Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

ОТЧЕТ ОБ УЧЕБНОЙ ПРАКТИКЕ

Тип практики: Проектно-технологическая практика

Название предприятия: НУК ИУ МГТУ им. Н.Э. Баумана

Студент группы ИУ6-23Б — 26.03.2023 (Подпись, дата) — В.К. Залыгин (И.О. Фамилия)

Руководитель практики — _____ (Подпись, дата) — А.М. Минитаева (И.О. Фамилия)

Оценка _____

2023 г.

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

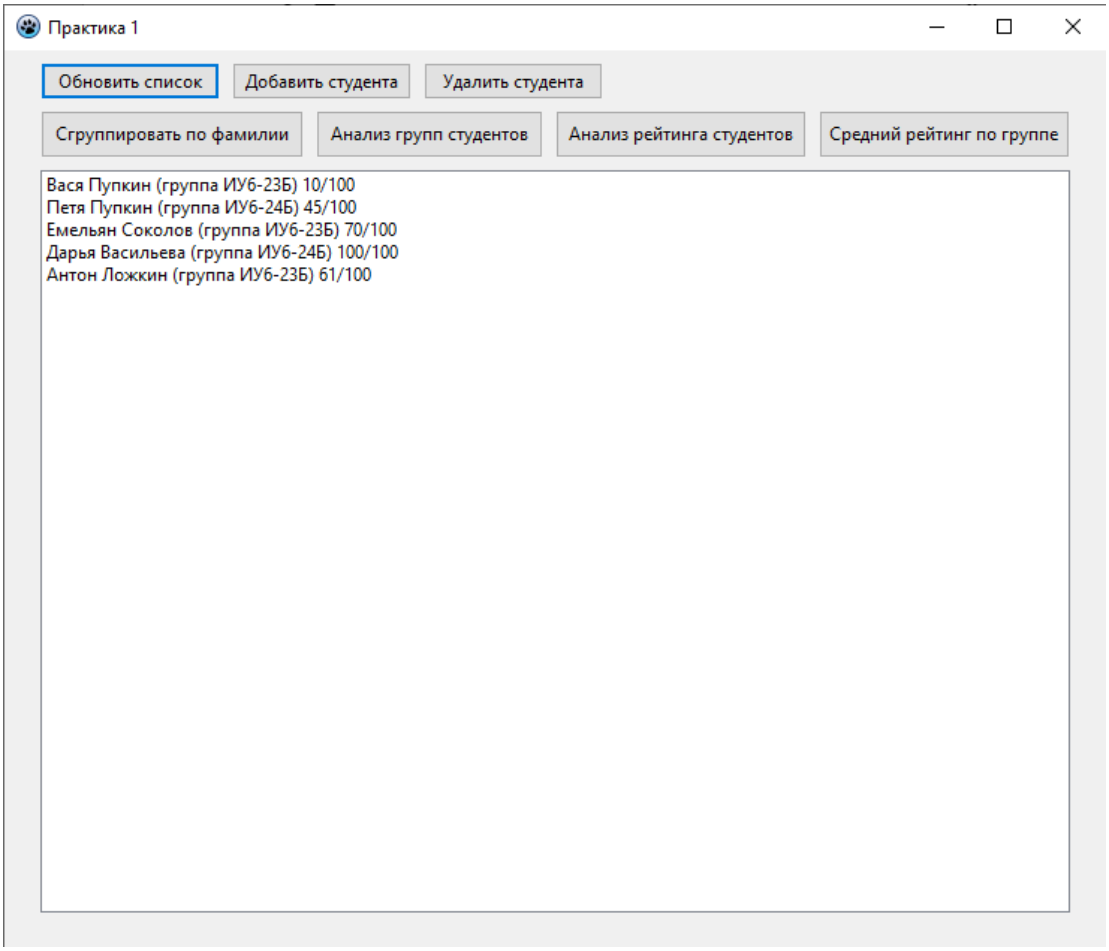

Рисунок 1 - главное меню программы

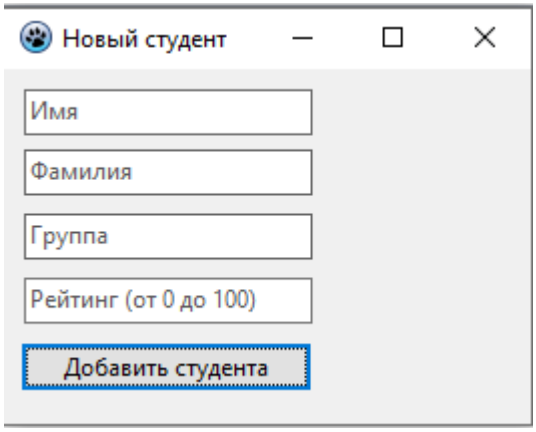

Рисунок 2 - форма добавления нового студента

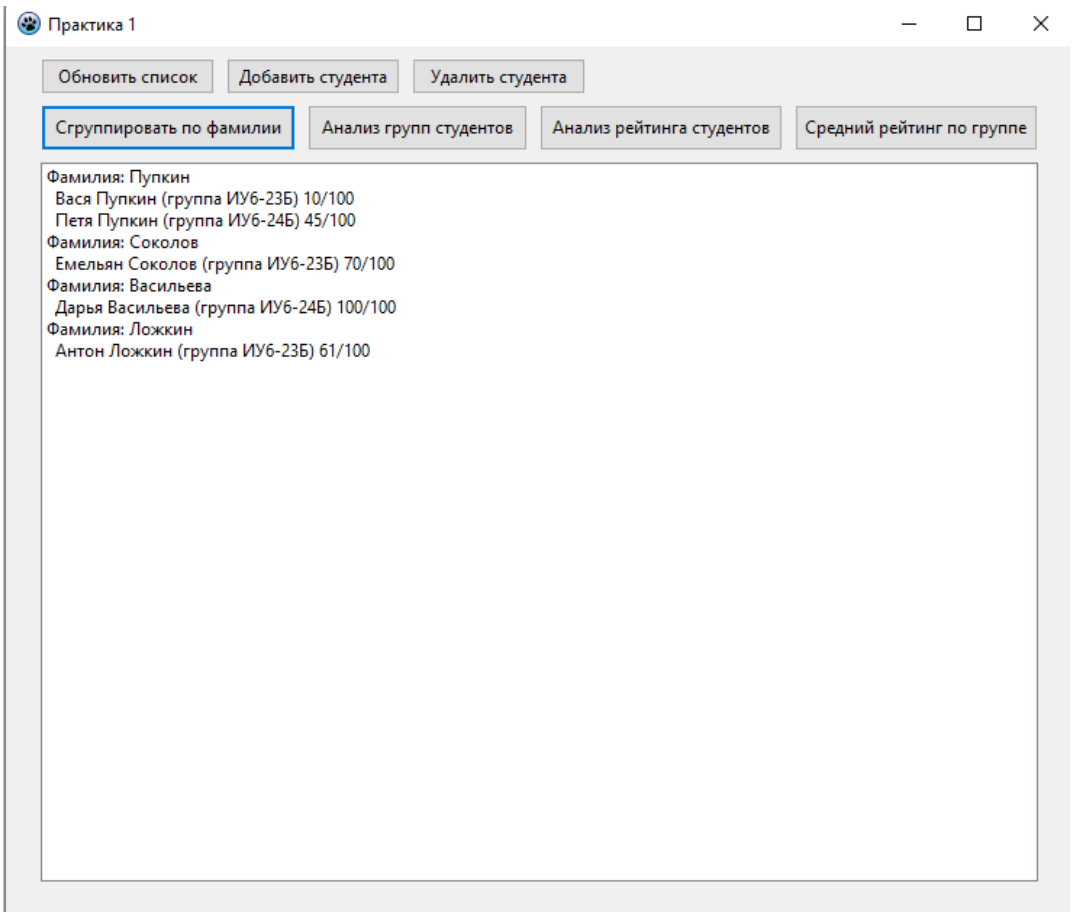

Рисунок 3 - режим группировки по фамилии

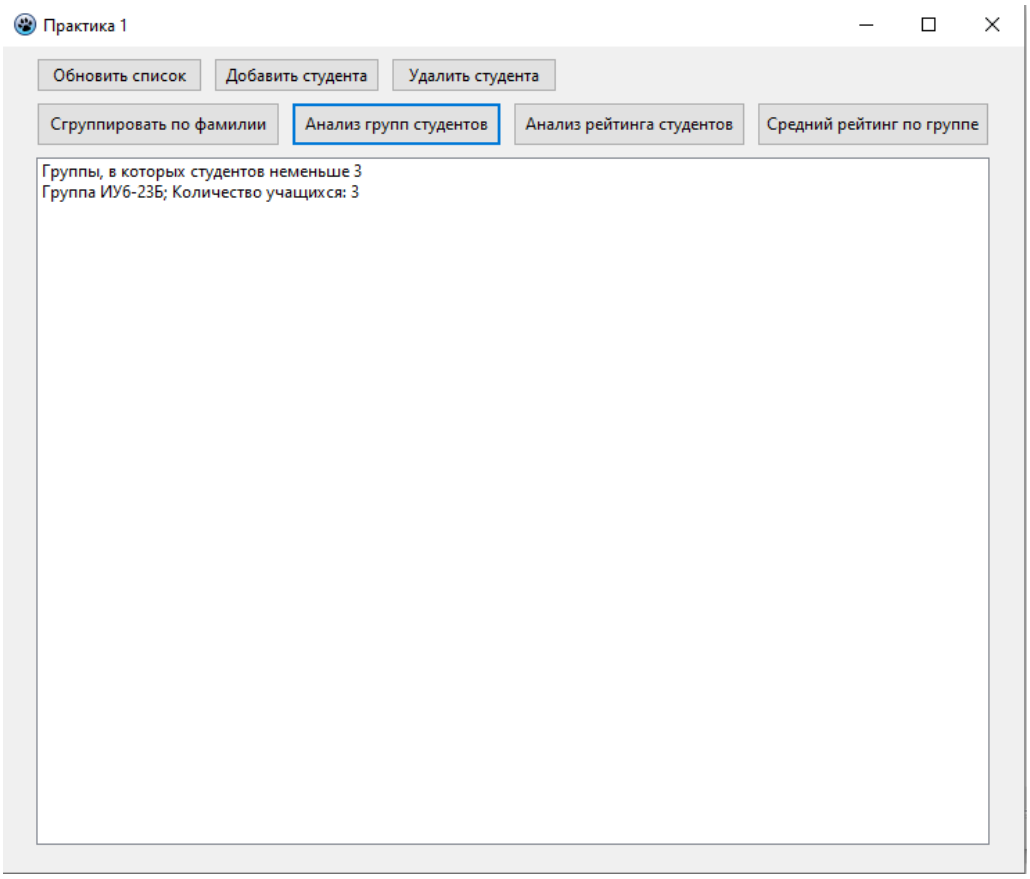

Рисунок 4 - группировка по количеству студентов в группе

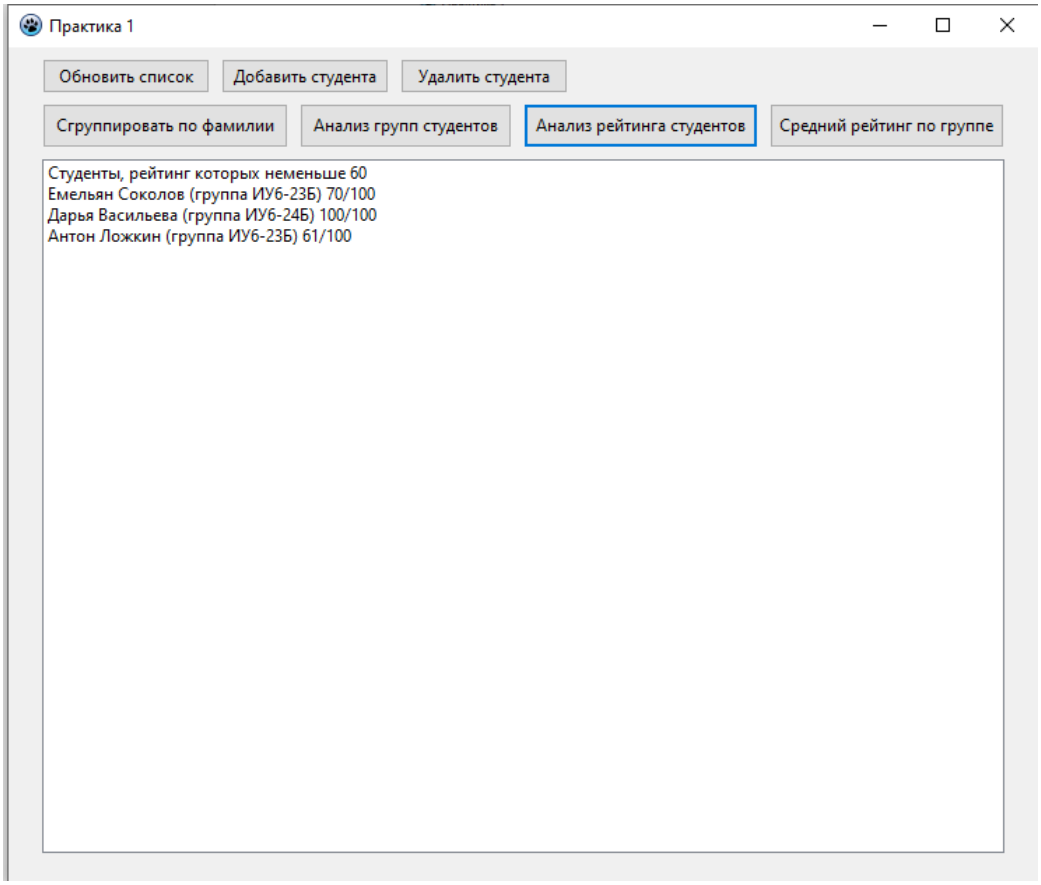

Рисунок 5 - фильтр по количеству баллов

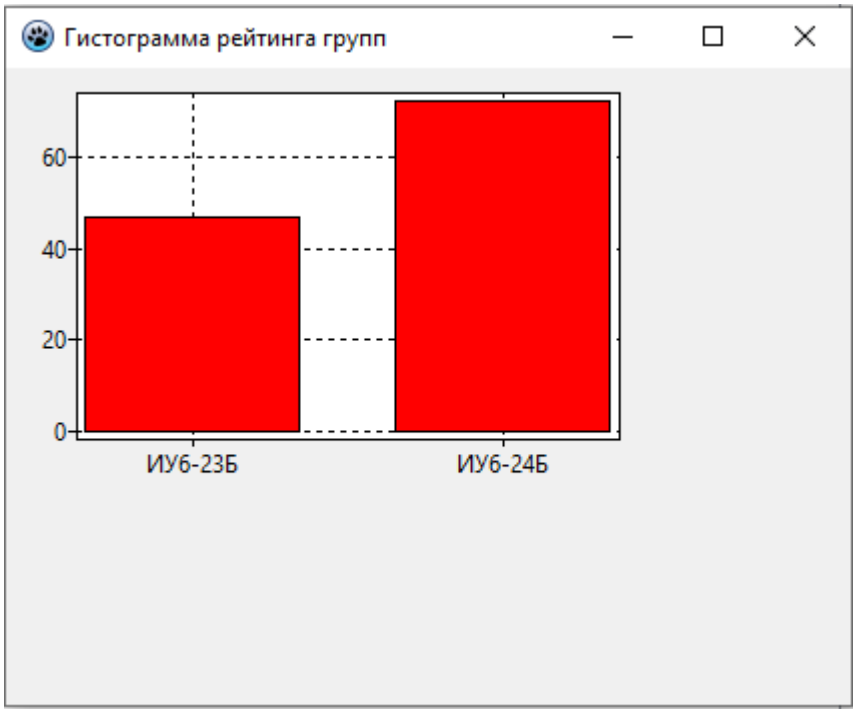

Рисунок 6 - гистограмма среднего рейтинга групп

Проект приложения

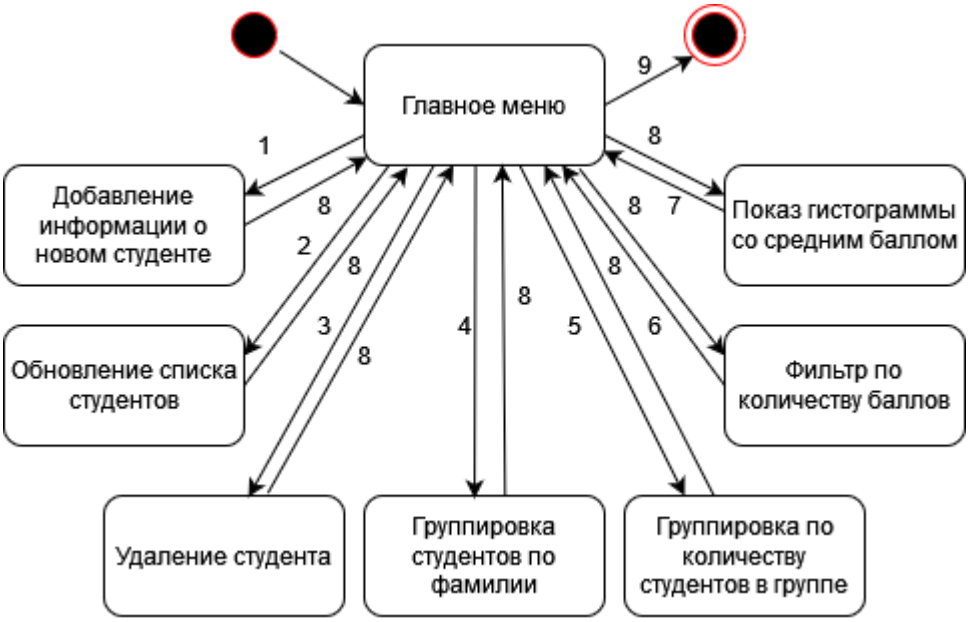

Рисунок 7 - диаграмма состояний

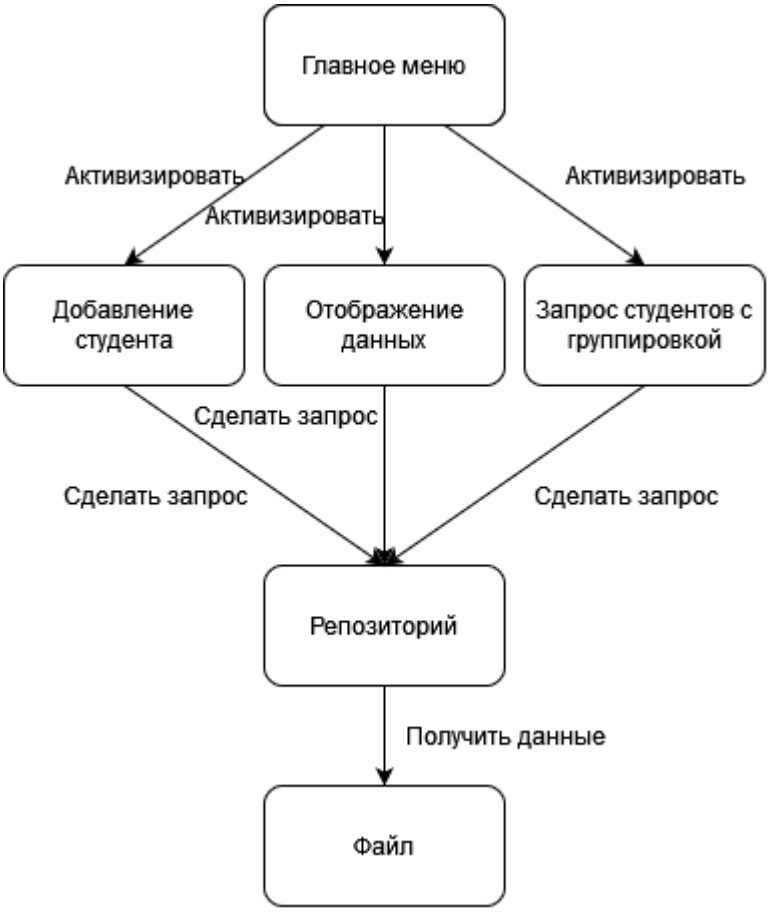

Рисунок 8 - объектная декомпозиция

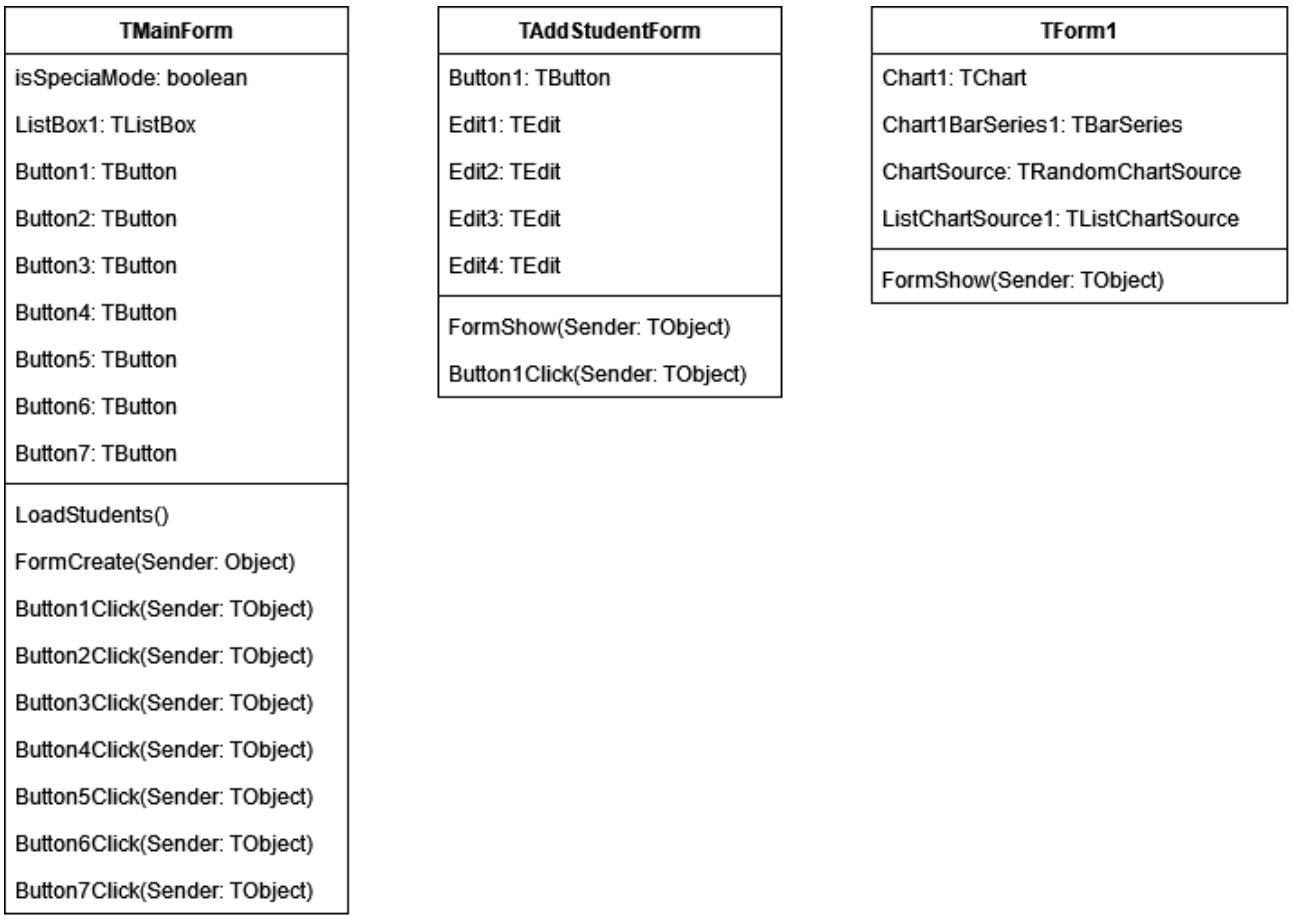

Рисунок 9 - объектная декомпозиция интерфейса

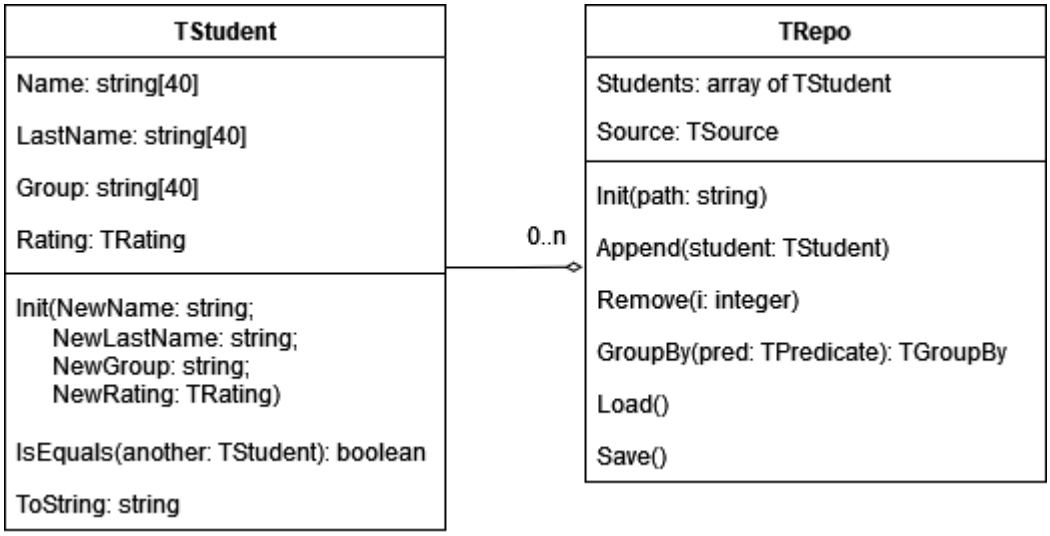

Рисунок 10 - объектная декомпозиция предметной области

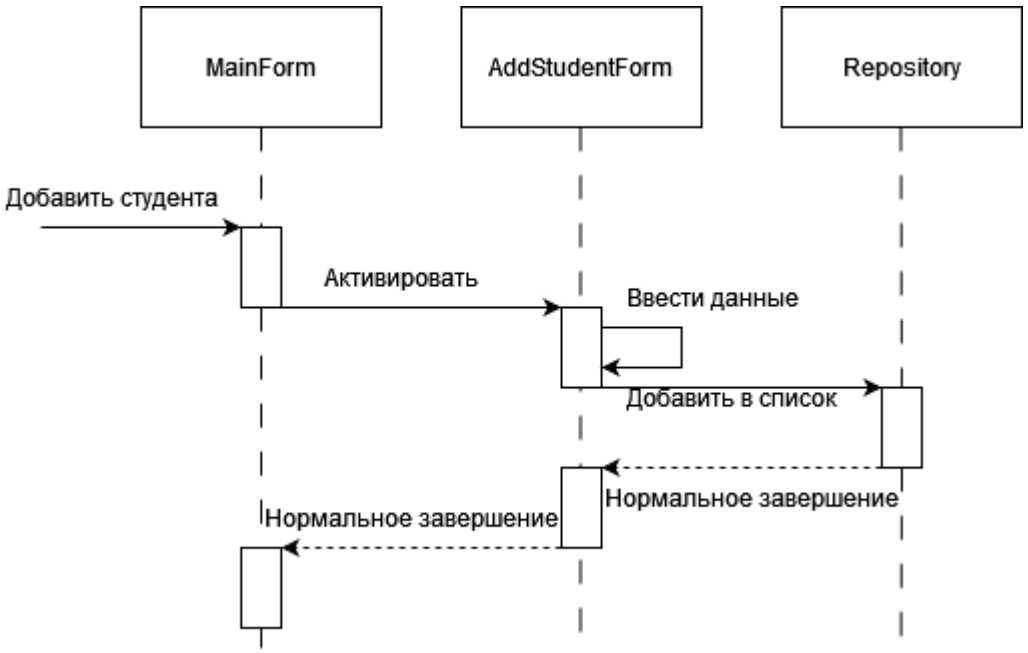

Рисунок 11 - диаграмма последовательности действий при добавлении нового студента

Код приложения

```
unit domain;

{$mode ObjFPC}{$H+}

interface

uses
  Classes, SysUtils;

type
  TRating = 0..100;

  TStudent = object
  public
    Name: string[40];
    LastName: string[40];
    Group: string[40];
    Rating: TRating;
  public
    constructor Init(
      NewName: string;
      NewLastName: string;
      NewGroup: string;
      NewRating: TRating);
    function IsEquals(another: TStudent): boolean;
    function ToString: string;
  end;

  TGroupBy = array of array of TStudent;
  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

```
unit repository;

{$mode delphi}{$H+}

interface

uses
  Classes, SysUtils, Domain;

type
  TSource = file of TStudent;

  TRepo = object
  public
    Students: array of TStudent;
    Source: TSource;
  public
    constructor Init(path: string);
    procedure Append(student: TStudent);
    procedure Remove(i: integer);
    function GroupBy(pred: TPredicate): TGroupBy;
    procedure Load;
    procedure Save;
end;

var Repo: TRepo;
```

Рисунок 14 - код модуля репозитория

```
implementation

constructor TRepo.Init(path: string);
begin
  if not FileExists(path) then
    FileCreate(path);
  AssignFile(Source, path);
end;

procedure TRepo.Append(student: TStudent);
begin
  SetLength(Students, Length(Students)+1);
  Students[Length(Students)-1] := student;
  Save;
end;

procedure TRepo.Remove(i: integer);
begin
  if (Length(Students) > 0) and (i > -1) then begin
    Students[i] := Students[Length(Students)-1];
    SetLength(Students, Length(Students)-1);
  end;
  Save;
end;

function TRepo.GroupBy(pred: TPredicate): TGroupBy;
var i, j: integer;
    found: boolean;
begin
  for i := 0 to High(Students) do begin
    found := false;
    j := 0;
    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;
      end;
      j += 1;
    end;
    if not found then begin
      j := Length(Result);
      SetLength(Result, j+1);
      SetLength(Result[j], 1);
      Result[j][0] := Students[i];
    end;
  end;
end;
```

Рисунок 15 - код модуля репозитория

```
procedure TRepo.Load;
var length, i: integer;
begin
  reset(Source);
  length := FileSize(Source);
  SetLength(Students, length);
  i := 0;
  while not eof(Source) do begin
    Read(Source, Students[i]);
    i += 1;
  end;
  close(Source);
end;

procedure TRepo.Save;
var i: integer;
begin
  rewrite(Source);
  for i := 0 to High(Students) do begin
    Write(Source, Students[i]);
  end;
  close(Source);
end;

end.
```

Рисунок 16 - код модуля репозитория

```
unit main;

{$mode delphi}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Repository, Domain, AddStudent, Gist;

type

  { TMainForm }

  TMainForm = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Button5: TButton;
    Button6: TButton;
    Button7: TButton;
    ListBox1: TListBox;
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button6Click(Sender: TObject);
    procedure Button7Click(Sender: TObject);
    procedure LoadStudents;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    isSpecialMode: boolean;
  public

  end;

var
  MainForm: TMainForm;

```

Рисунок 17 - код модуля main

```
implementation

{$R *.lfm}

{ TMainForm }

procedure TMainForm.LoadStudents;
var
  i: integer;
begin
  ListBox1.Clear;
  for i := 0 to High(Repo.Students) do
  begin
    ListBox1.Items.Add(Repo.Students[i].ToString);
  end;
  isSpecialMode := False;
end;

procedure TMainForm.FormCreate(Sender: TObject);
begin
  Repo.Init('students.dat');
  Repo.Load;
end;

procedure TMainForm.Button1Click(Sender: TObject);
begin
  AddStudentForm.Show;
end;

procedure TMainForm.Button2Click(Sender: TObject);
begin
  if not isSpecialMode then
  begin
    Repo.Remove(ListBox1.ItemIndex);
    LoadStudents;
  end;
end;

procedure TMainForm.Button3Click(Sender: TObject);
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;

125 procedure TMainForm.Button6Click(Sender: TObject);
    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;

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

Рисунок 19 - код модуля main

```
unit addStudent;

{$mode ObjFPC}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Domain, repository;

type

  { TAddStudentForm }

  TAddStudentForm = class(TForm)
    Button1: TButton;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    procedure Button1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
  private

  public

  end;

var
  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

```
unit gist;

{$mode delphi}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, TAGraph, TASources,
  TASeries, TACustomSource, TADbSource, Repository, Domain;

type

  { TForm1 }

  TForm1 = class(TForm)
    Chart1: TChart;
    Chart1BarSeries1: TBarSeries;
    ChartSource: TRandomChartSource;
    ListChartSource1: TListChartSource;
    procedure FormShow(Sender: TObject);
    function ListChartSource1Compare(AItem1, AItem2: Pointer): integer;
  private

  public

  end;

var
  GistForm: TForm1;

implementation
{$R *.lfm}

{ TForm1 }

procedure TForm1.FormShow(Sender: TObject);
var groupmates: TGroupBy;
    i, j: integer;
    avr: double;
begin
  groupmates := Repo.GroupBy(isGroupmates);
  ListChartSource1.Clear;
  for i := 0 to High(groupmates) do
  begin
    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);
  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.

Задание 2. Создание программной системы с элементарным интерфейсом консольного режима на C++

Задание

Выполнить структурную декомпозицию, разработать структурную схему, содержащую не менее 3 подпрограмм, и алгоритмы этих подпрограмм. Реализовать на C++ в консольном режиме. Предусмотреть примитивный интерфейс типа меню, позволяющий выбирать нужную подпрограмму.

Разработать программу, которая реализует операции над матрицами. Реализовать следующие операции: ввод матрицы, умножение матрицы на число, получение верней треугольной и нижней треугольной матриц из заданной матрицы, а также вывод результатов операций на экран.

Цель

Получить практические навыки разработки cli-приложений на языке C++.

Проект программы

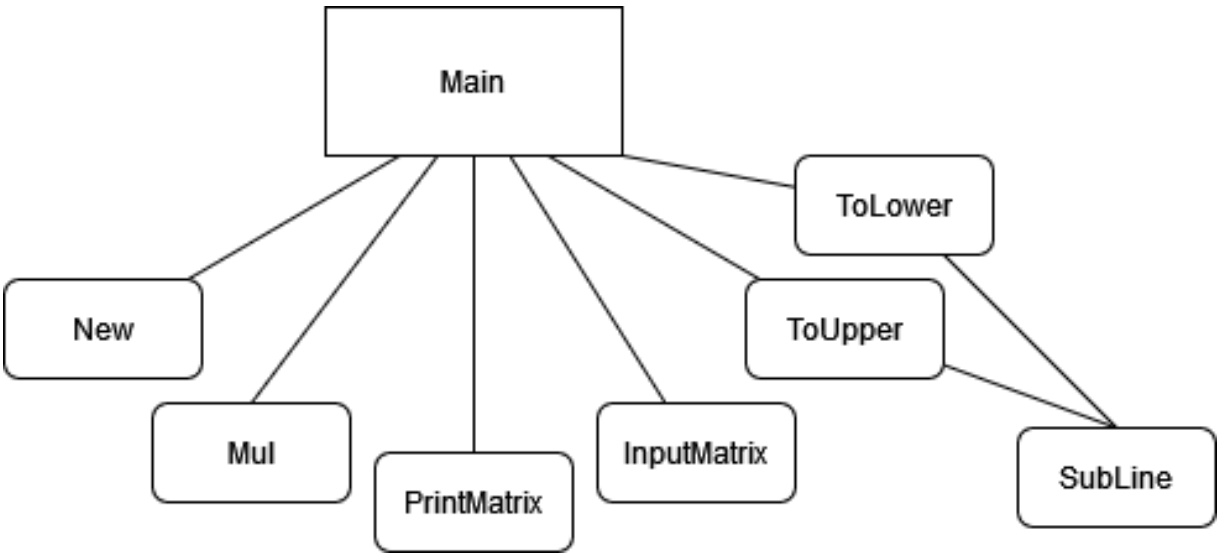

Рисунок 23 - структурная схема

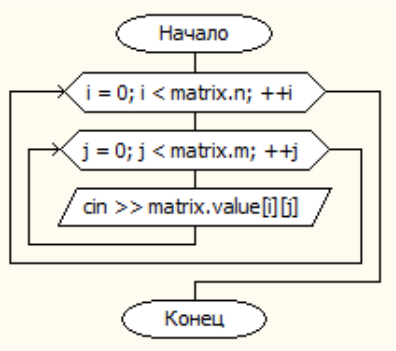

Рисунок 24 - схема алгоритма InputMatrix

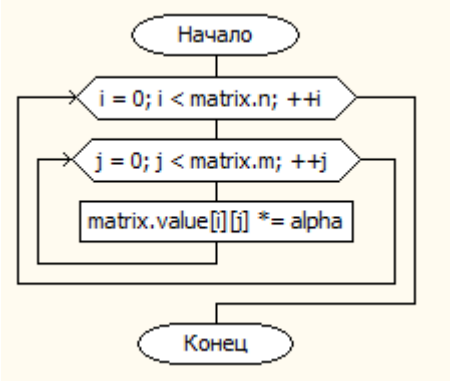

Рисунок 25 - схема алгоритма Mul

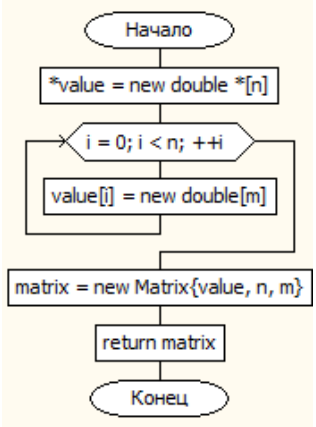

Рисунок 26 - схема алгоритма New

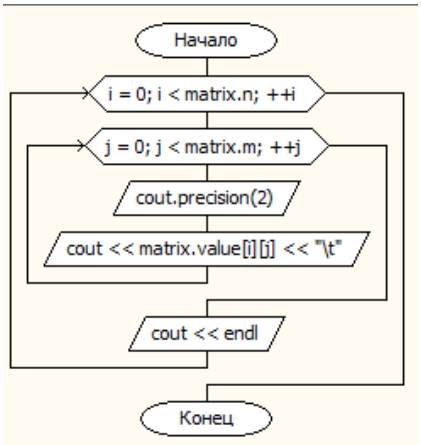

Рисунок 27 - схема алгоритма PrintMatrix

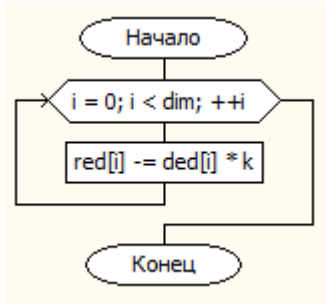

Рисунок 28 - схема алгоритма SubLine

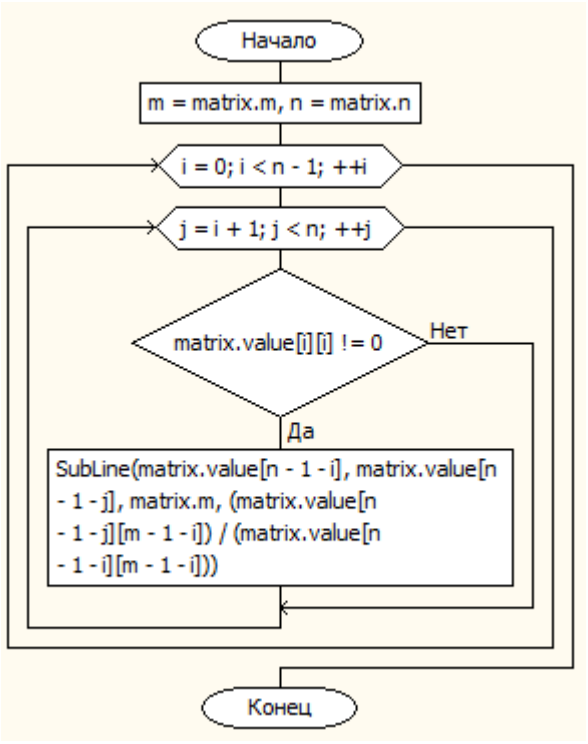

Рисунок 29 - схема алгоритма ToLower

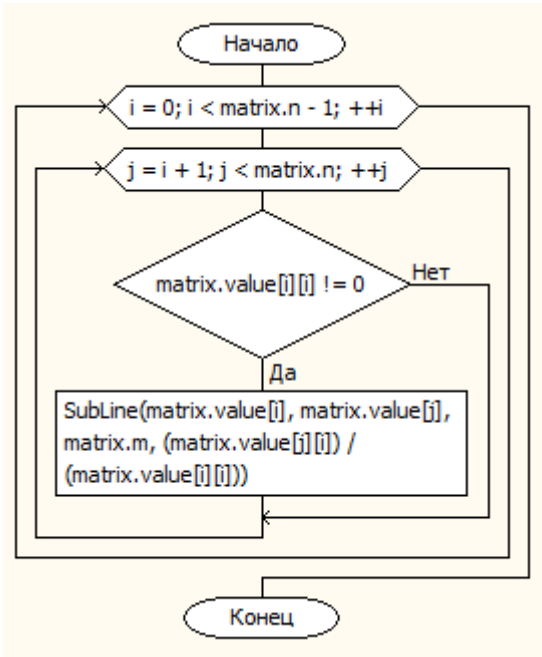

Рисунок 30 - схема алгоритма ToUpper

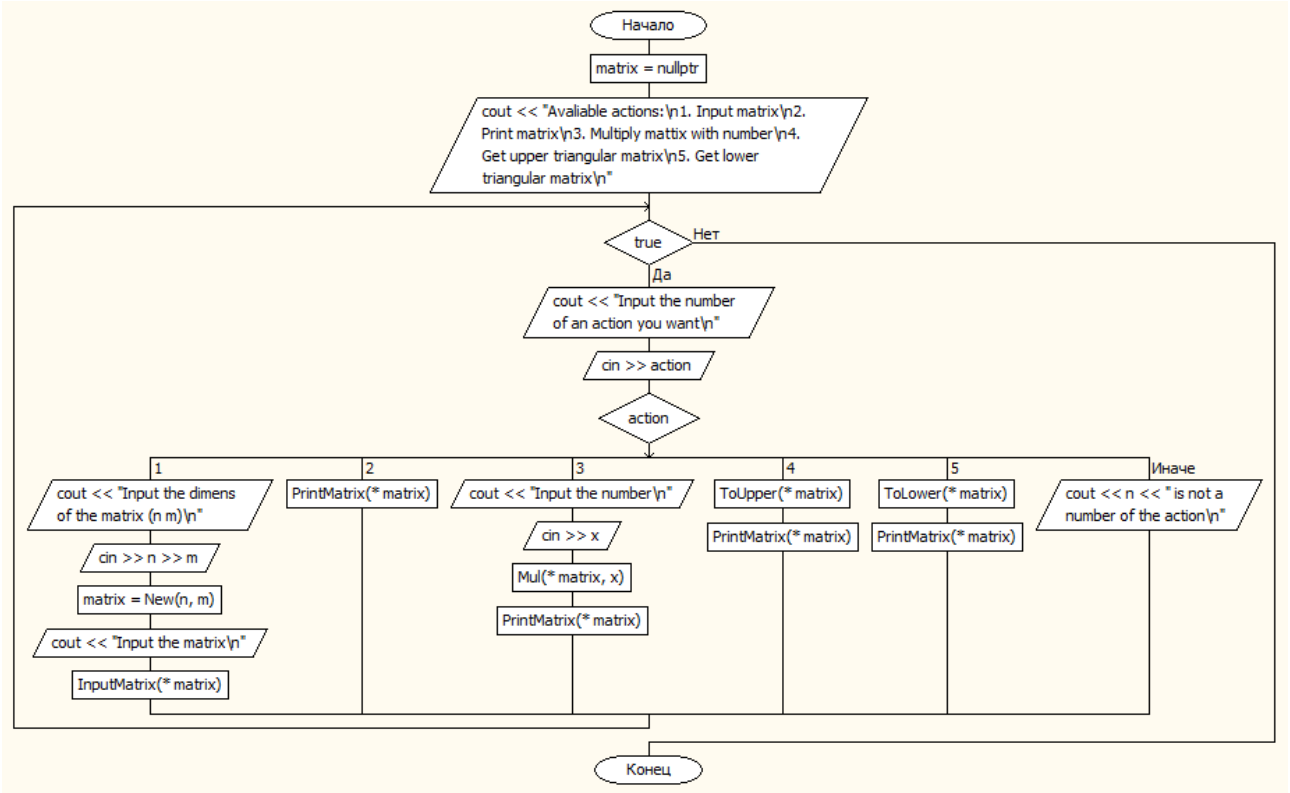

Рисунок 31 - схема алгоритма Main

Код программы

```cpp
#include <iostream>

using namespace std;

typedef struct Matrix {
    double** value;
    int n;
    int m;
} Matrix;

Matrix* New(int n, int m) {
    double** value = new double*[n];
    for (size_t i = 0; i < n; ++i) {
        value[i] = new double[m];
    }
    Matrix* matrix = new Matrix{
        value, n, m
    };
    return matrix;
}

void Mul(Matrix& matrix, double alpha) {
    for (size_t i = 0; i < matrix.n; ++i) {
        for (size_t j = 0; j < matrix.m; ++j) {
            matrix.value[i][j] *= alpha;
        }
    }
}
```

Рисунок 32 - код программы

```
30  void PrintMatrix(Matrix& matrix) {
31      for (size_t i = 0; i < matrix.n; ++i) {
32          for (size_t j = 0; j < matrix.m; ++j) {
33              cout.precision(2);
34              cout << matrix.value[i][j] << "\t";
35          }
36          cout << endl;
37      }
38  }
39
40  void InputMatrix(Matrix& matrix) {
41      for (size_t i = 0; i < matrix.n; ++i) {
42          for (size_t j = 0; j < matrix.m; ++j) {
43              cin >> matrix.value[i][j];
44          }
45      }
46  }
47
48  void SubLine(double*& ded, double*& red, int dim, double k) {
49      for (size_t i = 0; i < dim; ++i) {
50          red[i] -= ded[i] * k;
51      }
52  }
53
54  void ToUpper(Matrix& matrix) {
55      for (size_t i = 0; i < matrix.n-1; ++i) {
56          for (size_t j = i+1; j < matrix.n; ++j) {
57              if (matrix.value[i][i] != 0) {
58                  SubLine(matrix.value[i],
59                          matrix.value[j],
60                          matrix.m,
61                          (matrix.value[j][i])/(matrix.value[i][i])
62                  );
63              }
64          }
65      }
66  }
```

Рисунок 33 - код программы

```
68  void ToLower(Matrix& matrix) {
69      int m = matrix.m, n = matrix.n;
70      for (size_t i = 0; i < n-1; ++i) {
71          for (size_t j = i+1; j < n; ++j) {
72              if (matrix.value[i][i] != 0) {
73                  SubLine(matrix.value[n-1-i],
74                          matrix.value[n-1-j],
75                          matrix.m,
76                          (matrix.value[n-1-j][m-1-i])/(matrix.value[n-1-i][m-1-i])
77                  );
78              }
79          }
80      }
81  }
```

Рисунок 34 - код программы

```cpp
int main() {
    Matrix* matrix = nullptr;
    int n, m;

    cout << "Avaliable actions:\n1. Input matrix\n2. Print matrix\n3." +
            "Multiply mattix with number\n4. Get upper triangular matrix\n5. Get lower triangular matrix\n";

    while(true) {
      cout << "Input the number of an action you want\n";
      int action;
      cin >> action;
      switch(action) {
        case 1:
          cout << "Input the dimens of the matrix (n m)\n";
          cin >> n >> m;
          matrix = New(n, m);
          cout << "Input the matrix\n";
          InputMatrix(*matrix);
          break;
        case 2:
          PrintMatrix(*matrix);
          break;
        case 3:
          double x;
          cout << "Input the number\n";
          cin >> x;
          Mul(*matrix, x);
          PrintMatrix(*matrix);
          break;
        case 4:
          ToUpper(*matrix);
          PrintMatrix(*matrix);
          break;
        case 5:
          ToLower(*matrix);
          PrintMatrix(*matrix);
          break;
        default:
          cout << n << " is not a number of the action\n";
      }
    }
}
```

Рисунок 35 - код программы

Тестирование программы

```
> ./main
Avaliable actions:
1. Input matrix
2. Print matrix
3. Multiply mattix with number
4. Get upper triangular matrix
5. Get lower triangular matrix
Input the number of an action you want
1
Input the dimens of the matrix (n m)
2 3
Input the matrix
1 2 3
4 5 6
Input the number of an action you want
2
1   2   3
4   5   6
Input the number of an action you want
3
Input the number
4
4   8   12
16  20  24
Input the number of an action you want
4
4   8   12
0   -12 -24
```

Рисунок 36 - пример работы программы

Вывод

Были получены практические навыки разработки cli-приложений на языке C++.

Задание 3. Создание программной системы с Qt интерфейсом на C++

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 в среде Visual Studio или QT Creator.

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

Разработать настольное приложение на языке C++ с использованием библиотеки Qt. Изучить основы работы с библиотекой графических интерфейсов Qt, а также средой разработки Qt Creator.

Проект программы

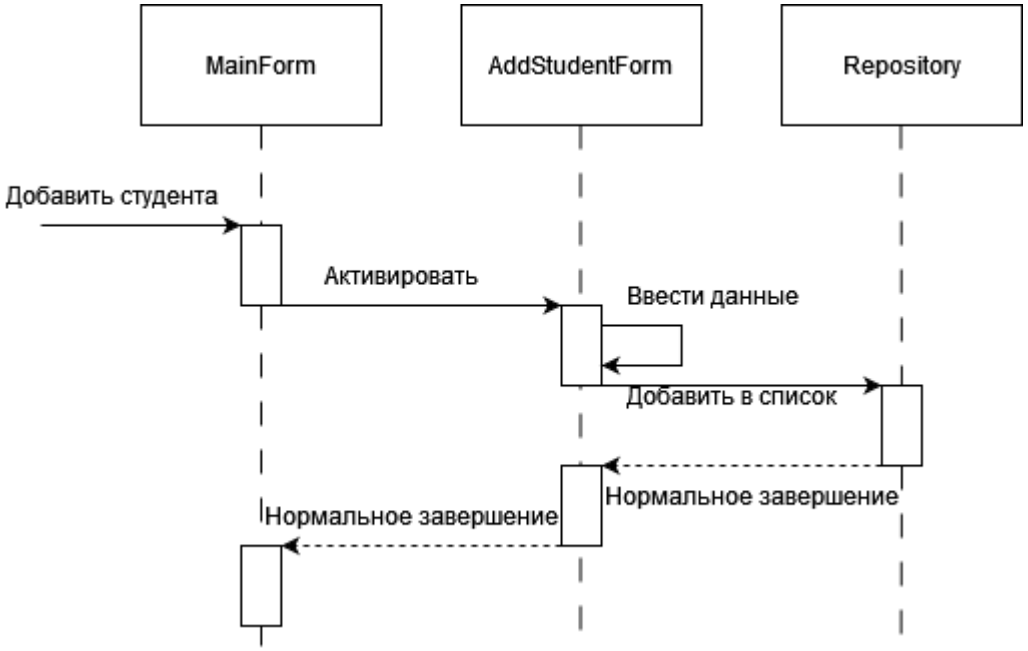

Рисунок 37 - диаграмма последовательности действий при добавлении нового пользователя

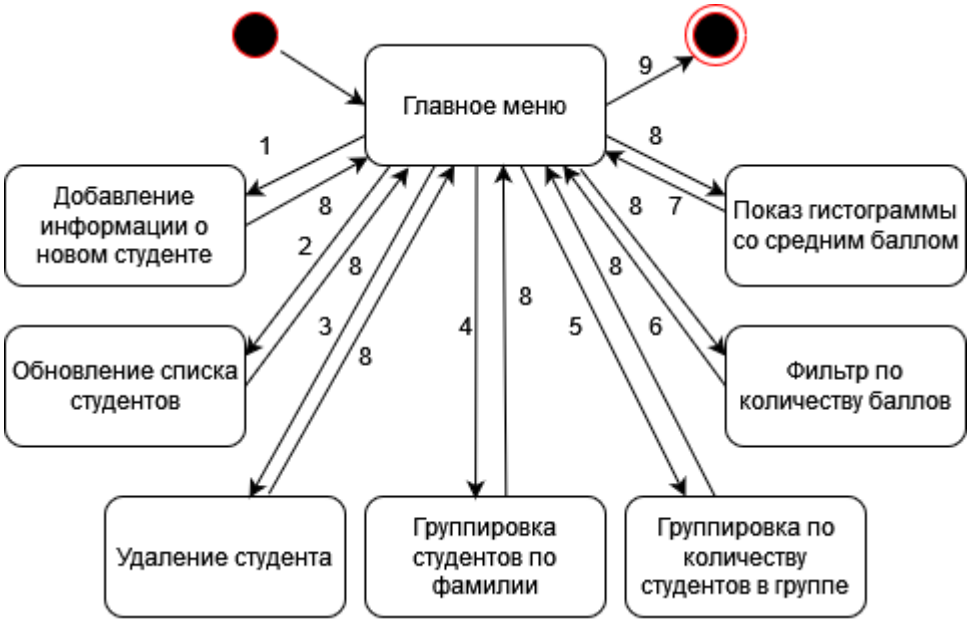

Рисунок 38 - диаграмма состояний

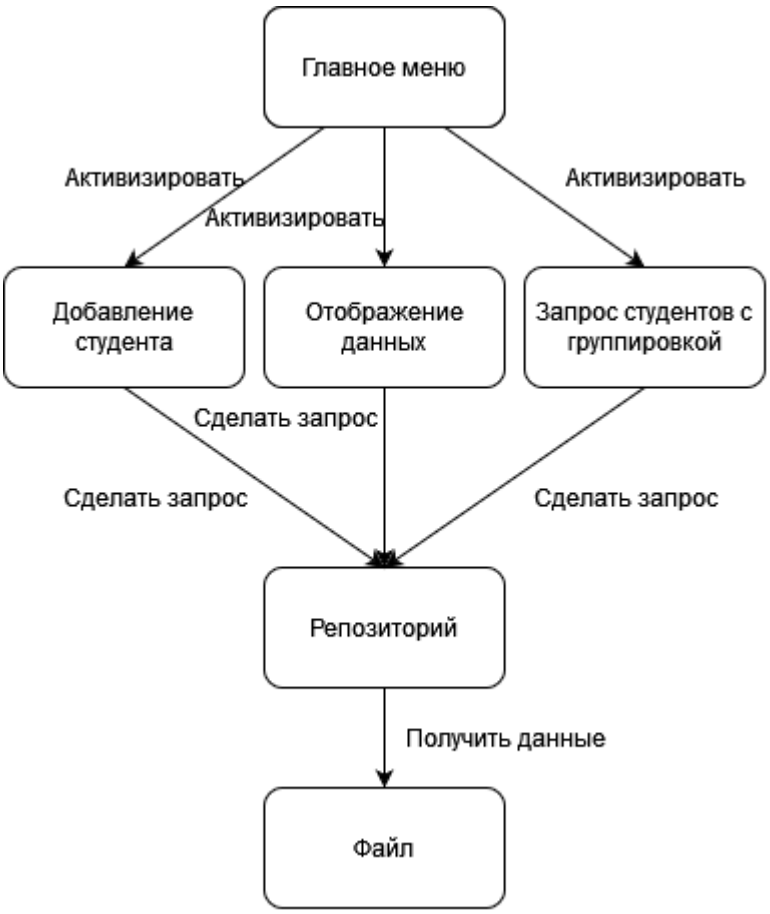

Рисунок 39 - объектная декомпозиция

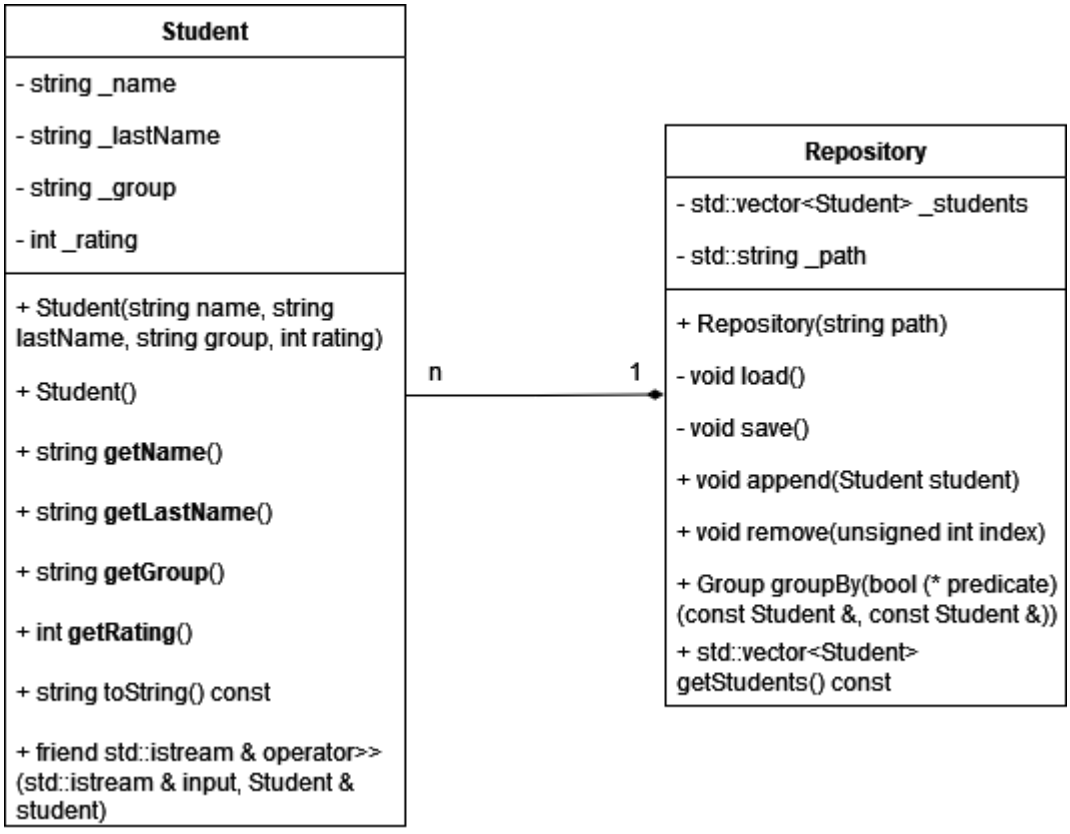

Рисунок 40 - диаграмма классов предметной области

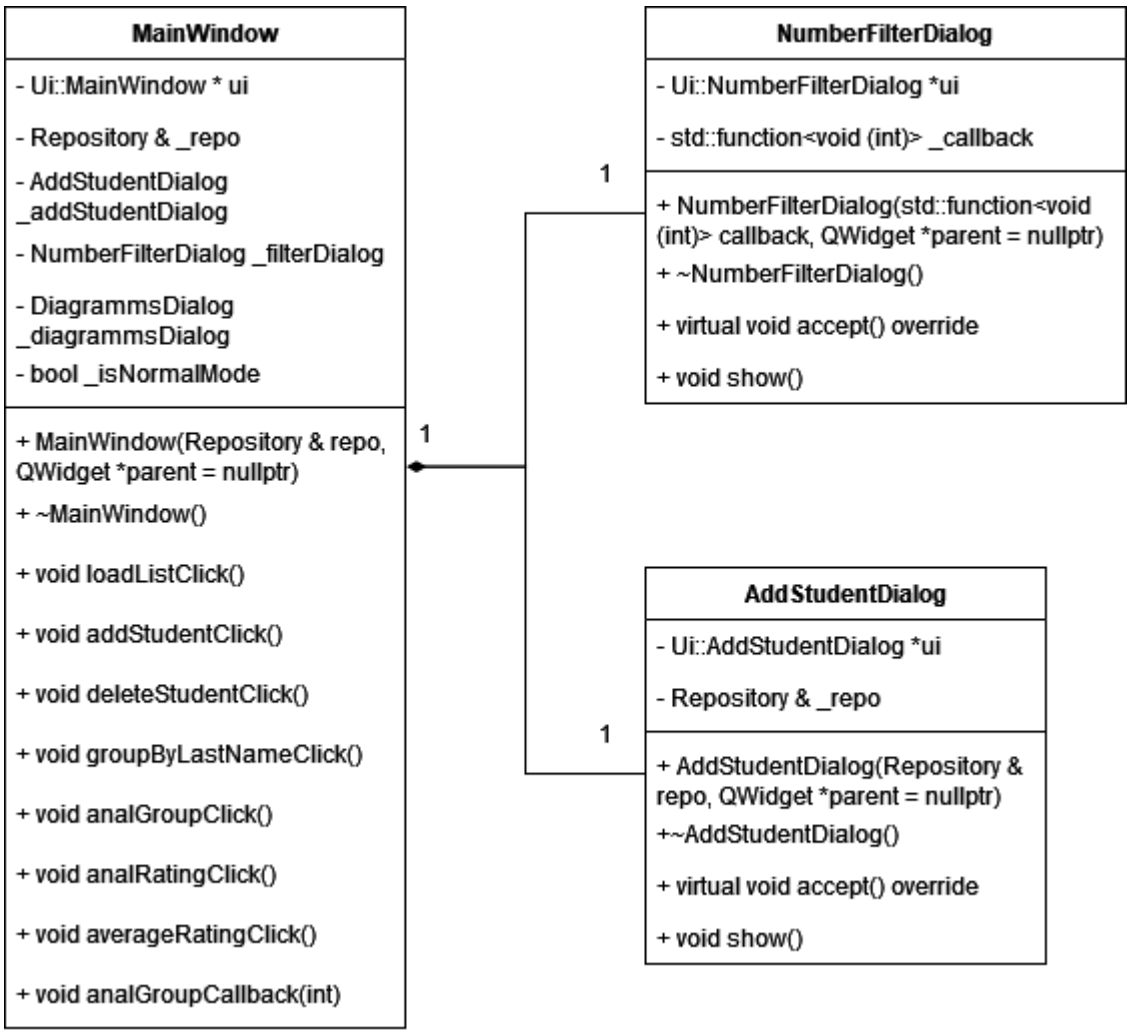

Рисунок 41 - диаграмма классов интерфейса

Интерфейс приложения

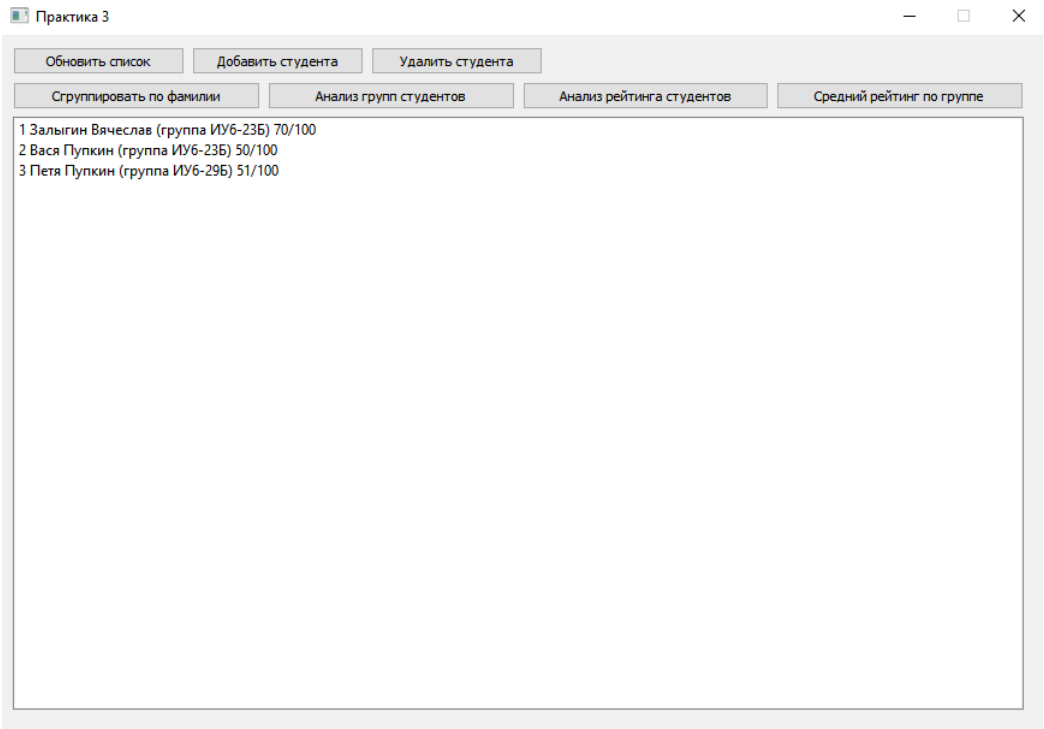

Рисунок 42 - главное меню

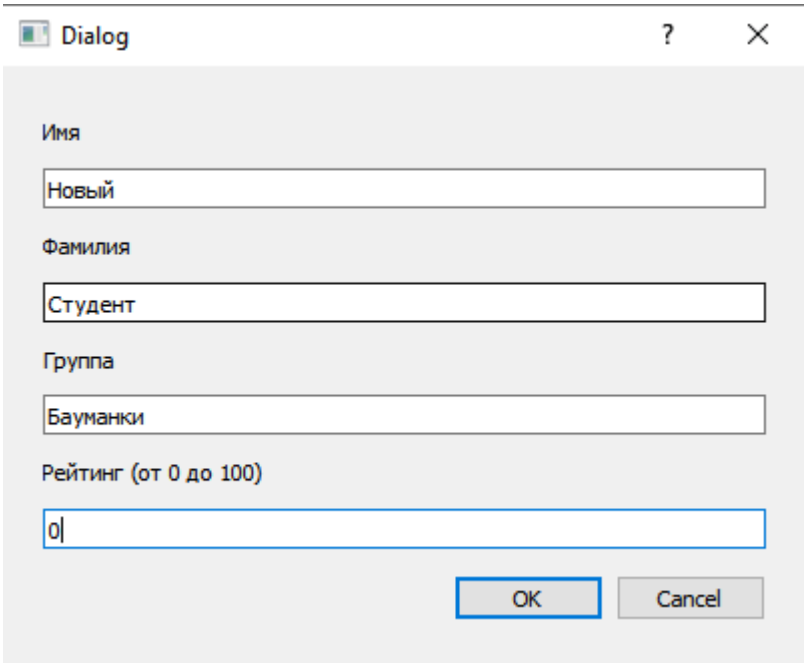

Рисунок 43 - диалог добавления нового студента

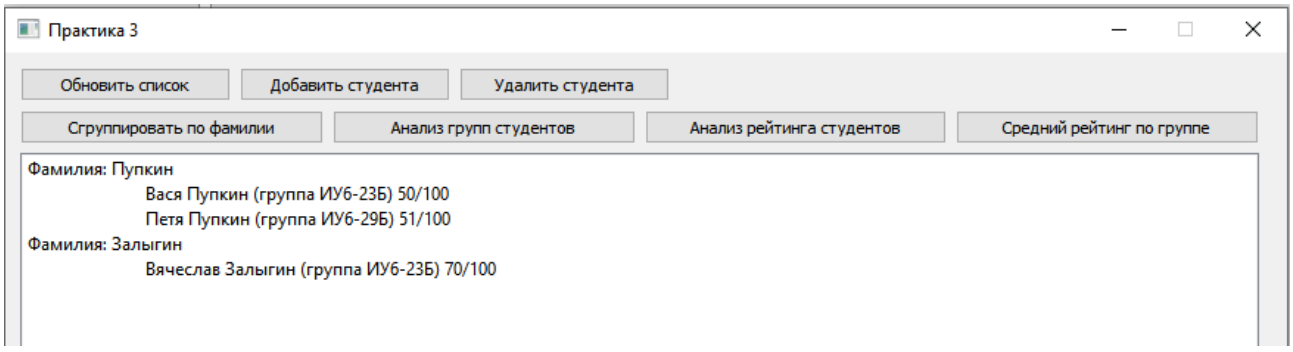

Рисунок 44 - режим группировки по фамилиям

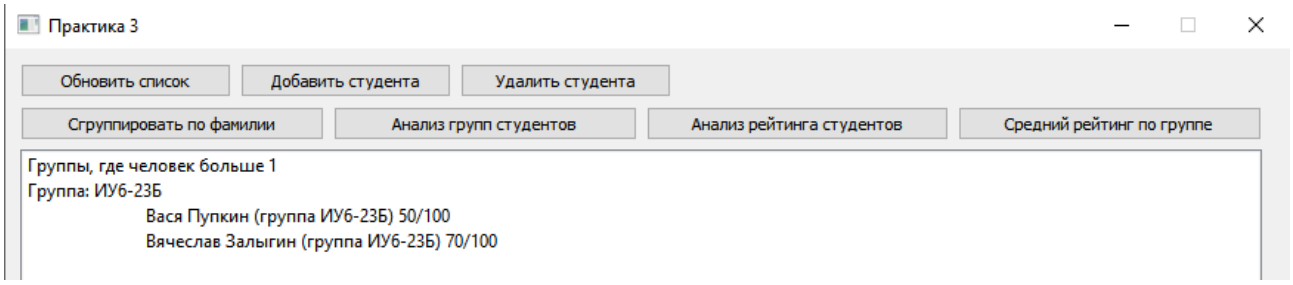

Рисунок 45 - режим фильтрации групп по количеству человек

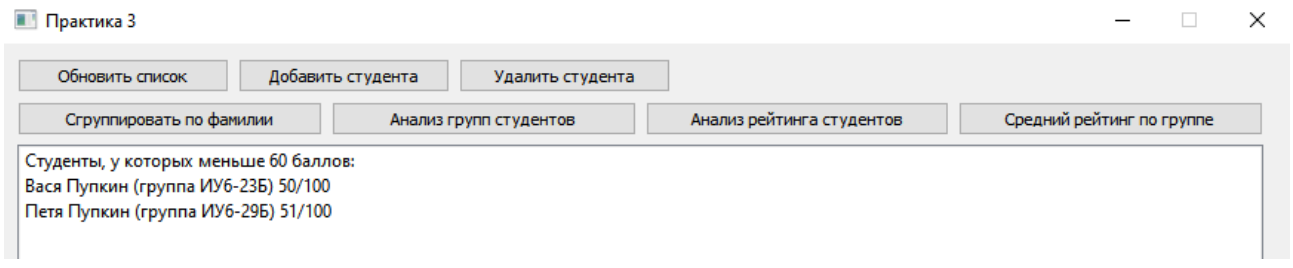

Рисунок 46 - режим фильтрации по рейтингу студента

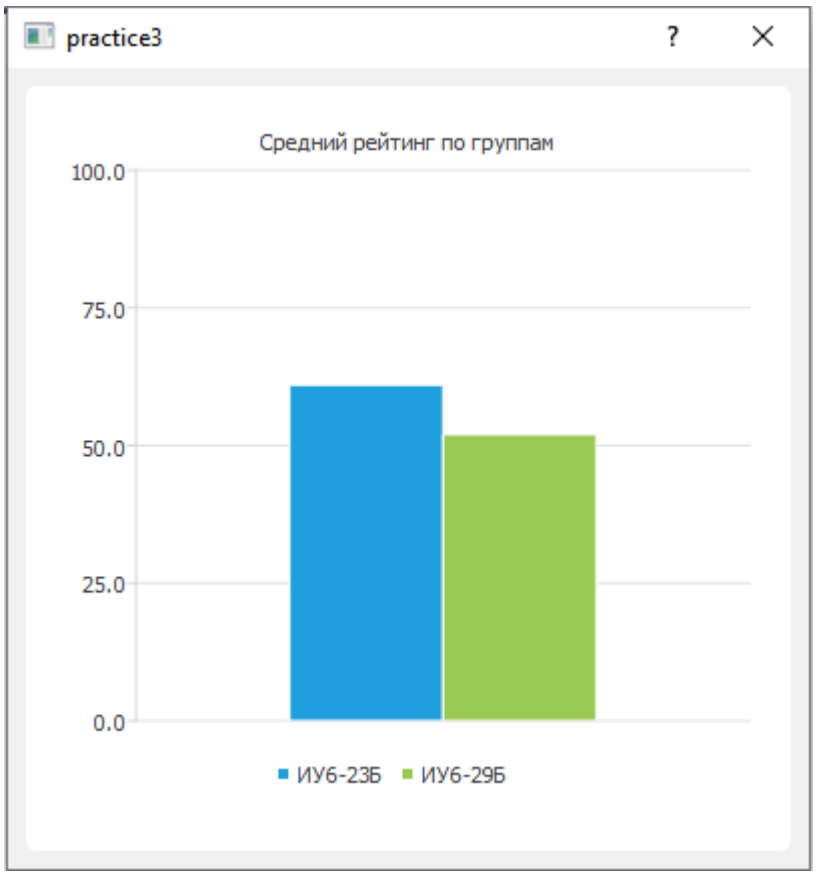

Рисунок 47 - диалог с диаграммой средних по группе баллов

Код программы

addstudentdialog.hpp

```
#ifndef ADDSTUDENTDIALOG_H
#define ADDSTUDENTDIALOG_H

#include <QDialog>
#include <sstream>
#include <memory>
#include "repository.hpp"

namespace Ui {
class AddStudentDialog;
}

class AddStudentDialog : public QDialog
{
    Q_OBJECT

public:
    explicit AddStudentDialog(Repository & repo, QWidget *parent = nullptr);
    ~AddStudentDialog();

    virtual void accept() override;
    void show();

private:
    Ui::AddStudentDialog *ui;
    Repository & _repo;
};

#endif // ADDSTUDENTDIALOG_H
```

Рисунок 48

diagrammsdialog.hpp | DiagrammsDialog

```hpp
#ifndef DIAGRAMMSDIALOG_H
#define DIAGRAMMSDIALOG_H

#include "repository.hpp"

#include <QDialog>
#include <QBarSet>
#include <QChart>
#include <QBarSeries>
#include <QValueAxis>
#include <QChartView>

using namespace QtCharts;

namespace Ui {
class DiagrammsDialog;
}

class DiagrammsDialog : public QDialog
{
    Q_OBJECT

public:
    DiagrammsDialog(Repository & repo, QWidget *parent = nullptr);
    ~DiagrammsDialog();

    void show();

private:
    QChartView * chartView;
    Repository & _repo;
};

#endif // DIAGRAMMSDIALOG_H
```

Рисунок 49

```hpp
#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(Repository & repo, QWidget *parent = nullptr);
    ~MainWindow() {
        delete ui;
    };

public slots:
    void loadListClick();
    void addStudentClick();
    void deleteStudentClick();
    void groupByLastNameClick();
    void analGroupClick();
    void analRatingClick();
    void averageRatingClick();
    void analGroupCallback(int);

private:
    Ui::MainWindow * ui;
    Repository & _repo;
    AddStudentDialog _addStudentDialog;
    NumberFilterDialog _filterDialog;
    DiagrammsDialog _diagrammsDialog;
    bool _isNormalMode;
};
#endif // MAINWINDOW_H
```

Рисунок 50

numberfilterdialog.hpp <No Symbols>

```cpp
#ifndef NUMBERFILTERDIALOG_H
#define NUMBERFILTERDIALOG_H

#include <QDialog>
#include <functional>

namespace Ui {
class NumberFilterDialog;
}

class NumberFilterDialog : public QDialog
{
    Q_OBJECT

public:
    explicit NumberFilterDialog(std::function<void (int)> callback, QWidget *parent = nullptr)
    ~NumberFilterDialog();

    virtual void accept() override;
    void show();
private:
    Ui::NumberFilterDialog *ui;

    std::function<void (int)> _callback;
};

#endif // NUMBERFILTERDIALOG_H
```

Рисунок 51

repository.hpp Repository

```cpp
#ifndef REPOSITORY_H
#define REPOSITORY_H

#include <vector>
#include <string>

#include <student.hpp>


typedef std::vector<std::vector<Student>> Group;

class Repository
{
private:
    std::vector<Student> _students;
    std::string _path;

    void load();
    void save();

public:
    Repository(string path)
        :_path(path) { load(); };

    void append(Student student);
    void remove(unsigned int index);
    Group groupBy(bool (* predicate)(const Student &, const Student &));
    std::vector<Student> getStudents() const { return _students; }
};
#endif // REPOSITORY_H
```

Рисунок 52

student.hpp | getName() -> std::string

```
#ifndef STUDENT_H
#define STUDENT_H

#include <string>
#include <fstream>

using std::string;

class Student
{
private:
    string _name;
    string _lastName;
    string _group;
    int _rating;
public:
    Student(string name, string lastName, string group, int rating)
        : _name(name), _lastName(lastName), _group(group), _rating(rating) {};

    Student()
        : _name("empty"), _lastName("empty"), _group("empty"), _rating(-1) {};

    string getName() const { return _name; }
    string getLastName() const { return _lastName; }
    string getGroup() const { return _group; }
    int getRating() const { return _rating; }

    string toString() const;

    friend std::istream & operator>>(std::istream & input, Student & student);
};

bool operator==(Student & left, Student & right);

std::istream & operator>>(std::istream & input, Student & student);
std::ostream & operator<<(std::ostream & output, const Student & student);

#endif // STUDENT_H
```

Рисунок 53

addstudentdialog.cpp | AddStudentDialog::show() -> void

```cpp
#include "addstudentdialog.hpp"
#include "ui_addstudentdialog.h"

#include <QMessageBox>

AddStudentDialog::AddStudentDialog(Repository & repo, QWidget *parent)
    : QDialog(parent), ui(new Ui::AddStudentDialog), _repo(repo)
{
    ui->setupUi(this);
}

AddStudentDialog::~AddStudentDialog()
{
    delete ui;
}

bool checkStr(std::string str) {
    for (const auto & e : str) {
        if (e == ' ') {
            return false;
        }
    }
    return true;
}

bool checkRating(std::string rating) {
    std::stringstream ss(rating);
    int a;
    if ((ss >> a).fail() || !(ss >> std::ws).eof())
        return false;
    return a >= 0 && a <= 100;
}

void AddStudentDialog::accept() {
    if (    checkStr(ui->nameEdit->text().toStdString()) &&
            checkStr(ui->lastNameEdit->text().toStdString()) &&
            checkStr(ui->groupEdit->text().toStdString()) &&
            checkRating(ui->ratingEdit->text().toStdString())) {
        int rating;
        std::stringstream ss(ui->ratingEdit->text().toStdString());
        ss >> rating;
        Student s(ui->nameEdit->text().toStdString(), ui->lastNameEdit->text().toStdString(), ui->groupEdit->text().toStdString(), rating);
        _repo.append(std::move(s));
        QDialog::accept();
    } else {
        QMessageBox msg(QMessageBox::Information, "Некорректные данные", "Некорректный формат данных.",QMessageBox::Ok);
        msg.exec();
    }
}

void AddStudentDialog::show()
{
    ui->nameEdit->clear();
    ui->lastNameEdit->clear();
    ui->groupEdit->clear();
    ui->ratingEdit->clear();
    QDialog::show();
}

```

Рисунок 54

diagrammsdialog.cpp | DiagrammsDialog::show() -> void

```cpp
#include "diagrammsdialog.hpp"

DiagrammsDialog::DiagrammsDialog(Repository & repo, QWidget *parent) :
    QDialog(parent), _repo(repo)
{
    resize(400, 400);
}

DiagrammsDialog::~DiagrammsDialog() { }

bool sameGroup(const Student & lhs, const Student & rhs) { return lhs.getGroup() == rhs.getGroup(); }

void DiagrammsDialog::show()
{
    QBarSeries * series = new QBarSeries(this);
    for (const auto & group : _repo.groupBy(sameGroup)) {
        int average = 0;
        for (const auto & student : group) { average += student.getRating(); }
        average /= group.size();
        average += 1;
        QBarSet * bar = new QBarSet(QString::fromStdString(group[0].getGroup()));
        *bar << average;
        series->append(bar);
    }
    QChart * chart = new QChart();
    chart->addSeries(series);
    chart->setTitle("Средний рейтинг по группам");
    chart->setAnimationOptions(QChart::SeriesAnimations);

    QValueAxis *axisY = new QValueAxis();
    axisY->setRange(0,100);
    chart->addAxis(axisY, Qt::AlignLeft);
    series->attachAxis(axisY);

    chart->legend()->setVisible(true);
    chart->legend()->setAlignment(Qt::AlignBottom);

    chartView = new QChartView(chart, this);
    chartView->setRenderHint(QPainter::Antialiasing);
    chartView->resize(400, 400);
    QDialog::show();
}

```

Рисунок 55

```cpp
#include "mainwindow.hpp"

#include <QApplication>

#include <repository.hpp>

int main(int argc, char *argv[])
{
    Repository repo("./data.dat");
    QApplication a(argc, argv);
    MainWindow w(repo);
    w.show();
    return a.exec();
}

```

Рисунок 56

mainwindow.cpp

```cpp
#include "mainwindow.hpp"
#include "ui_mainwindow.h"
#include "numberfilterdialog.hpp"

#include <QMessageBox>
#include <QString>
#include <vector>
#include <sstream>
#include <string>

MainWindow::MainWindow(Repository & repo, QWidget *parent)
    : QMainWindow(parent), ui(new Ui::MainWindow), _repo(repo), _addStudentDialog(_repo, this), _isNormalMode(false
      _filterDialog([this](int filter) { this->analGroupCallback(filter); }, this), _diagrammsDialog(_repo, this)
{
    ui->setupUi(this);
    connect(ui->addStudentButton, SIGNAL(clicked(bool)),        this, SLOT(addStudentClick()));
    connect(ui->loadListButton, SIGNAL(clicked(bool)),          this, SLOT(loadListClick()));
    connect(ui->deleteStudentButton, SIGNAL(clicked(bool)),     this, SLOT(deleteStudentClick()));
    connect(ui->groupByLastNameButton, SIGNAL(clicked(bool)),   this, SLOT(groupByLastNameClick()));
    connect(ui->analGroupButton, SIGNAL(clicked(bool)),         this, SLOT(analGroupClick()));
    connect(ui->analRatingButton, SIGNAL(clicked(bool)),        this, SLOT(analRatingClick()));
    connect(ui->averageRatingButton, SIGNAL(clicked(bool)),     this, SLOT(averageRatingClick()));
}

void MainWindow::loadListClick() {
    std::vector<Student> students = _repo.getStudents();
    ui->list->clear();
    for (unsigned int i = 0; i < students.size(); ++i) {
        ui->list->addItem(QString::fromStdString(std::to_string(i+1) + " " + students[i].toString()));
    }
    _isNormalMode = true;
}

void MainWindow::addStudentClick() {
    _addStudentDialog.show();
}

void MainWindow::deleteStudentClick() {
    if (_isNormalMode) {
        std::stringstream ss(ui->list->currentItem()->text().toStdString());
        int i;
        ss >> i;
        _repo.remove(i-1);
        loadListClick();
    }
}

bool equalsByLastName(const Student & lhs, const Student & rhs) { return lhs.getLastName() == rhs.getLastName(); }
```

Рисунок 57

numberfilterdialog.cpp

```cpp
#include "numberfilterdialog.hpp"
#include "ui_numberfilterdialog.h"

#include <QMessageBox>

#include <sstream>

NumberFilterDialog::NumberFilterDialog(std::function<void (int)> callback, QWidget *parent) :
    QDialog(parent), ui(new Ui::NumberFilterDialog), _callback(callback)
{
    ui->setupUi(this);
}

NumberFilterDialog::~NumberFilterDialog()
{
    delete ui;
}

bool parseFilter(std::string filter, int & result) {
    std::stringstream ss(filter);
    int a;
    if ((ss >> a).fail() || !(ss >> std::ws).eof())
        return false;
    result = a;
    return a >= 0;
}

void NumberFilterDialog::accept()
{
    int filter;
    if (parseFilter(ui->filterEdit->text().toStdString(), filter)) {
        _callback(filter);
        QDialog::accept();
    } else {
        QMessageBox msg(QMessageBox::Information, "Некорректные данные", "Некорректный формат данных.",QMessageBox::Ok);
        msg.exec();
    }
}

void NumberFilterDialog::show() {
    ui->filterEdit->clear();
    QDialog::show();
}
```

Рисунок 58

repository.cpp — Repository::load() -> void

```cpp
#include "repository.hpp"

#include <fstream>
void Repository::load() {
    Student s;
    _students.clear();

    std::ifstream in(_path, std::ios::in);
    while (in >> s) {
        _students.push_back(s);
    }
    in.close();
}

void Repository::save() {
    std::ofstream out(_path, std::ios::out | std::ios::trunc);
    for (const auto & student : _students) {
        out << student;
    }
    out.close();
}

void Repository::append(Student student) {
    _students.push_back(student);
    save();
}

void Repository::remove(unsigned int index) {
    _students.erase(_students.begin()+index);
    save();
}

Group Repository::groupBy(bool (*predicate)(const Student &, const Student &)) {
    Group res;

    for (const auto & student : _students) {
        bool isInGroup = false;
        for (auto & group : res) {
            if (predicate(student, group[0]) && !isInGroup) {
                group.push_back(Student(student));
                isInGroup = true;
            }
        }
        if (!isInGroup) {
            res.push_back({Student(student)});
        }
    }

    return res;
}
```

Рисунок 59

student.cpp — <Select Symbol>

```cpp
#include "student.hpp"

bool operator==(Student & left, Student & right) {
    return  (left.getName() == right.getName()) &&
            (left.getLastName() == right.getLastName()) &&
            (left.getGroup() == right.getGroup()) &&
            (left.getRating() == right.getRating());
}

string Student::toString() const {
    return _name + " " + _lastName + " (группа " + _group + ") " + std::to_string(_rating) + "/100";
}

std::istream & operator>>(std::istream & input, Student & student) {
    input >> student._name >> student._lastName >> student._group >> student._rating;
    return input;
}

std::ostream & operator<<(std::ostream & output, const Student & student) {
    output << student.getName() << " " << student.getLastName() << " " << student.getGroup() << " " << student.getRating() << " ";
    return output;
}

```

Рисунок 60

Вывод

Было разработано настольное приложение на языке C++ с использованием библиотеки Qt. Были изучены основы работы с библиотекой графических интерфейсов Qt, а также средой разработки Qt Creator.