

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

2023 Շ.

Задание 1. Создание программной системы на Object Pascal

Выполнить объектную декомпозицию, разработать формы интерфейса, диаграмму состояний интерфейса, диаграммы классов интерфейсной и предметной областей, диаграмму последовательности действий одной из реализуемых операций. Разработать, протестировать и отладить программу.

Сведения о студентах включают: фамилию, имя, индекс группы, рейтинг (от 0 до 100). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Выяснить, имеются ли в институте однофамильцы. Если имеются, показать их фамилии и имена.
2. Выяснить, в каких группах обучается количество студентов более заданного.
3. Получить список студентов с указанием группы, рейтинг которых не меньше зачетного (60).
4. Построить гистограмму, показывающую средний рейтинг по каждой группе.

Цель работы

Получение практических навыков разработки прикладного приложения с графическим интерфейсом с использованием языка программирования ObjectPascal и библиотеки компонентов LCL.

Интерфейс приложения

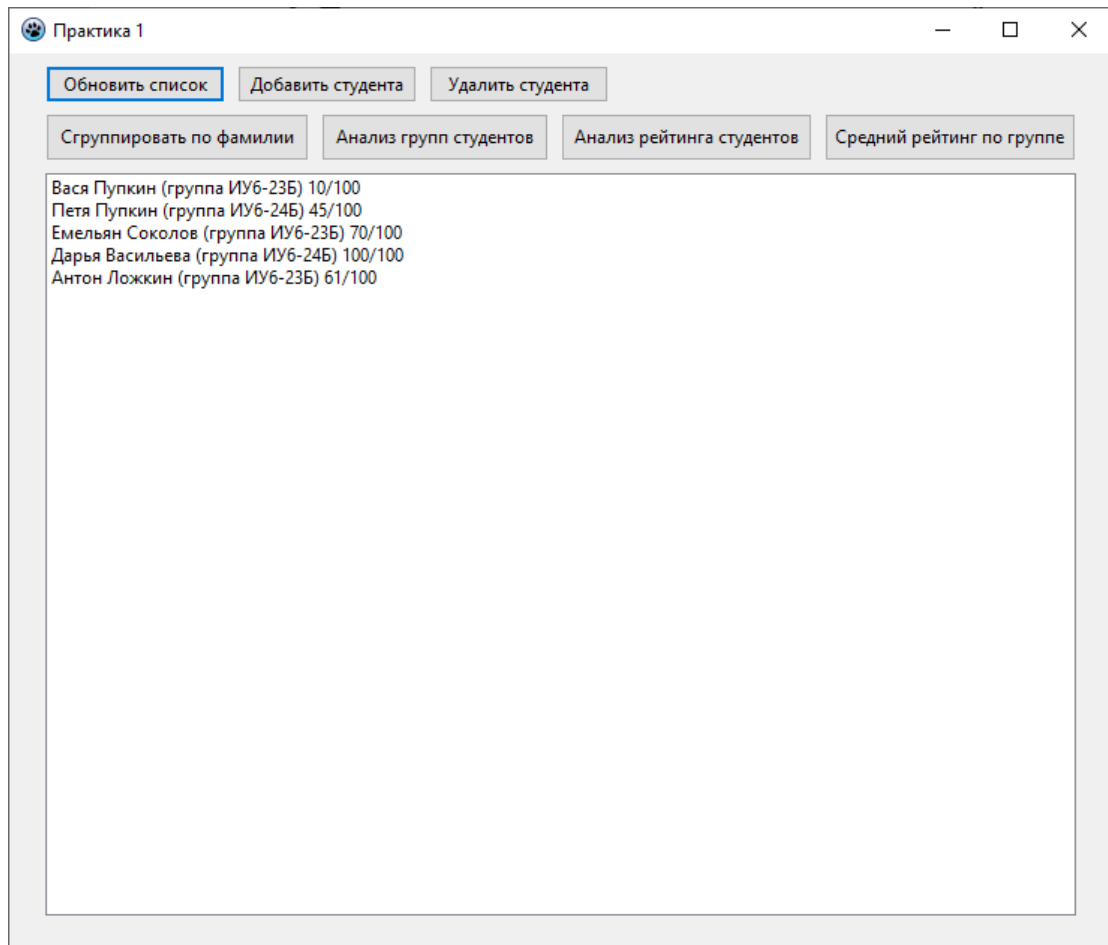


Рисунок 1 - главное меню программы

The screenshot shows a window titled 'Новый студент'. It contains four text input fields stacked vertically, labeled 'Имя', 'Фамилия', 'Группа', and 'Рейтинг (от 0 до 100)'. At the bottom of the form is a button labeled 'Добавить студента', which is highlighted with a blue border.

Рисунок 2 - форма добавления нового студента

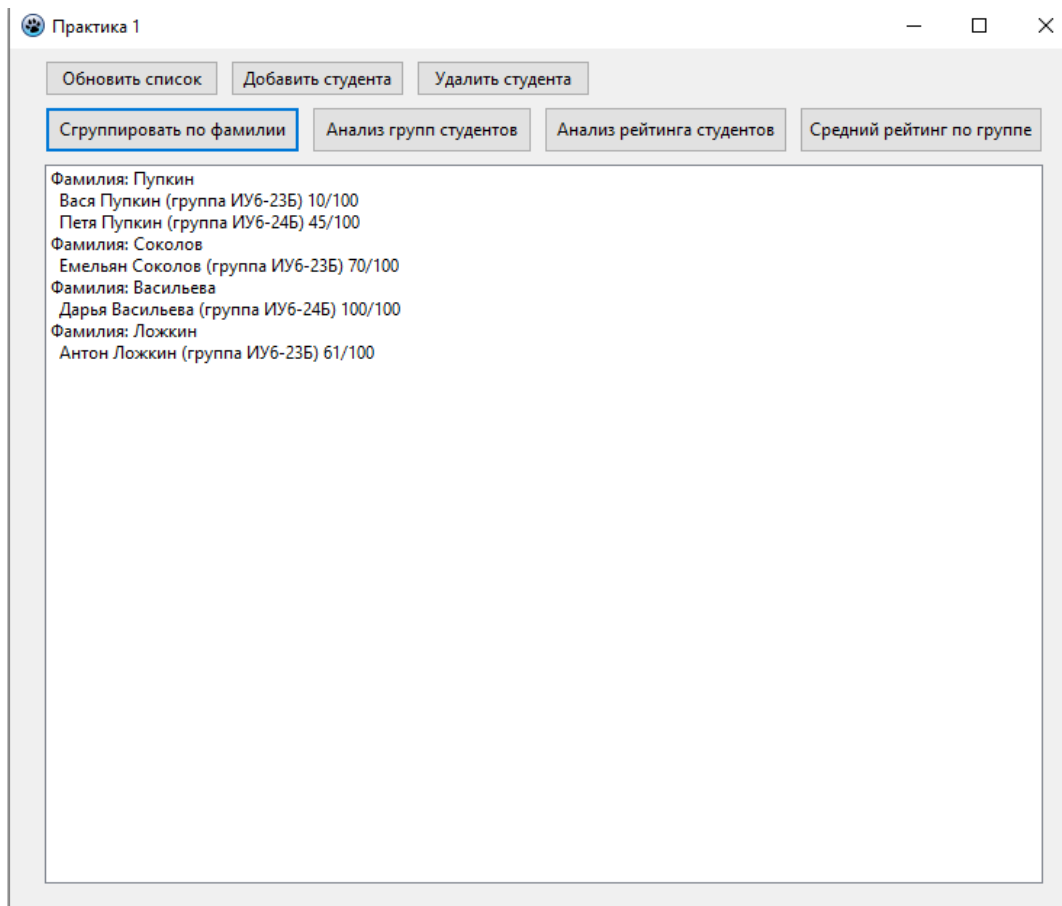


Рисунок 3 - режим группировки по фамилии

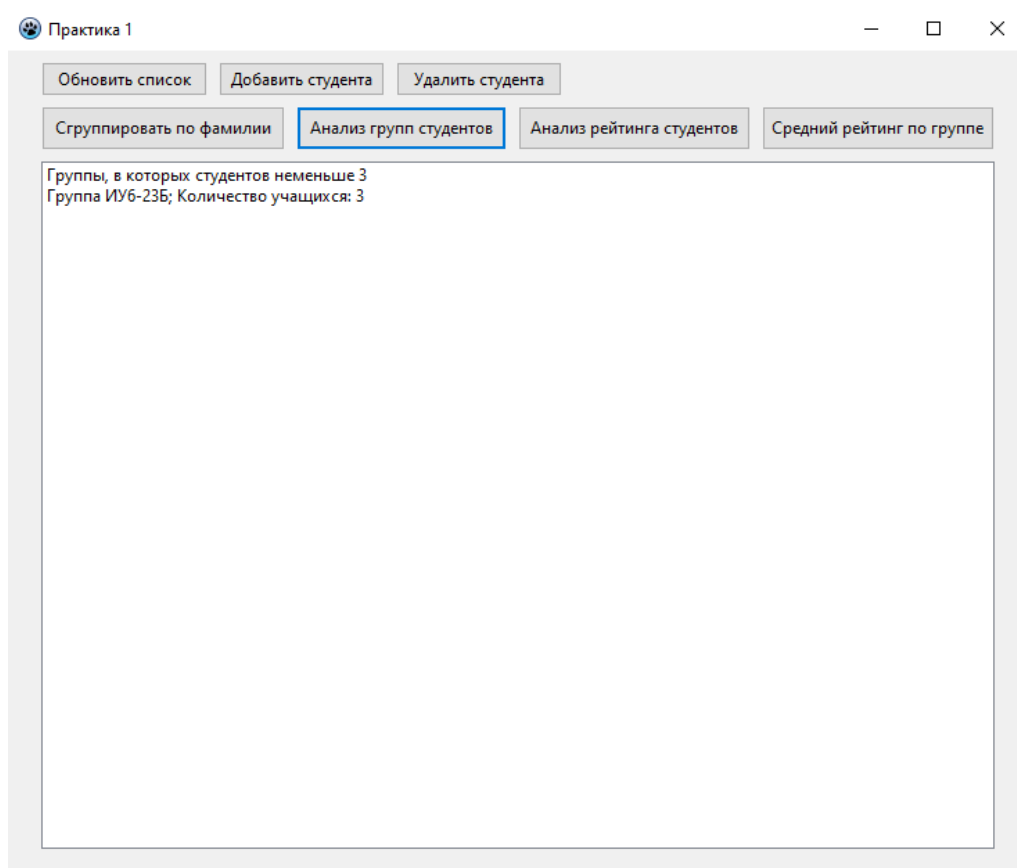


Рисунок 4 - группировка по количеству студентов в группе

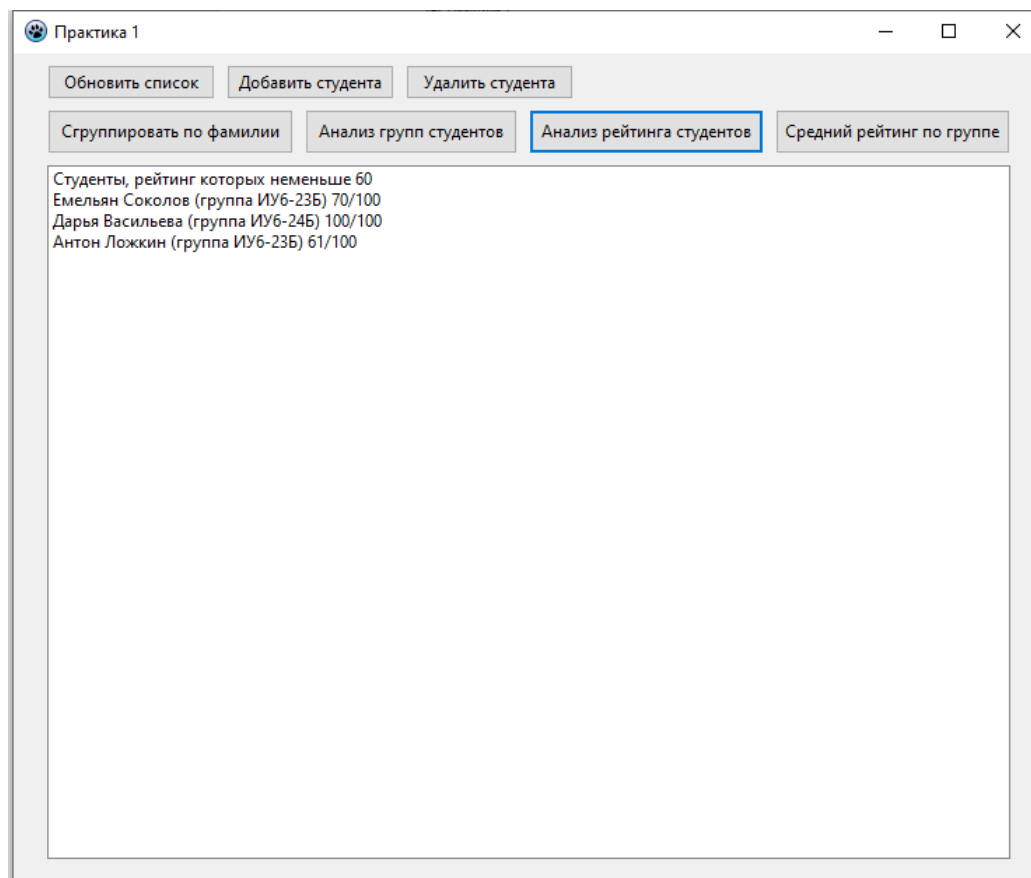


Рисунок 5 - фильтр по количеству баллов

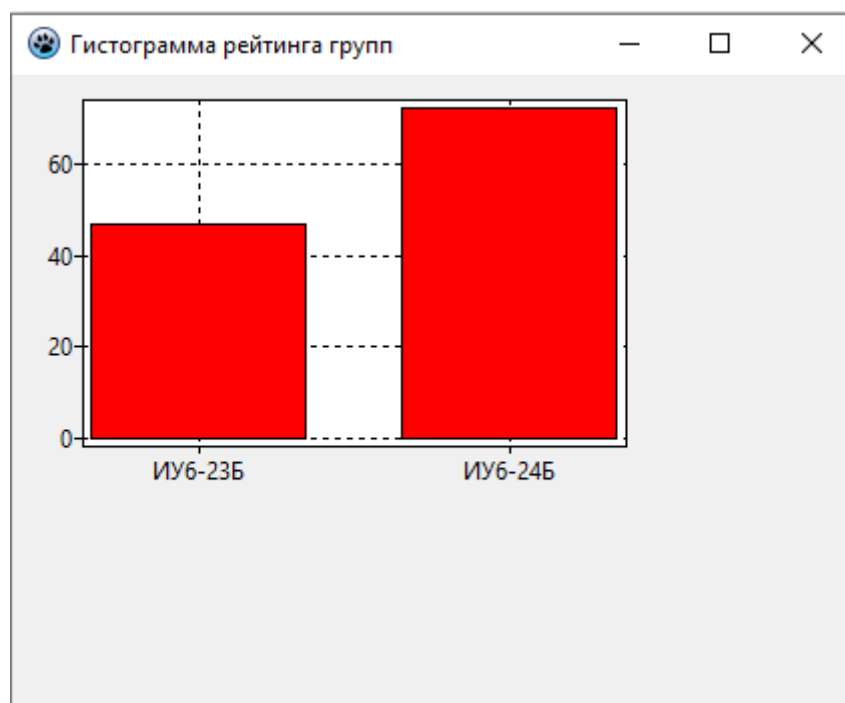


Рисунок 6 - гистограмма среднего рейтинга групп

Проект приложения

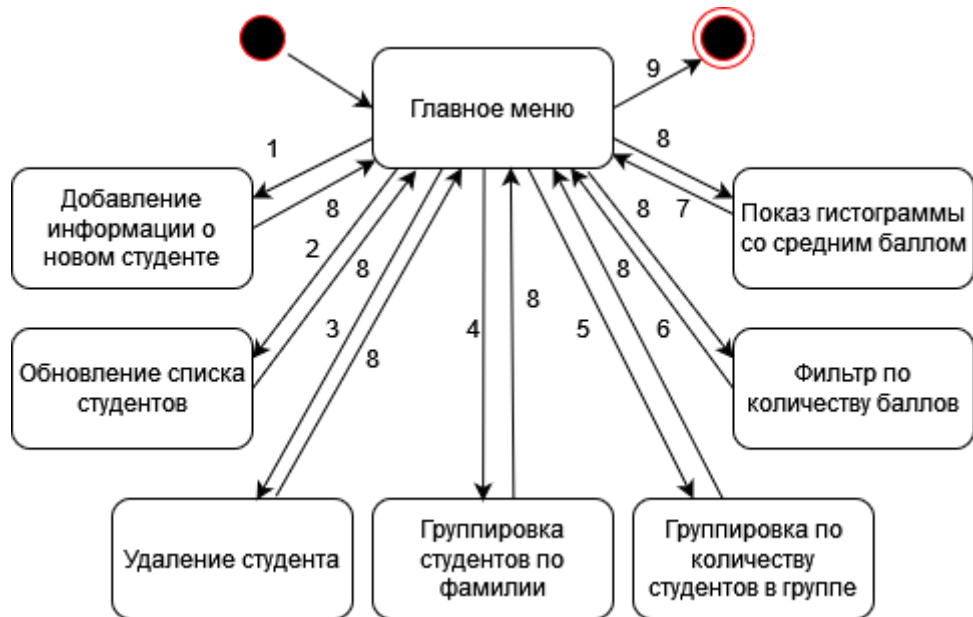


Рисунок 7 - диаграмма состояний

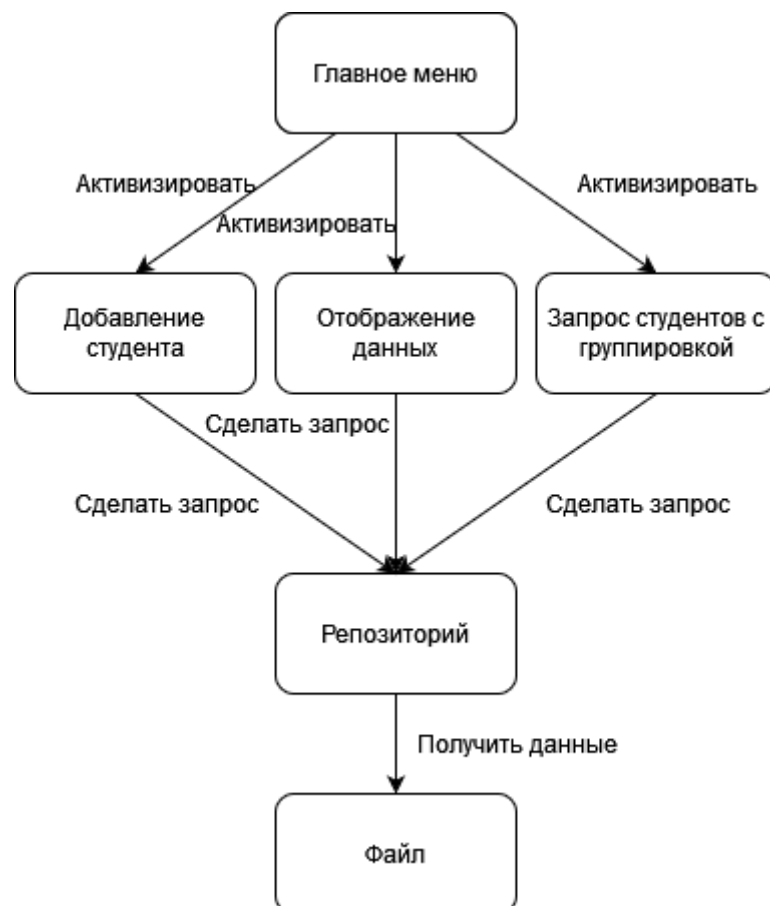


Рисунок 8 - объектная декомпозиция

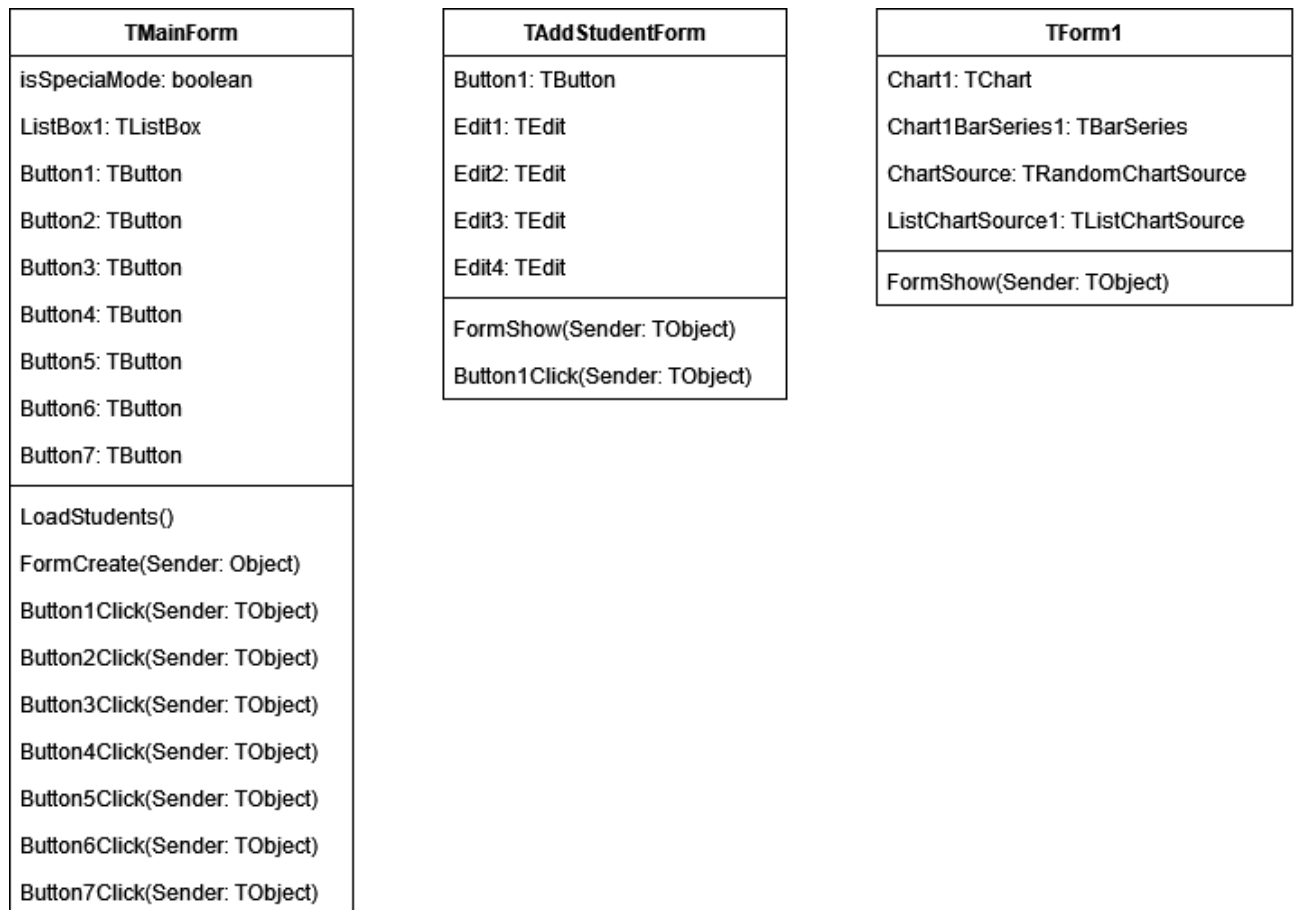


Рисунок 9 - объектная декомпозиция интерфейса

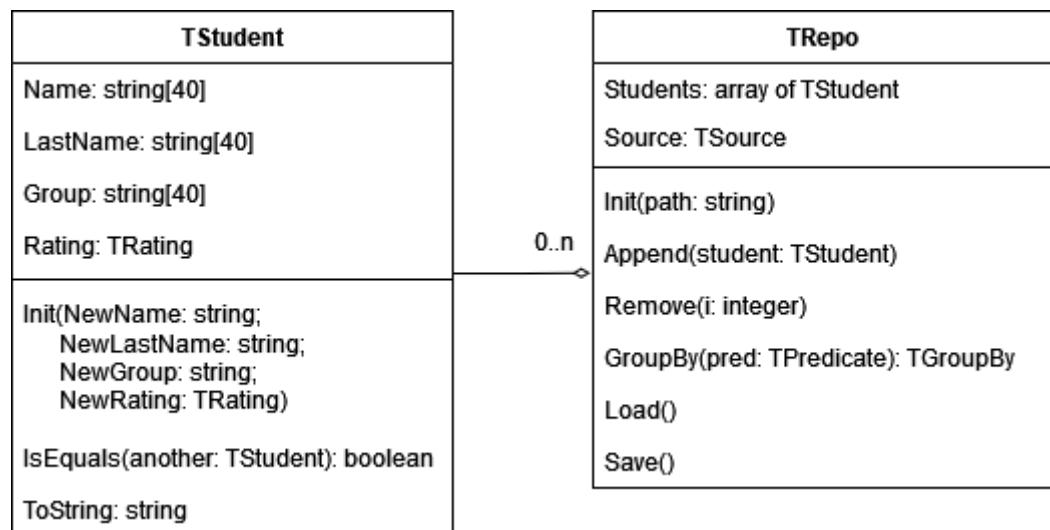


Рисунок 10 - объектная декомпозиция предметной области

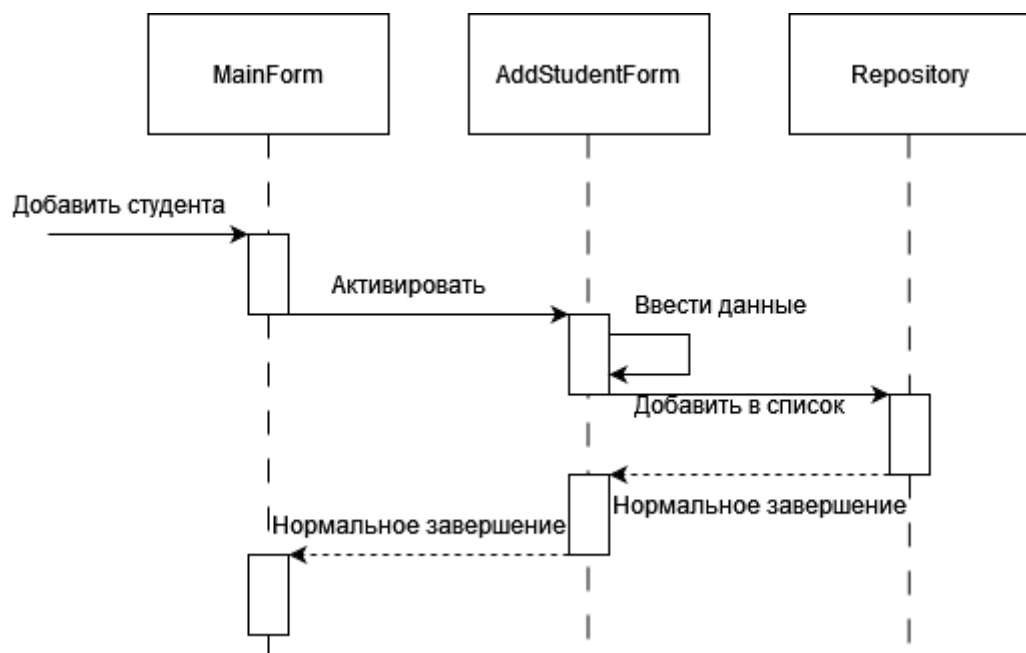


Рисунок 11 - диаграмма последовательности действий при добавлении нового студента

Код приложения


```

1  unit domain;
.
.  {$mode ObjFPC}{$H+}
.
5  interface
.
.  uses
.  [
.    Classes, SysUtils;
.  ]
.
10 type
.   TRating = 0..100;
.
.   TStudent = object
.   public
15     Name: string[40];
.     LastName: string[40];
.     Group: string[40];
.     Rating: TRating;
.   public
20     constructor Init(
.       NewName: string;
.       NewLastName: string;
.       NewGroup: string;
.       NewRating: TRating);
25     function IsEquals(another: TStudent): boolean;
.     function ToString: string;
.   end;
.
.   TGroupBy = array of array of TStudent;
30   TPredicate = function(a, b: TStudent): boolean;
.
.   function isNamesakes(a, b: TStudent): boolean;
.   function isGroupmates(a, b: TStudent): boolean;

```

Рисунок 12 - код модуля домена

```

35  implementation
.
.
.  constructor TStudent.Init(
.      NewName: string;
.      NewLastName: string;
40      NewGroup: string;
.      NewRating: TRating);
.  begin
.      Name := NewName;
.      LastName := NewLastName;
45      Group := NewGroup;
.      Rating := NewRating;
.  end;
.
48  function TStudent.IsEquals(another: TStudent): boolean;
50  begin
.      Result :=
.          (Name = another.Name) and
.          (LastName = another.LastName) and
.          (Group = another.Group) and
55      (Rating = another.Rating);
.  end;
.
.  function TStudent.ToString: string;
.  begin
60      Result :=
.          Name + ' ' + LastName +
.          ' (группа ' + Group + ') ' +
.          IntToStr(Rating) + '/100';
.  end;
.
65  function isNamesakes(a, b: TStudent): boolean;
.  begin
.      Result := a.LastName = b.LastName;
.  end;
.
70  function isGroupmates(a, b: TStudent): boolean;
.  begin
.      Result := a.Group = b.Group;
.  end;
.
75
76  end.

```

Рисунок 13 - код модуля домена

```

1  unit repository;
.
.  {$mode delphi}{$H+}
.
5  interface
.
.  uses
.  [
.    Classes, SysUtils, Domain;
.  ]
.
10 type
.   TSource = file of TStudent;
.
.  TRepo = object
.  [
.  [
15     Students: array of TStudent;
.     Source: TSource;
.  [
.  [
.     constructor Init(path: string);
.     procedure Append(student: TStudent);
20     procedure Remove(i: integer);
.     function GroupBy(pred: TPredicate): TGroupBy;
.     procedure Load;
23     procedure Save;
.   end;
.
25
.  var Repo: TRepo;
.

```

Рисунок 14 - код модуля репозитория

```

.   implementation
.
30  constructor TRepo.Init(path: string);
.   begin
.       if not FileExists(path) then
.           FileCreate(path);
.           AssignFile(Source, path);
35  end;
.
.   procedure TRepo.Append(student: TStudent);
.   begin
.       SetLength(Students, Length(Students)+1);
40  Students[Length(Students)-1] := student;
.       Save;
.   end;
.
.   procedure TRepo.Remove(i: integer);
45  begin
.   if (Length(Students) > 0) and (i > -1) then begin
.       Students[i] := Students[Length(Students)-1];
.       SetLength(Students, Length(Students)-1);
.   end;
50  Save;
.   end;
.
.   function TRepo.GroupBy(pred: TPredicate): TGroupBy;
.   var i, j: integer;
55  found: boolean;
.   begin
.   for i := 0 to High(Students) do begin
.       found := false;
.       j := 0;
60  while (j < Length(Result)) and not found do begin
.   if pred(Students[i], Result[j][0]) then begin
.       SetLength(Result[j], Length(Result[j]) + 1);
.       Result[j][Length(Result[j])-1] := Students[i];
.       found := true;
65  end;
.       j += 1;
.   end;
.   if not found then begin
.       j := Length(Result);
70  SetLength(Result, j+1);
.       SetLength(Result[j], 1);
.       Result[j][0] := Students[i];
.   end;
.   end;
75  end;

```

Рисунок 15 - код модуля репозитория

```

. procedure TRepo.Load;
.   var length, i: integer;
.   begin
80     reset(Source);
.     length := FileSize(Source);
.     SetLength(Students, length);
.     i := 0;
.   while not eof(Source) do begin
85     Read(Source, Students[i]);
.     i += 1;
.   end;
.   close(Source);
.   end;
90
. procedure TRepo.Save;
.   var i: integer;
.   begin
.     rewrite(Source);
95   for i := 0 to High(Students) do begin
.     Write(Source, Students[i]);
.   end;
.   close(Source);
.   end;
100
. end.

```

Рисунок 16 - код модуля репозитория

```

1  unit main;
.
.  {$mode delphi}{$H+}
.
5  interface
.
.  uses
8    Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
.    Repository, Domain, AddStudent, Gist;
10
.  type
.
.  { TMainForm }
.
15 TMainForm = class(TForm)
.    Button1: TButton;
.    Button2: TButton;
.    Button3: TButton;
.    Button4: TButton;
20    Button5: TButton;
.    Button6: TButton;
.    Button7: TButton;
.    ListBox1: TListBox;
.    procedure Button4Click(Sender: TObject);
25    procedure Button5Click(Sender: TObject);
.    procedure Button6Click(Sender: TObject);
.    procedure Button7Click(Sender: TObject);
.    procedure LoadStudents;
.    procedure Button1Click(Sender: TObject);
30    procedure Button2Click(Sender: TObject);
.    procedure Button3Click(Sender: TObject);
.    procedure FormCreate(Sender: TObject);
.  private
.    isSpecialMode: boolean;
35  public
.
.    end;
.
.  var
40    MainForm: TMainForm;
.

```

Рисунок 17 - код модуля main

```

.   implementation
.
.   {$R *.lfm}
45
.   { TMainForm }
47
.   procedure TMainForm.LoadStudents;
.   var
50     i: integer;
.   begin
.     ListBox1.Clear;
.     for i := 0 to High(Repo.Students) do
.     begin
55       ListBox1.Items.Add(Repo.Students[i].ToString);
.     end;
.     isSpecialMode := False;
.     end;
.
.   procedure TMainForm.FormCreate(Sender: TObject);
.   begin
.     Repo.Init('students.dat');
.     Repo.Load;
.     end;
65
.   procedure TMainForm.Button1Click(Sender: TObject);
.   begin
.     AddStudentForm.Show;
.     end;
70
.   procedure TMainForm.Button2Click(Sender: TObject);
.   begin
.     if not isSpecialMode then
.     begin
75       Repo.Remove(ListBox1.ItemIndex);
.       LoadStudents;
.     end;
.     end;
.
.   procedure TMainForm.Button3Click(Sender: TObject);
80   begin
.     LoadStudents;
.     end;

```

Рисунок 18 - код модуля main

```

85 procedure TMainForm.Button4Click(Sender: TObject);
. var
.   namesakes: TGroupBy;
.   i, j: integer;
. begin
90   namesakes := Repo.GroupBy(isNamesakes);
.   isSpecialMode := True;
.   ListBox1.Clear;
.   for i := 0 to High(namesakes) do
.   begin
95     ListBox1.Items.Add('Фамилия: ' + namesakes[i][0].LastName);
.     for j := 0 to High(namesakes[i]) do
.     begin
.       ListBox1.Items.Add(' ' + namesakes[i][j].ToString);
.     end;
100   end;
. end;

. procedure TMainForm.Button5Click(Sender: TObject);
. var
105   groupmates: TGroupBy;
.   StudentsAmountStr: string;
.   i, StudentsAmountInt: integer;
. begin
.   InputQuery('Фильтр',
110     'Порог фильтрации групп по количеству студентов (>=)',
.     StudentsAmountStr);
.   StudentsAmountInt := StrToInt(StudentsAmountStr);
.   groupmates := Repo.GroupBy(isGroupmates);
.   ListBox1.Clear;
115   ListBox1.Items.Add('Группы, в которых студентов неменьше ' + StudentsAmountStr);
.   isSpecialMode := True;
.   for i := 0 to High(groupmates) do
.   begin
.     if Length(groupmates[i]) >= StudentsAmountInt then
120       ListBox1.Items.Add('Группа ' + groupmates[i][0].Group +
.         '; Количество учащихся: ' + IntToStr(Length(groupmates[i])));
.     end;
.   end;

. procedure TMainForm.Button6Click(Sender: TObject);
125   var i: integer;
.   begin
.     ListBox1.Clear;
.     ListBox1.Items.Add('Студенты, рейтинг которых неменьше 60');
130     for i := 0 to High(Repo.Students) do
.       if Repo.Students[i].Rating >= 60 then
.         ListBox1.Items.Add(Repo.Students[i].ToString);
.     end;

. procedure TMainForm.Button7Click(Sender: TObject);
135   begin
.     GistForm.Show;
.   end;

```

Рисунок 19 - код модуля main


```

1  unit addStudent;
.
.  {$mode ObjFPC}{$H+}
.
5  interface
.
.  uses
.  [
.      Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
.      Domain, repository;
10 ]
.
.  type
.
.  [
.      { TAddStudentForm }
.
15 ]
.  TAddStudentForm = class(TForm)
.      Button1: TButton;
.      Edit1: TEdit;
.      Edit2: TEdit;
.      Edit3: TEdit;
20      Edit4: TEdit;
.      procedure Button1Click(Sender: TObject);
.      procedure FormShow(Sender: TObject);
.  private
.
25 [
.      public
.
.      end;
.
.
29 var
.      |
30      AddStudentForm: TAddStudentForm;

```

Рисунок 20 - код модуля addStudent

```

implementation

{$R *.lfm}

[ { TAddStudentForm }

[ procedure TAddStudentForm.FormShow(Sender: TObject);
[ begin
.      Edit1.Clear;
.      Edit2.Clear;
.      Edit3.Clear;
.      Edit4.Clear;
.      end;

[ procedure TAddStudentForm.Button1Click(Sender: TObject);
.      var student: TStudent;
[ begin
.      student.Init(edit1.text, edit2.text, edit3.text, StrToInt(edit4.text));
.      Repo.Append(student);
.      Hide;
.      end;

end.

```

Рисунок 21 - код модуля addStudent

```

1  unit gist;
.
.  {$mode delphi}{$H+}
.
5  interface
.
.  uses
.  [ Classes, SysUtils, Forms, Controls, Graphics, Dialogs, TAGraph, TASources,
.  [   TASeries, TACustomSource, TADbSource, Repository, Domain;
10
.  type
.
.  [ { TForm1 }
.
15 [ TForm1 = class(TForm)
.  [   Chart1: TChart;
.  [   Chart1BarSeries1: TBarSeries;
.  [   ChartSource: TRandomChartSource;
.  [   ListChartSource1: TListChartSource;
20 [   procedure FormShow(Sender: TObject);
.  [   function ListChartSource1Compare(AItem1, AItem2: Pointer): integer;
.  [ private
.  [
.  [ public
25 [
.  [   end;
.
.
.  var
.  [   GistForm: TForm1;
30
31 implementation
.  [ {$R *.lfm}
.
.  [ { TForm1 }
35
.  [ procedure TForm1.FormShow(Sender: TObject);
.  [ var groupmates: TGroupBy;
.  [   i, j: integer;
.  [   avr: double;
40 [ begin
.  [   groupmates := Repo.GroupBy(isGroupmates);
.  [   ListChartSource1.Clear;
.  [   for i := 0 to High(groupmates) do
.  [   [ begin
45 [     avr := 0;
.  [     for j := 0 to High(groupmates[i]) do
.  [       avr += groupmates[i][j].Rating;
.  [     avr /= Length(groupmates[i]);
.  [     ListChartSource1.Add(i, avr, groupmates[i][0].Group);
50 [   end;
.  [ end;
.

```

Рисунок 22 - код модуля gist

Выводы

В результате были получены практические навыки разработки прикладных приложений с графическим интерфейсом на языке программирования ObjectPascal с использованием библиотеки компонентов LCL.