1.Width;
    StringGrid1.SetFocus
end;
initialization
  {$I unit1.lrs}
end.
```

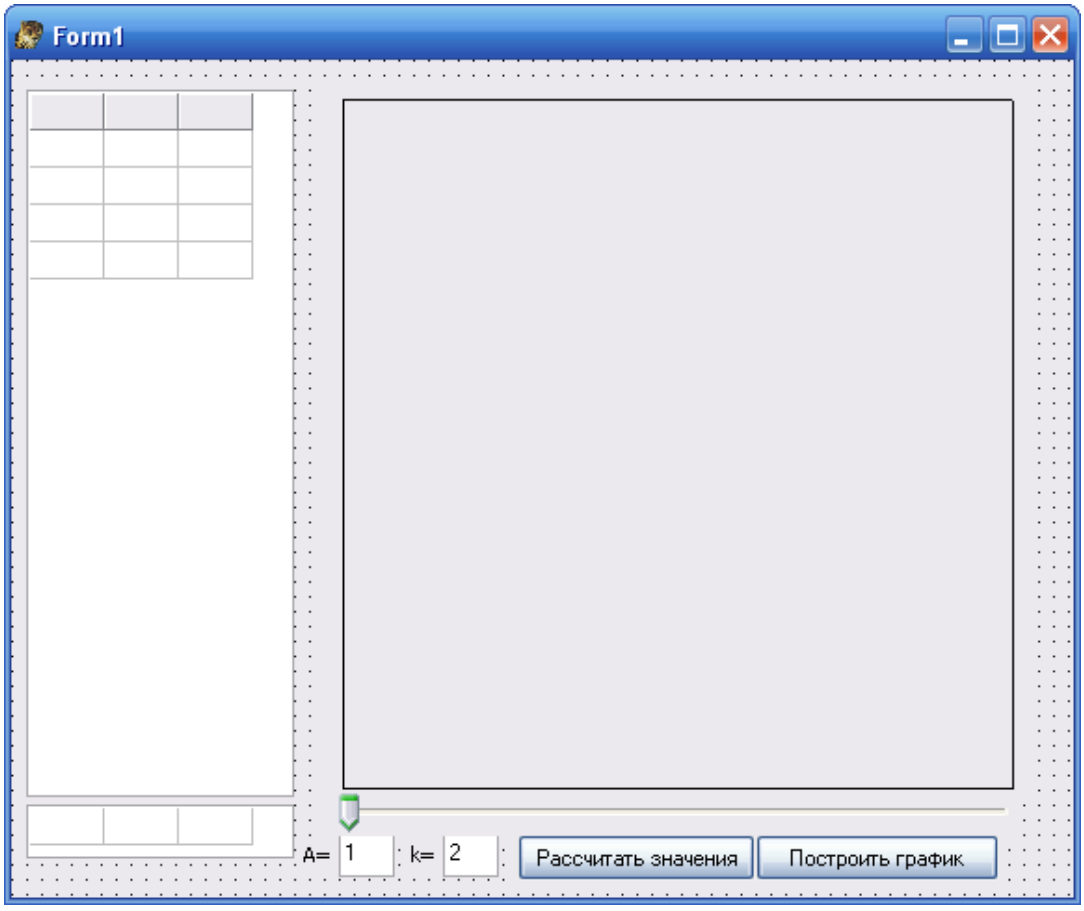


Рисунок 104 – Вид приложения на этапе дизайна

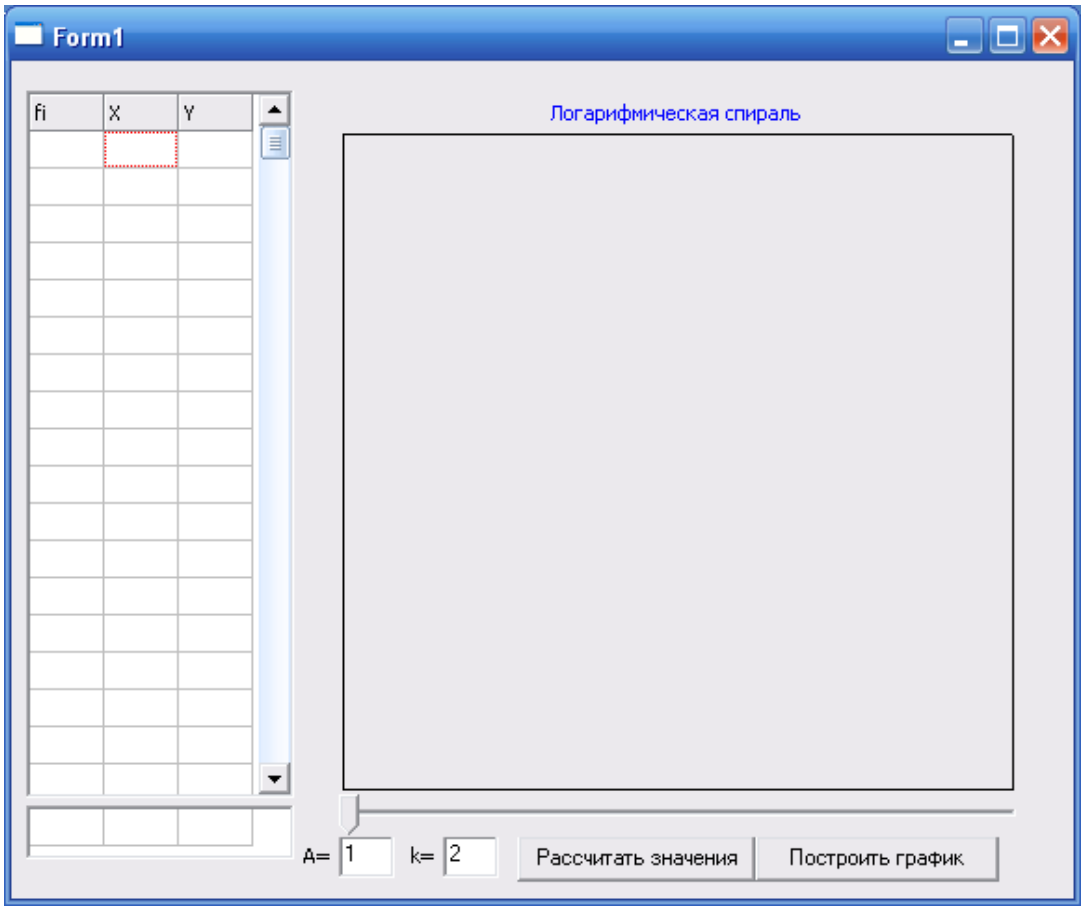


Рисунок 105 – Работа приложения: начало

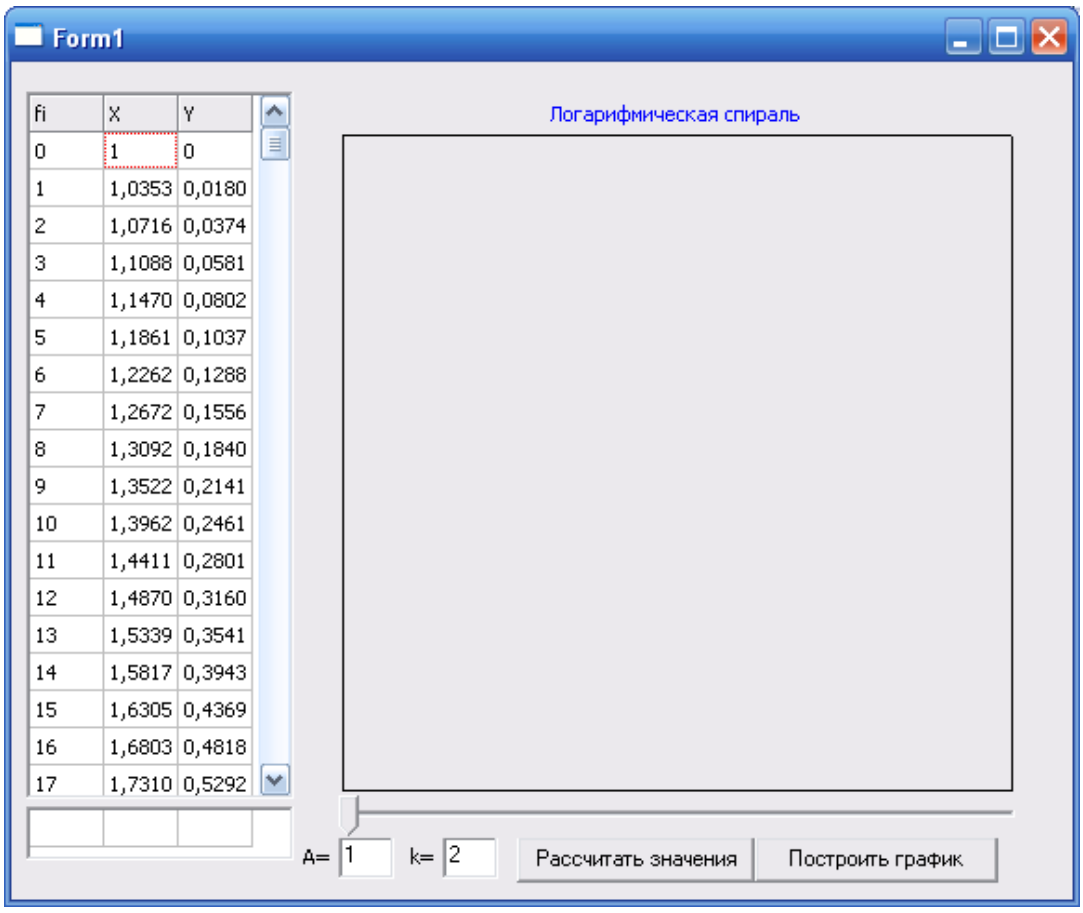


Рисунок 106 – Работа приложения: расчет значений

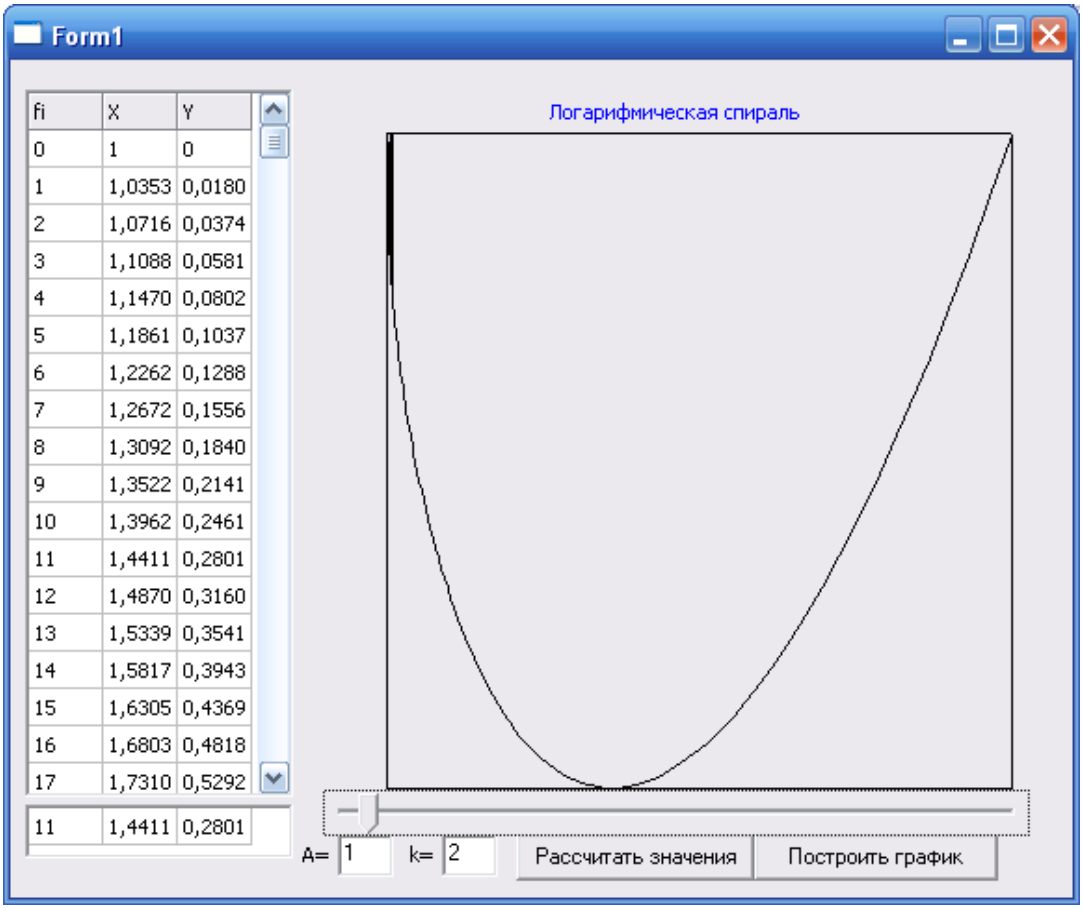


Рисунок 107 – Работа приложения: построение графика для A = 1, k = 2

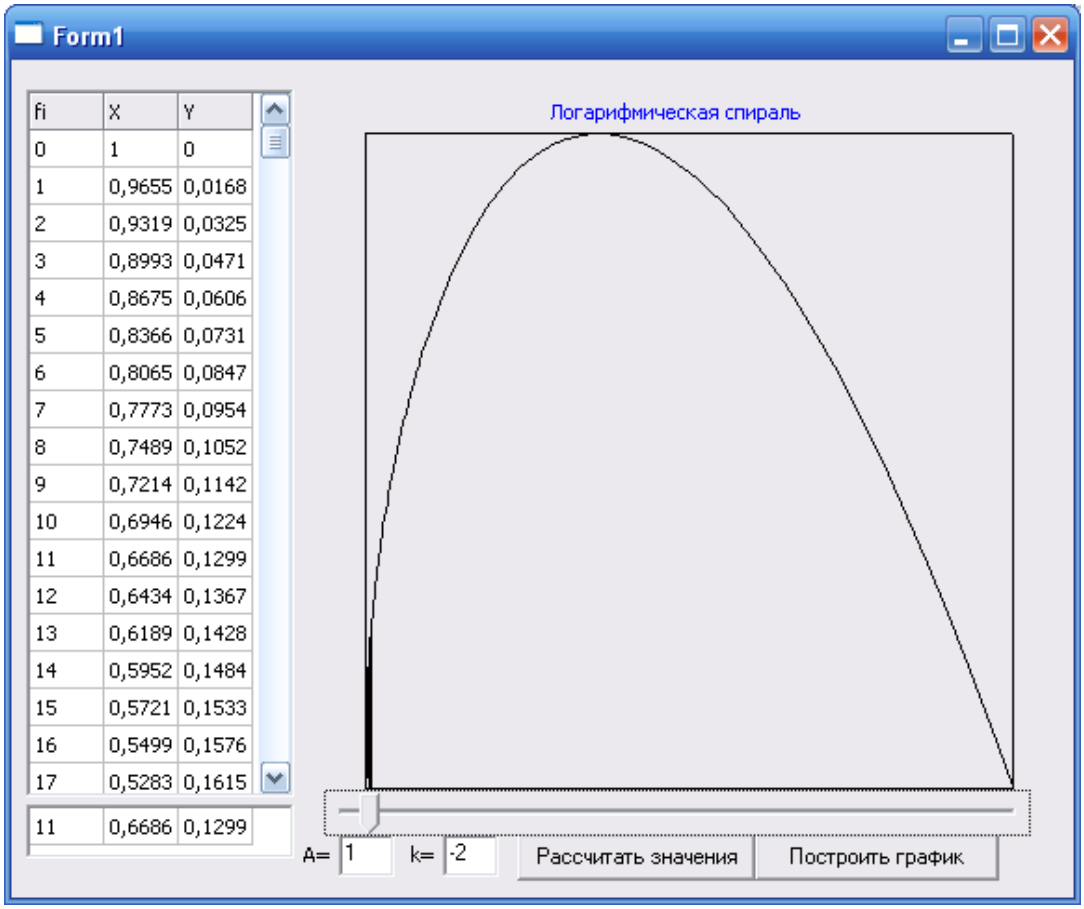


Рисунок 108 – Работа приложения: построение графика для A = 1, k = –2

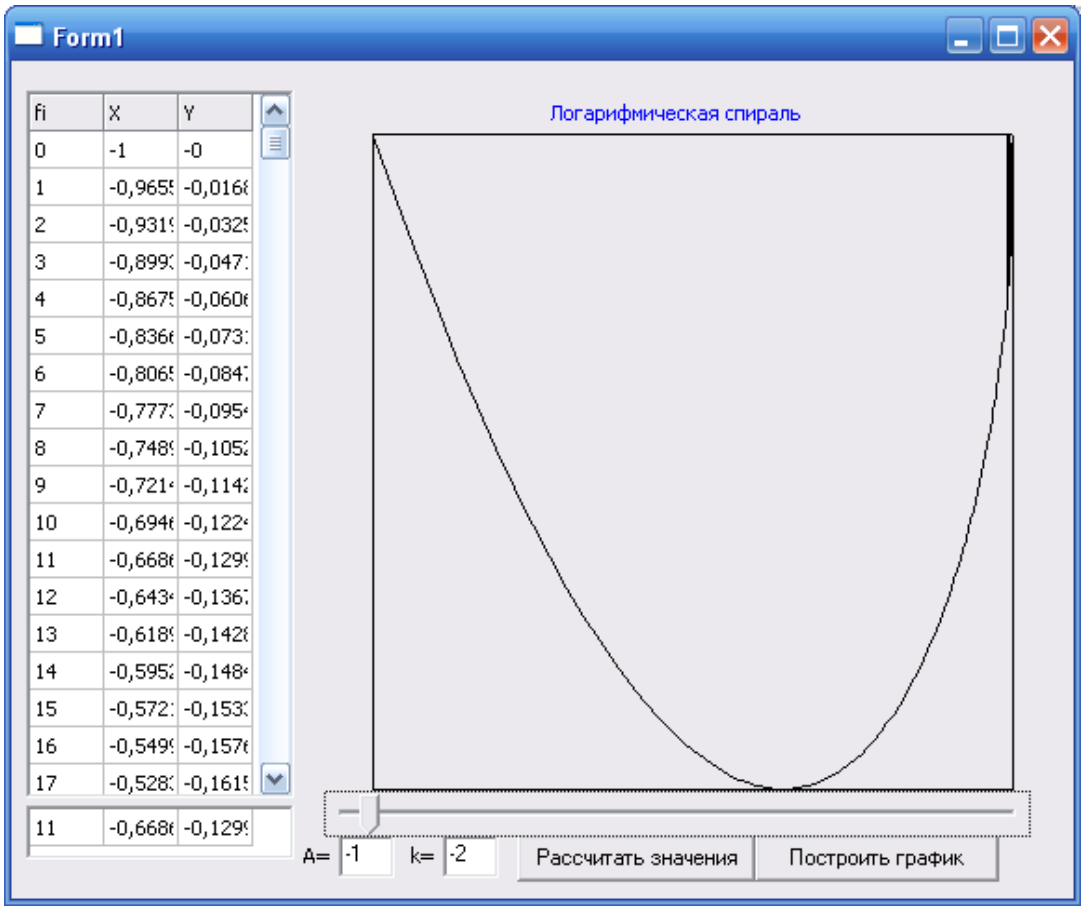


Рисунок 109 – Работа приложения: построение графика для A = -1, k = –2

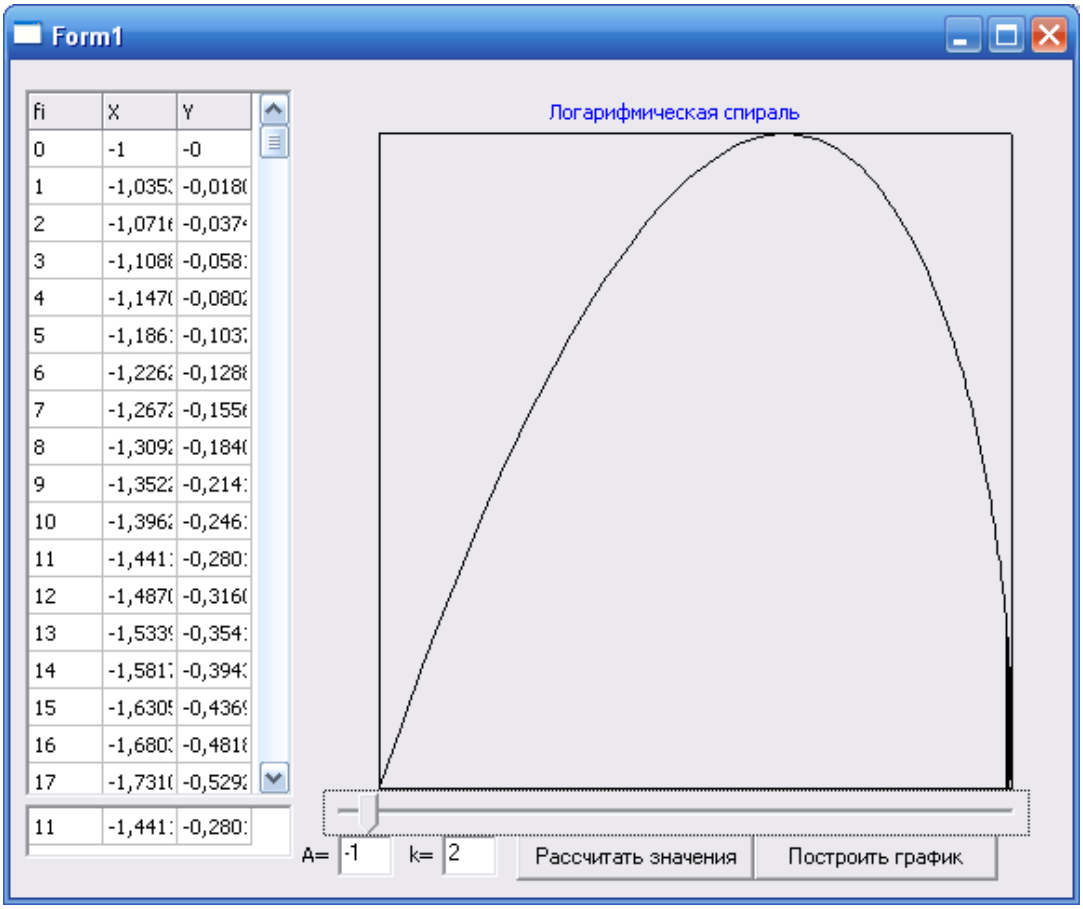


Рисунок 110 – Работа приложения: построение графика для A = -1, k = 2

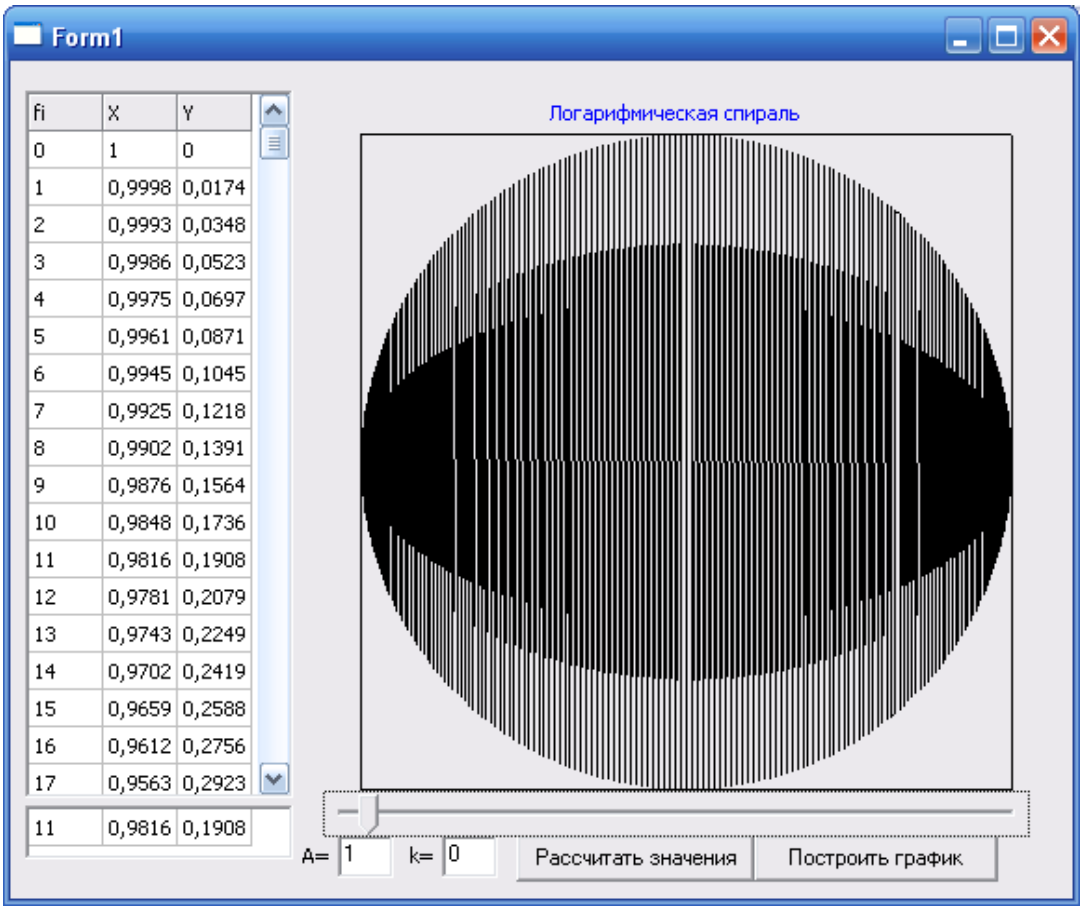


Рисунок 111 – Работа приложения: построение графика для A = 1, k = 0

Выводы. В ходе лабораторной работы было создано приложение для построения графика функции логарифмической спирали. Для этого использовались компоненты Button, LabeledEdit, StringGrid, Chart, TrackBar, Timer, ToolBar; созданы методы для обработки событий FormCreate, FormPaint, ButtonClick, TrackBarChange. Приложение реализовано в виде одной формы; выбор пользователя осуществляется при помощи двух стандартных кнопок и «ползунка».

5.4 Лабораторная работа 4. Работа с базами данных в среде Lazarus

Цель работы. Получить навыки разработки приложений, позволяющих создавать и обрабатывать простейшие базы данных.

Задание к работе. Разработать приложение, которое создает базу данных о студентах в следующем виде:

Фамилия Имя Отчество Группа Дата_рождения R1 R2 R3 R4 R5

(здесь R1 .. R5 – рейтинги по пяти предметам).

Например: Иванов Иван Иванович ИС-15-5 9.09.99 77 75 81 85 92

Две записи должны создаваться программой автоматически, еще не менее пяти добавляются в базу непосредственно при работе приложения с помощью визуальных компонент. Кроме создания базы и визуальной работы с ней, приложение должно также предоставить возможность навигации по базе данных. Предусмотреть индексирование (сортировку) базы данных по своему варианту (табл.6) и ее обработку (табл.7), а также фильтрацию данных (табл.8).

Таблица 6 – Поле для сортировки

Вариант	**Поле для сортировки**
1..5	Фамилия
6..10	Имя
11.15	Группа
16..20	Дата рождения
21..25	R1
26..30	R5

Таблица 7 – Условие для обработки

Вариант	Условие для обработки
1,6,11,16, 21,26	Добавить новое поле – максимальный рейтинг RS по всем предметам; рассчитать его для каждого студента
2,7,12,17, 22,27	Добавить новое поле – минимальный рейтинг RS по всем предметам; рассчитать его для каждого студента
3,8,13,18, 23,28	Добавить новое поле – средний рейтинг RS по всем предметам; рассчитать его для каждого студента
4,9,14,19, 24,29	Добавить новое поле – год рождения; рассчитать его для каждого студента путем анализа поля «Дата рождения»
5,10,15,20, 25,30	Добавить новое поле – месяц рождения; рассчитать его для каждого студента путем анализа поля «Дата рождения»

Таблица 8 – Условие для фильтрации данных

Вар.	Условие для фильтрации данных
1	Фамилии начинаются с букв «А» - «К»
2	Родились позже 1995 года
3	Имена начинаются с букв «А» - «К»
4	Имеют хотя бы один рейтинг «100»
5	Родились раньше 1995 года
6	Имена начинаются с букв «Л» - «Я»
7	Учатся в группах «ИС»
8	Не имеют ни одного рейтинга выше 89 баллов
9	Поступили учиться в 2011 году
10	Родились позже 1993 года
11	Не учатся в группах «ИС»
12	Поступили учиться не в 2011 году
13	Не имеют ни одного рейтинга ниже 75 баллов
14	Фамилии начинаются с букв «Л» - «Я»
15	Не имеют ни одного рейтинга ниже 90 баллов
16	Отчества начинаются с букв «А» - «К»
17	Не имеют ни одного рейтинга выше 74 балла
18	Отчества начинаются с букв «Л» - «Я»
19	Родились раньше 1993 года
20	Имеют хотя бы один рейтинг ниже 75 баллов
21, 26	Рассчитанный рейтинг RS выше 95 баллов
22, 27	Рассчитанный рейтинг RS ниже 75 баллов
23, 28	Рассчитанный рейтинг RS выше 74 баллов
24, 29	Год рождения – 1999
25, 30	Родились летом

Пример выполнения работы

Задание: разработать приложение, осуществляющую работу с базой данных вышеописанного формата, предусмотрев индексирование по фамилии и имени студента. Добавить новое поле – средний рейтинг RS по всем предметам; рассчитать его для каждого студента. Отфильтровать всех отличников (средний рейтинг не ниже 90 баллов).

Текст программы приведен ниже, экранные формы на этапе дизайна и работы приложения – на рис. 112-117.

```
unit Unit1;
{$mode objfpc}{$H+}
interface
uses
  Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs,  StdCtrls, Buttons, ExtCtrls, sqldb, dbf, DBGrids, db, variants, DbCtrls;
type
  { TForm1 }
  TForm1 = class(TForm)
    ButtonCalc: TButton;
    ButtonFilterOn: TButton;
    ButtonFilterOff: TButton;
    ButtonCreate: TButton;
    ButtonFirst: TButton;
    ButtonClose: TButton;
    ButtonNext: TButton;
    ButtonLast: TButton;
    ButtonPrior: TButton;
    ButtonOpen: TButton;
    Datasource1: TDatasource;
    Dbf1: TDbf;
    DBGrid1: TDBGrid;
    DBNavigator1: TDBNavigator;
    procedure ButtonCalcClick(Sender: TObject);
    procedure ButtonCloseClick(Sender: TObject);
```

```
    procedure ButtonCreateClick(Sender: TObject);
    procedure ButtonFilterOffClick(Sender: TObject);
    procedure ButtonFilterOnClick(Sender: TObject);
    procedure ButtonFirstClick(Sender: TObject);
    procedure ButtonLastClick(Sender: TObject);
    procedure ButtonNextClick(Sender: TObject);
    procedure ButtonOpenClick(Sender: TObject);
    procedure ButtonPriorClick(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;
var
  Form1: TForm1;
implementation
{ TForm1 }
procedure TForm1.ButtonOpenClick(Sender: TObject);
var i:integer;
begin
     with Dbf1 do begin
     FilePath:='';
     TableName:='mybase.dbf';
     Open; IndexName:='indFIO';
     DBGrid1.Columns[0].Title.Caption:='Фамилия';
     DBGrid1.Columns[1].Title.Caption:='Имя';
     DBGrid1.Columns[2].Title.Caption:='Отчество';
     DBGrid1.Columns[3].Title.Caption:='Группа';
     DBGrid1.Columns[4].Title.Caption:='Д/р';
     for i:=5 to 9 do DBGrid1.Columns[i].Title.Caption:='R'+IntToStr(i-4);
     for i:=1 to FieldCount do begin
     DBGrid1.Columns[i-1].Title.Alignment:=taCenter;
     case i of
     1..3: DBGrid1.Columns[i-1].Width:=
round(DBGrid1.Columns[i-1].Width*2/3);
```

```
    4,5: DBGrid1.Columns[i-1].Width:=DBGrid1.Columns[i-1].Width-1;
  else DBGrid1.Columns[i-1].Width:=DBGrid1.Columns[i-1].Width div 2
    end end;
    DBGrid1.TitleFont.Style:=[fsBold];
    end
end;
procedure TForm1.ButtonCloseClick(Sender: TObject);
begin
    Dbf1.Close
end;
procedure TForm1.ButtonCalcClick(Sender: TObject);
var i:integer;s:real;
begin
    with Dbf1 do begin
        First;
        while not EOF do begin
            s:=0;
            for i:=1 to 5 do
               s:=s+Fields[4+i].asInteger;
            s:=s/5;
            Edit;
            FieldByName('SR').asFloat:=s;
            Post;
            Next
        end
    end;
end;
procedure TForm1.ButtonCreateClick(Sender: TObject);
begin
    with Dbf1 do begin
    FilePath:='';                    // В текущей папке
    TableName:='mybase.dbf';
    with FieldDefs do begin
        Clear;
        Add('Fam',ftString,20,true);
```

```
    Add('Name',ftString,20,false);
    Add('Parent',ftString,20,false);
    Add('Group',ftString,10,false);
    Add('DR',ftDate,0,false);
    Add('R1',ftInteger,0,false);
    Add('R2',ftInteger,0,false);
    Add('R3',ftInteger,0,false);
    Add('R4',ftInteger,0,false);
    Add('R5',ftInteger,0,false);
    Add('SR',ftFloat,0,false)
    end;{FieldDefs}
CreateTable;
Open;
AddIndex('indFIO','Fam+Name',[ixExpression]);
IndexName:='indFIO';
Append;
FieldByName('Fam').AsString:='Иванов';
FieldByName('Name').AsString:='Петр';
FieldByName('Parent').AsString:='Сергеевич';
FieldByName('Group').AsString:='ИС-12-2';
FieldByName('DR').AsDateTime:=StrToDate('12.01.95');
FieldByName('R1').AsInteger:=55;
FieldByName('R2').AsInteger:=75;
FieldByName('R3').AsInteger:=80;
FieldByName('R4').AsInteger:=65;
FieldByName('R5').AsInteger:=81;
Post;
Append;
FieldByName('Fam').AsString:='Сидорова';
FieldByName('Name').AsString:='Ксения';
FieldByName('Parent').AsString:='Эдуардовна';
FieldByName('Group').AsString:='ИС-12-2';
FieldByName('DR').AsDateTime:=StrToDate('9.09.95');
FieldByName('R1').AsInteger:=75;
FieldByName('R2').AsInteger:=75;
```

```
    FieldByName('R3').AsInteger:=76;
    FieldByName('R4').AsInteger:=77;
    FieldByName('R5').AsInteger:=75;
    Post;
    Close
    end;
    ButtonOpenClick(Sender)
end;
procedure TForm1.ButtonFilterOffClick(Sender: TObject);
begin
    with Dbf1 do begin Filtered:=False; Filter:='' end
end;
procedure TForm1.ButtonFilterOnClick(Sender: TObject);
begin
    with Dbf1 do begin Filter:='SR >= 90'; Filtered:=True end
end;
procedure TForm1.ButtonFirstClick(Sender: TObject);
begin
    Dbf1.First
end;
procedure TForm1.ButtonLastClick(Sender: TObject);
begin
    Dbf1.Last
end;
procedure TForm1.ButtonNextClick(Sender: TObject);
begin
    if not Dbf1.EOF then Dbf1.Next
end;
procedure TForm1.ButtonPriorClick(Sender: TObject);
begin
    if not Dbf1.BOF then Dbf1.Prior
end;
initialization
  {$I unit1.lrs}
end.
```

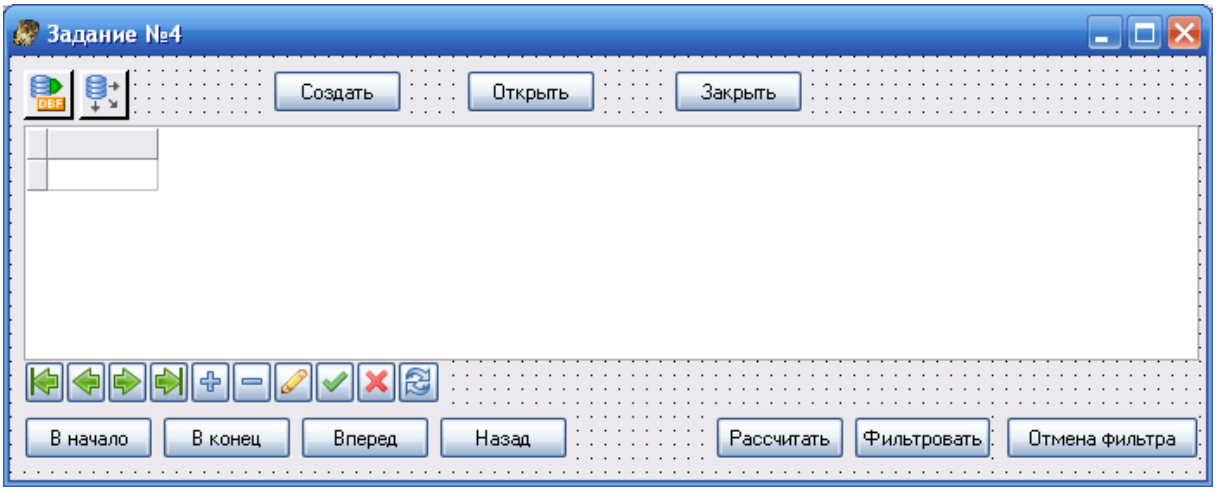


Рисунок 112 – Вид приложения на этапе дизайна

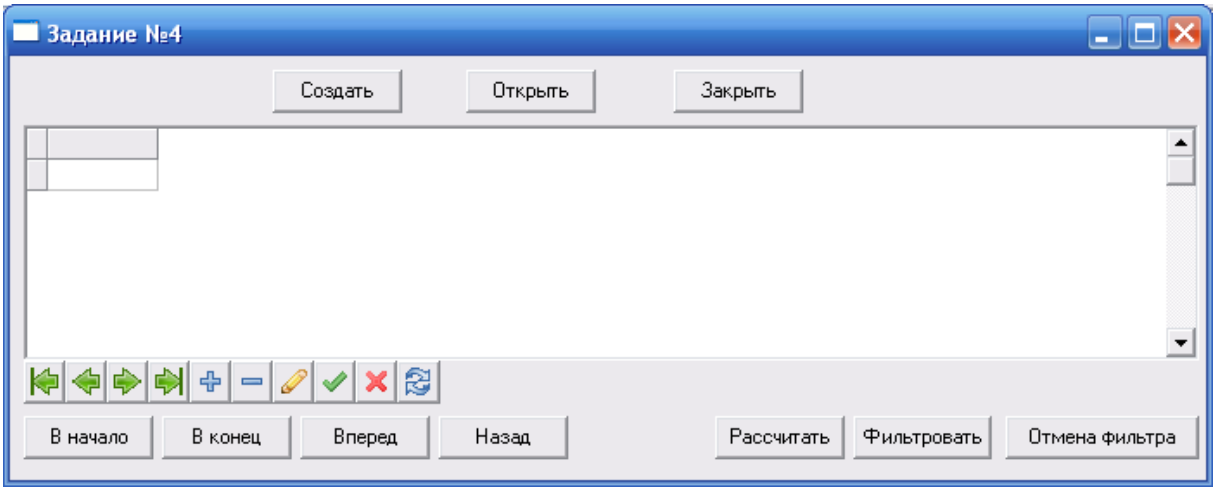


Рисунок 113 – Работа приложения: начало

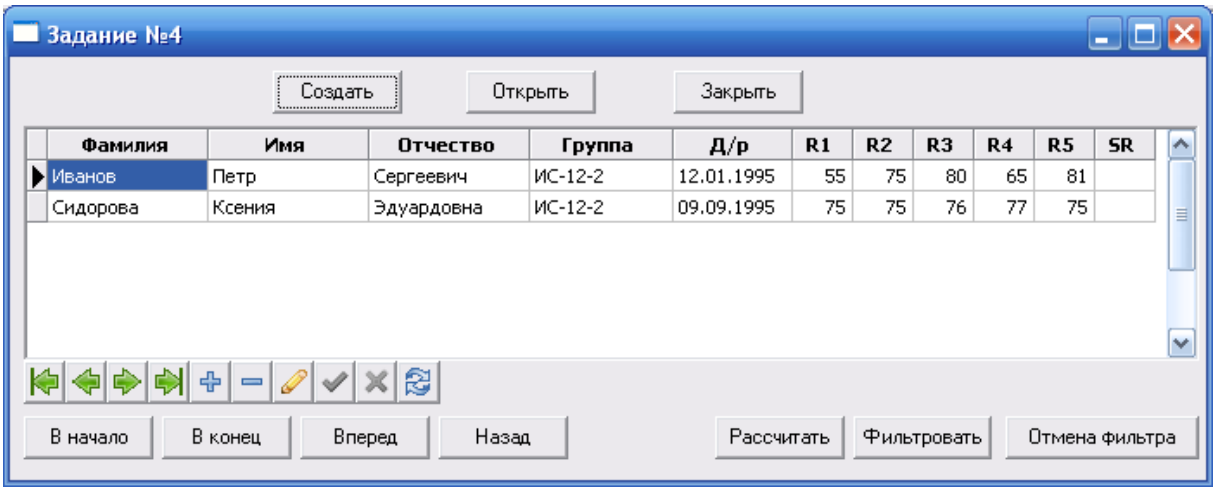


Рисунок 114 – Работа приложения: создание базы данных

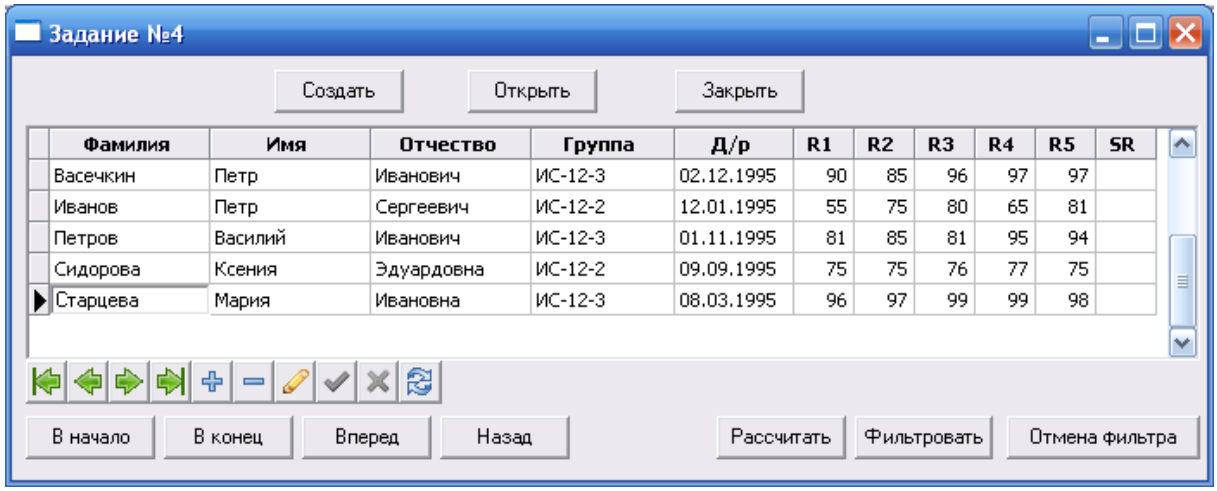


Рисунок 115 – Работа приложения: добавление новых записей

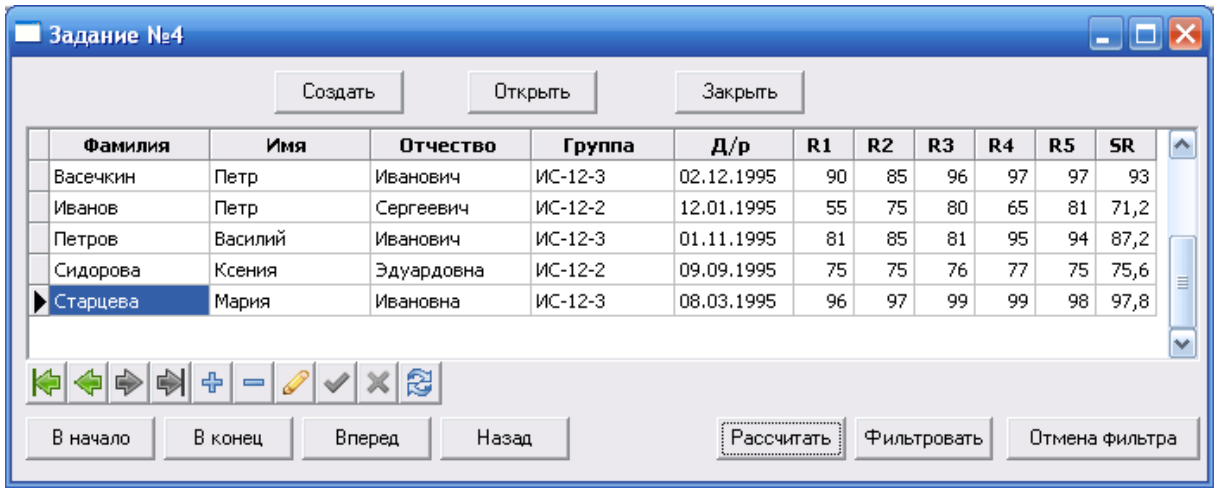


Рисунок 116 – Работа приложения: расчет средних рейтингов

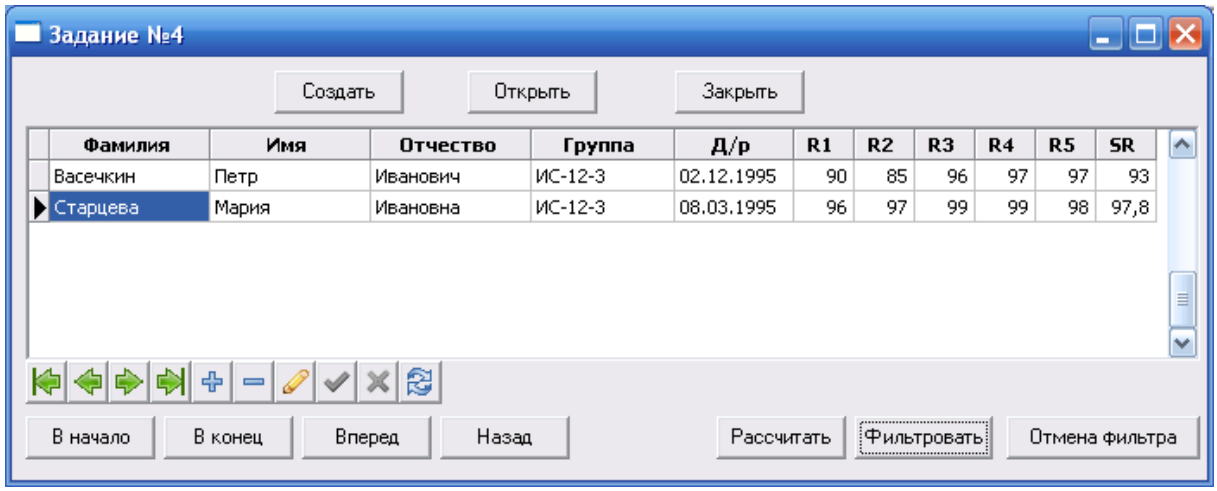


Рисунок 117 – Работа приложения: фильтрация

Выводы. В ходе лабораторной работы было создано приложение для создания и обработки базы данных заданного формата. Для этого использовались компоненты Button, Datasource, Dbf, DBGrid, DBNavigator. Приложение реализовано в виде одной формы; выбор пользователя осуществляется при помощи ряда стандартных кнопок. Пользователю предлагается возможность создать новую базу данных, открыть и закрыть существующую базу данных, перемещаться по набору данных, вводить и модифицировать данные, производить расчет дополнительного поля (средний рейтинг), фильтровать данные (покзывать только студентов – отличников) и отменять фильтр.

5.5 Самостоятельная работа. Создание пакета для обработки данных в среде Lazarus

Задание к работе.

1 Создать текстовый файл, содержащий данные по студентам в следующем виде: Фамилия Имя Группа Год_рождения R1 R2 R3 R4 R5. Например: Иванов Иван ИС-12-3 1995 77 75 81 85 92, где R1..R5 – рейтинги по пяти предметам по 100-балльной шкале; всего файл должен содержать не менее 10 строк – записей.

2 Разработать полнофункциональное Windows-приложение в среде визуального программирования Lazarus, которое бы считывало информацию из файла в набор данных (компонента SdfDataSet) и осуществляло его обработку. Приложение должно содержать, как минимум, главное меню («Вывод данных на экран», «Вывод данных в текстовый файл», «Расчет дополнительного поля», «Сортировка данных по заданному полю», «Построение диаграмм», «Выход»), memo-поле для вывода результатов и PaintBox (Panel) для построения диаграмм. Добавление других компонент для улучшения интерфейса приложения приветствуется.

3 Имена файлов для загрузки/сохранения данных должны задаваться с применением стандартных диалоговых компонент (OpenDialog, SaveDialog). Вывод данных предполагается как на экран, так и в текстовый файл (по выбору пользователя).

4 Дополнительное поле вычисляется согласно индивидуальному заданию (табл. 9). Массив сортируется по заданному полю заданным методом (табл. 10, кол.2 и 3).

5 Столбчатая диаграмма должна отражать сравнение рейтингов по заданному предмету и RS (RM) для каждого студента (номера предметов приведены в табл. 33, кол.4; названия придумать самостоятельно). Требуемый вид диаграммы приведен на рис. 118. Построение диаграмм возможно как при помощи свойства Canvas компоненты PaintBox, так и путем размещения нескольких компонент Shape (с автоматическим вычислением размеров).

Таблица 9 – Варианты для расчета дополнительного поля

Вариант	**Условие**
1..8	Найти максимальный рейтинг RM по всем предметам у каждого студента
9..16	Найти минимальный рейтинг RM по всем предметам у каждого студента
17..30	Найти средний рейтинг RS по всем предметам у каждого студента

Таблица 10 – Варианты заданий

Вар.	**Поле для сортировки**	**Алгоритм сортировки**	**Номер предмета**
1	**2**	**3**	**4**
1	Фамилия	Сортировка обменами	Четвертый
2	Имя	Сортировка вставками	Пятый
3	Группа	Сортировка выбором	Первый
4	Год рождения	Быстрая сортировка Хоора	Второй
5	Рейтинг	Пирамида Уильямса-Флойда	Третий
6	Рейтинг	Сортировка обменами	Четвертый
7	Фамилия	Сортировка вставками	Пятый
8	Имя	Сортировка выбором	Первый
9	Группа	Быстрая сортировка Хоора	Второй
10	Год рождения	Пирамида Уильямса-Флойда	Третий
11	Год рождения	Сортировка обменами	Четвертый
12	Рейтинг	Сортировка вставками	Пятый
13	Фамилия	Сортировка выбором	Первый
14	Имя	Быстрая сортировка Хоора	Второй
15	Группа	Пирамида Уильямса-Флойда	Третий

Продолжение таблицы 10

1	2	3	4
16	Группа	Сортировка обменами	Четвертый
17	Год рождения	Сортировка вставками	Пятый
18	Рейтинг	Сортировка выбором	Первый
19	Фамилия	Быстрая сортировка Хоора	Второй
20	Имя	Пирамида Уильямса-Флойда	Третий
21	Имя	Сортировка обменами	Четвертый
22	Группа	Сортировка вставками	Пятый
23	Год рождения	Сортировка выбором	Первый
24	Рейтинг	Быстрая сортировка Хоора	Второй
25	Фамилия	Пирамида Уильямса-Флойда	Третий
26	Имя	Быстрая сортировка Хоора	Четвертый
27	Группа	Пирамида Уильямса-Флойда	Пятый
28	Год рождения	Быстрая сортировка Хоора	Первый
29	Рейтинг	Сортировка обменами	Второй
30	Год рождения	Сортировка вставками	Третий

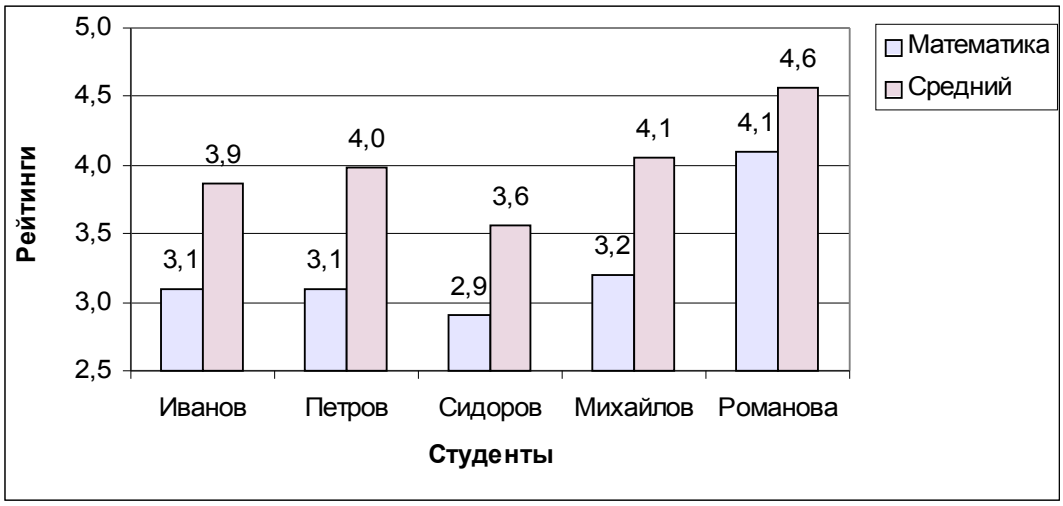


Рисунок 118 – Примерный вид результатов расчетов

СПИСОК ЛИТЕРАТУРЫ

1 **Алексеев, Е.Р.** Самоучитель по программированию на FreePascal и Lazarus / Е.Р. Алексеев, О.В. Чеснокова, Т.В. Кучер. – Донецк. : ДонНТУ, Технопарк ДонНТУ УНИТЕХ, 2009. – 503 с. – ISBN 978-966-8248-26-9.

2 **Мансуров, К.Т.** Основы программирования в среде Lazarus / К.Т. Мансуров. – 2010. – 772 с. – ISBN 978-9967-03-646-8.

3 **Фаронов, В.В.** Система программирования Delphi / В.В. Фаронов. – СПб. : БХВ–Петербург, 2003. – 912 с.

4 **Гофман, В.Э.** Работа с базами данных в Delphi / В.Э. Гофман, А.Д. Хомоненко. – СПб : БХВ – Санкт-Петербург, 2001. – 656 с.