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

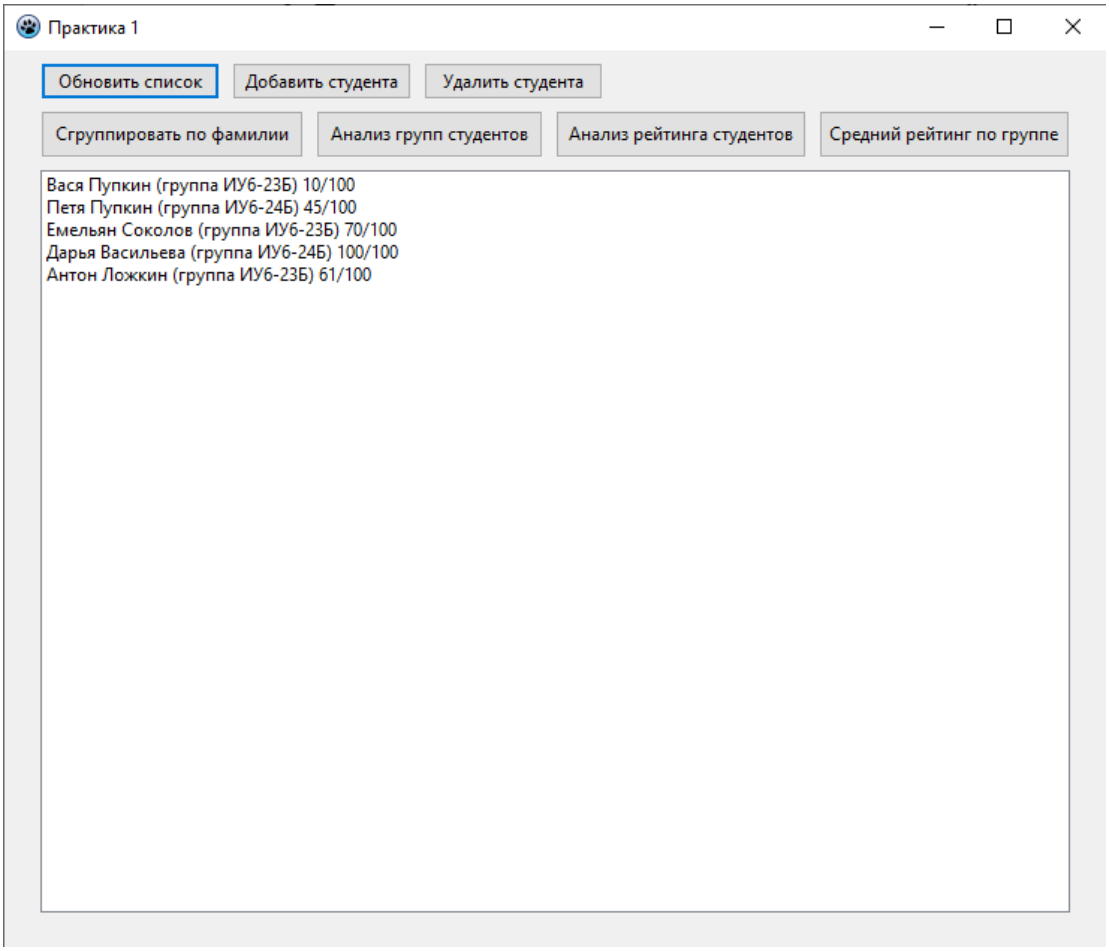

Рисунок 1 - главное меню программы

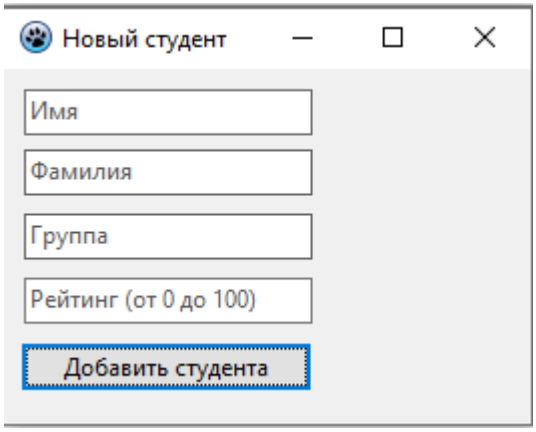

Рисунок 2 - форма добавления нового студента

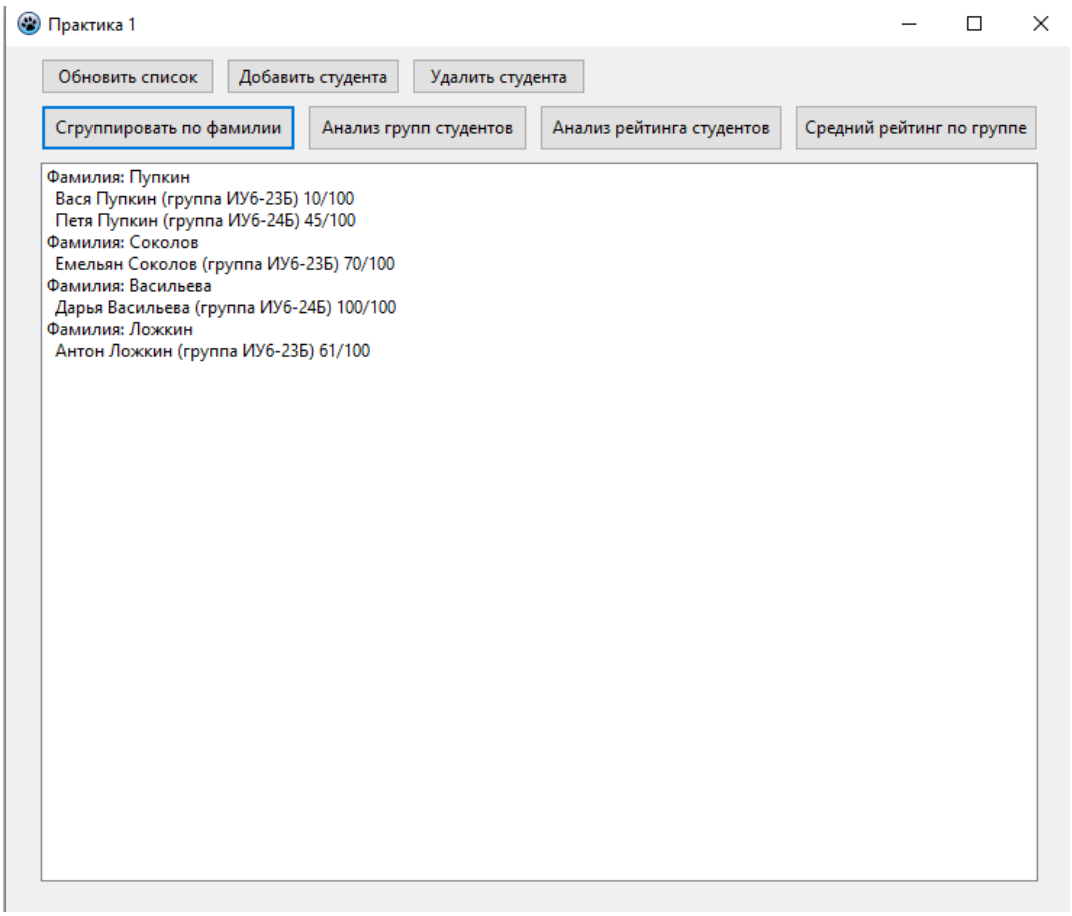

Рисунок 3 - режим группировки по фамилии

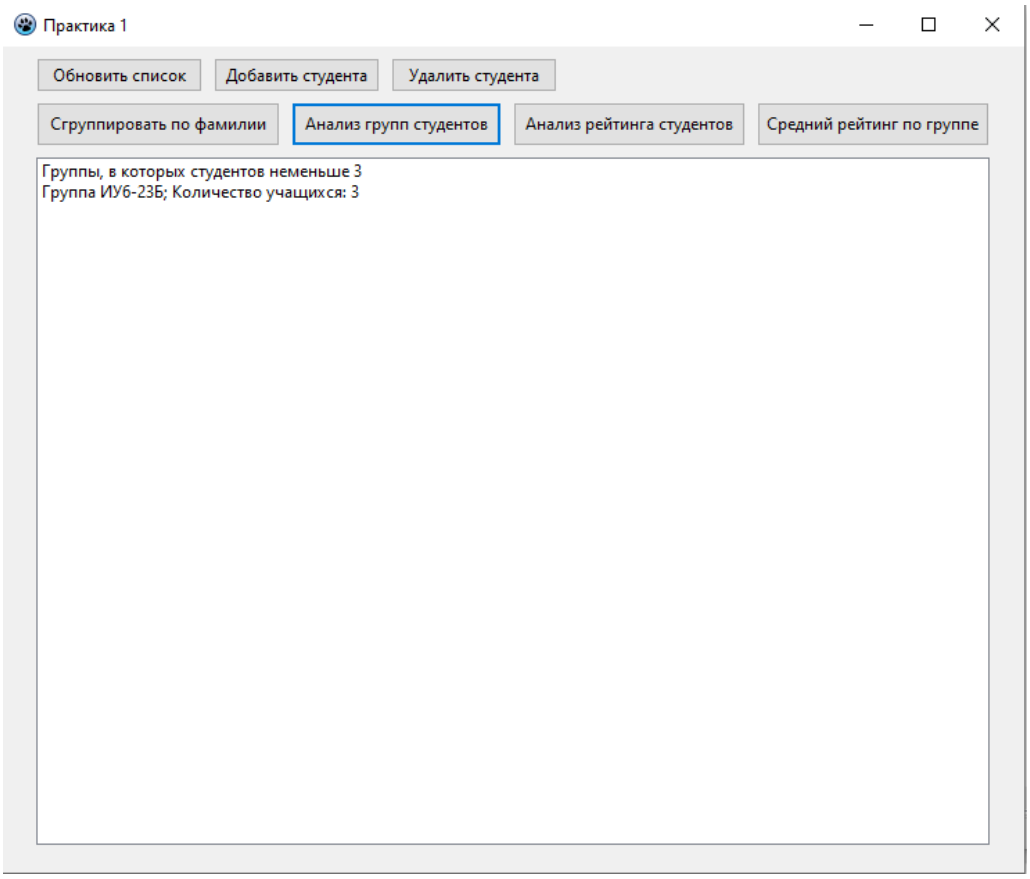

Рисунок 4 - группировка по количеству студентов в группе

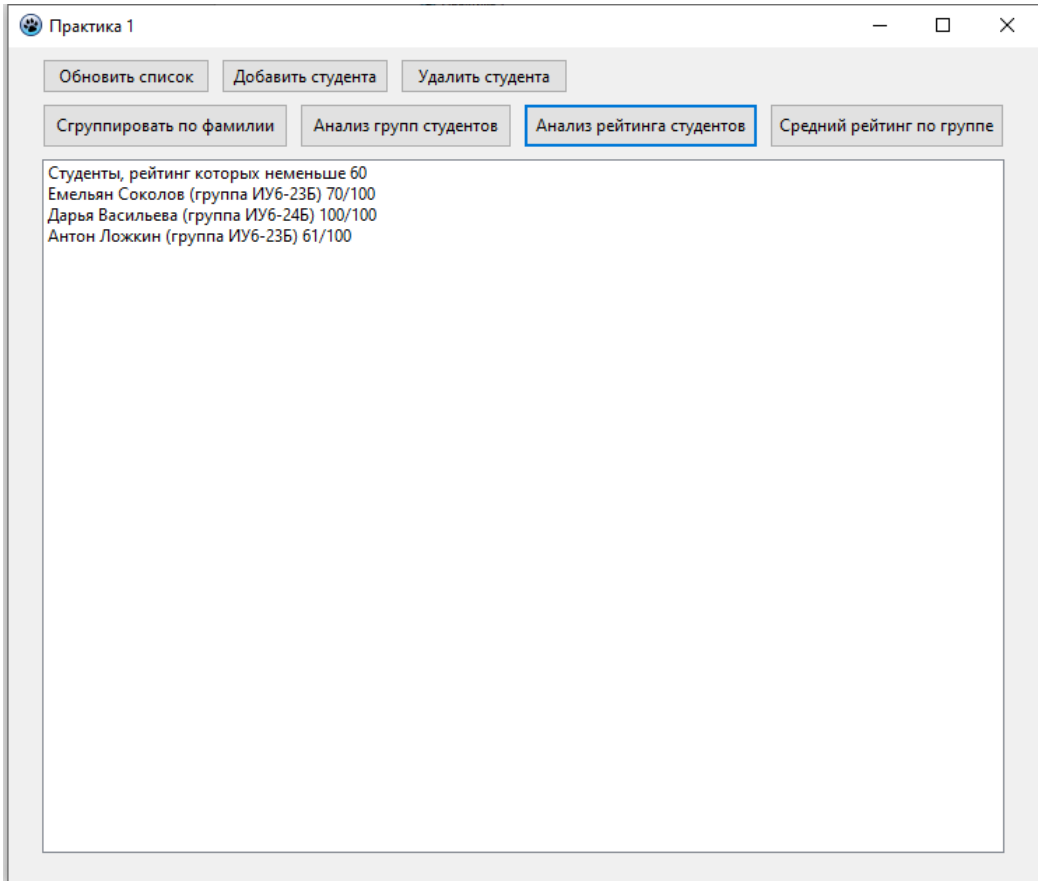

Рисунок 5 - фильтр по количеству баллов

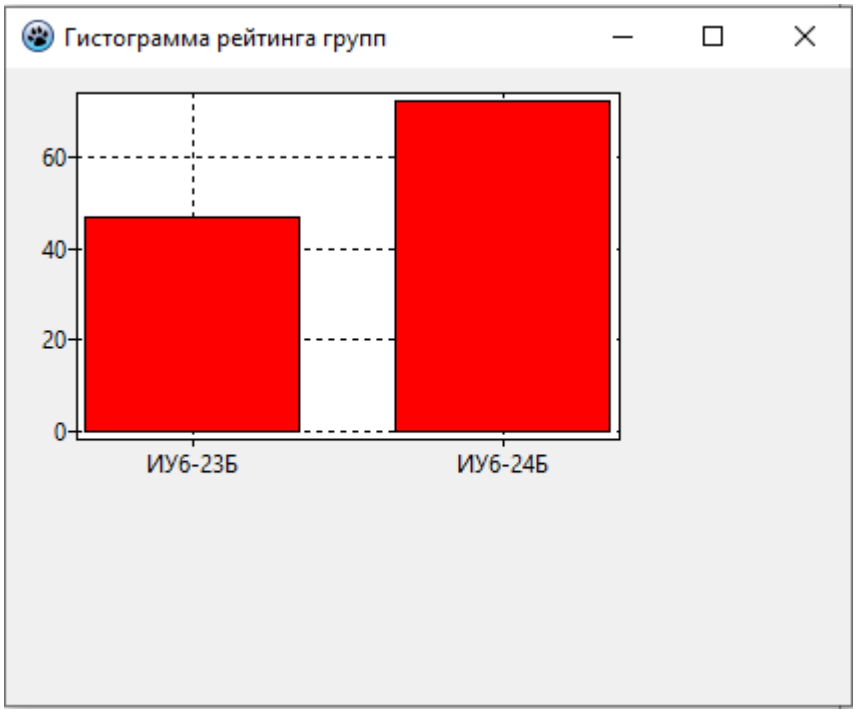

Рисунок 6 - гистограмма среднего рейтинга групп

Проект приложения

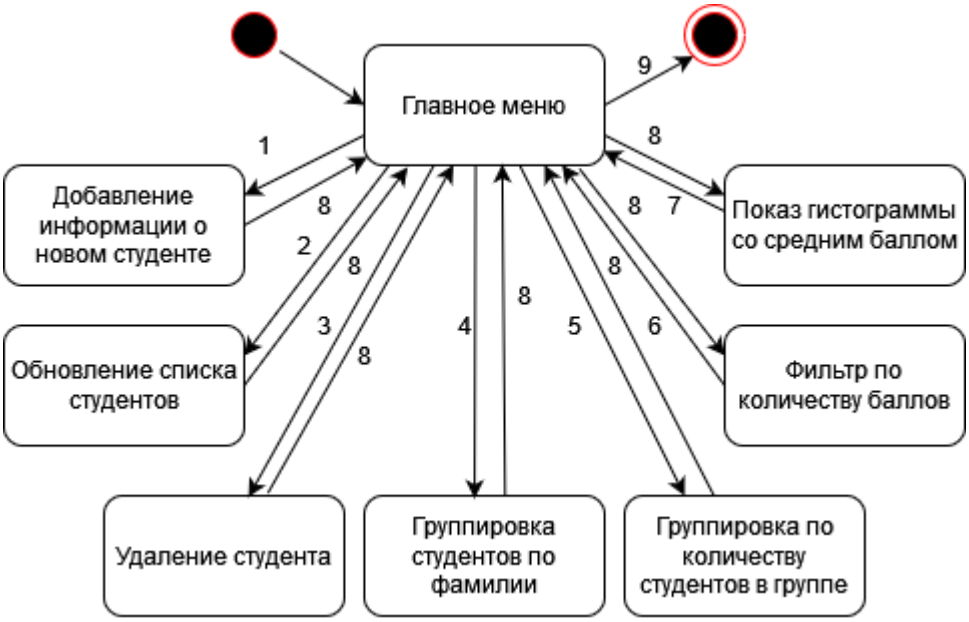

Рисунок 7 - диаграмма состояний

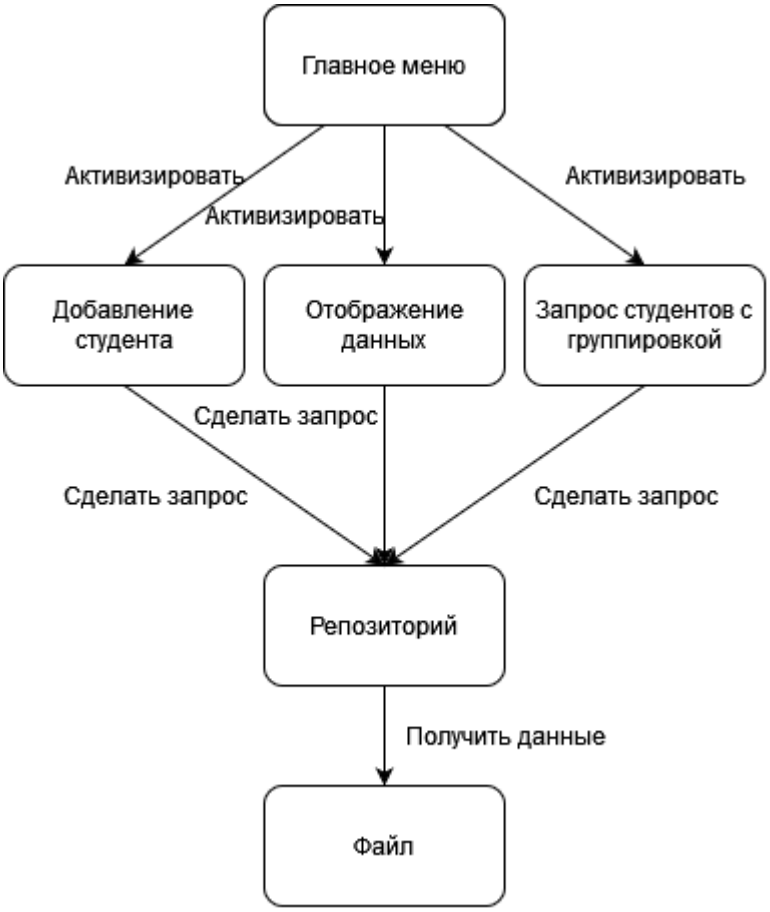

Рисунок 8 - объектная декомпозиция

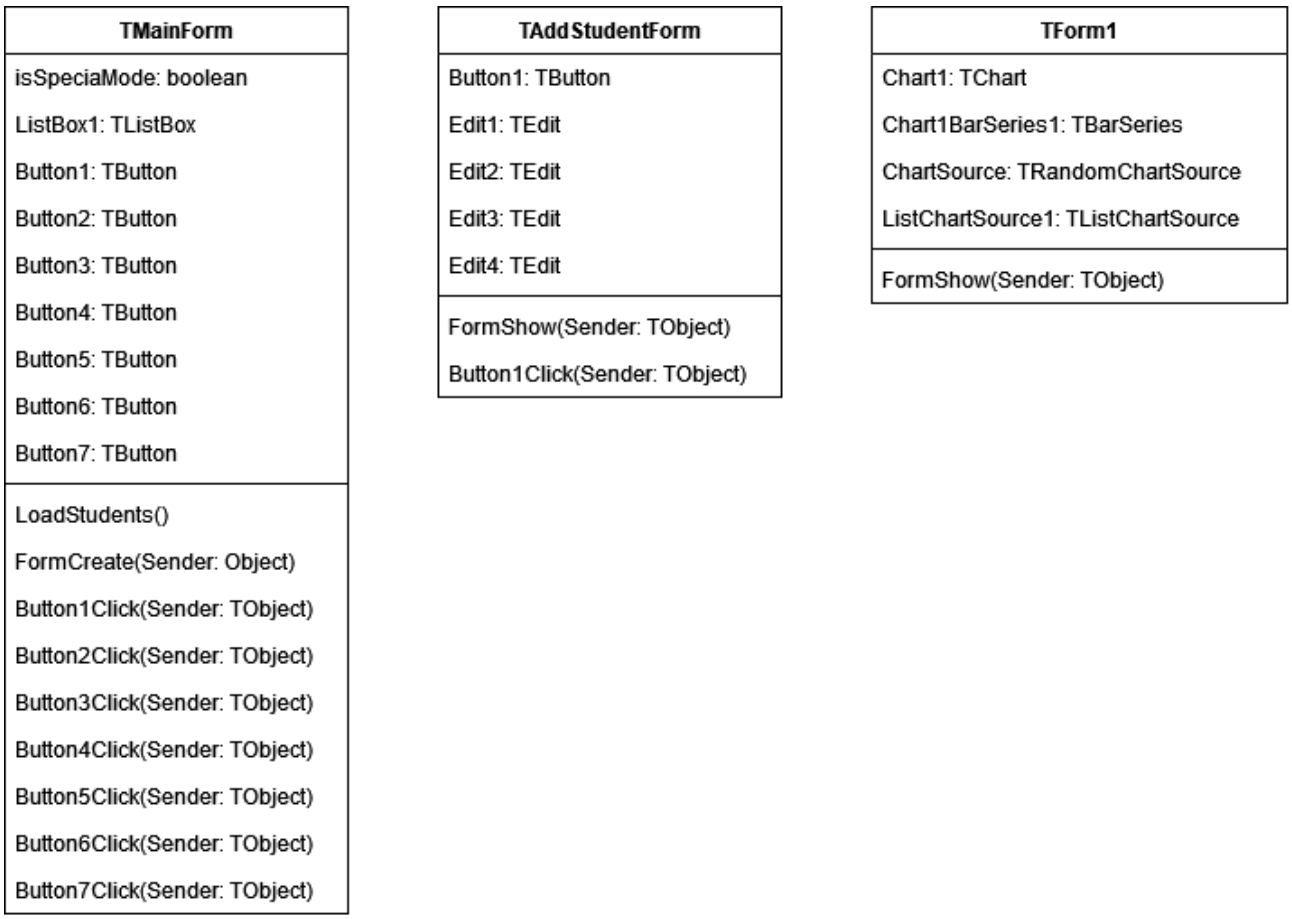

Рисунок 9 - объектная декомпозиция интерфейса

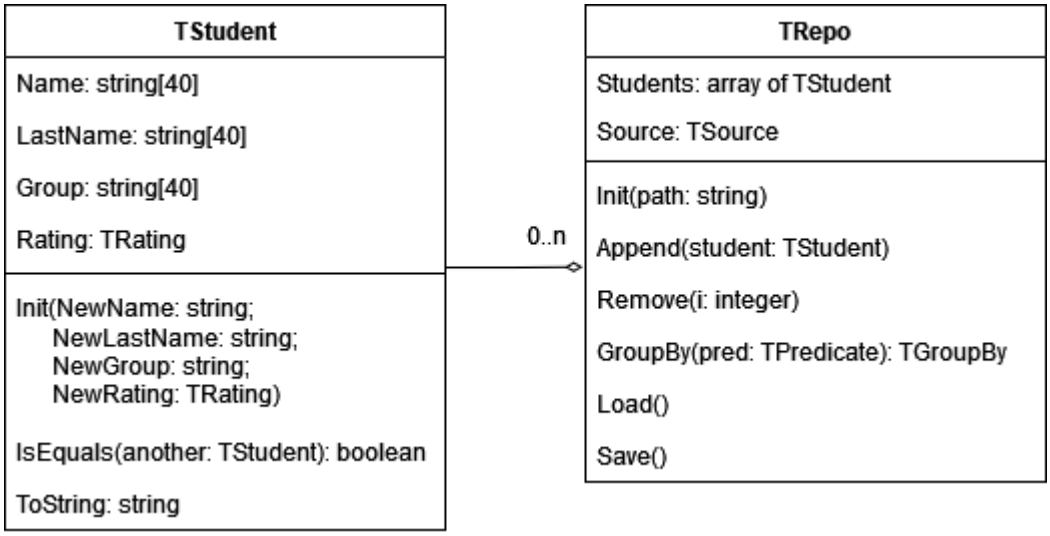

Рисунок 10 - объектная декомпозиция предметной области

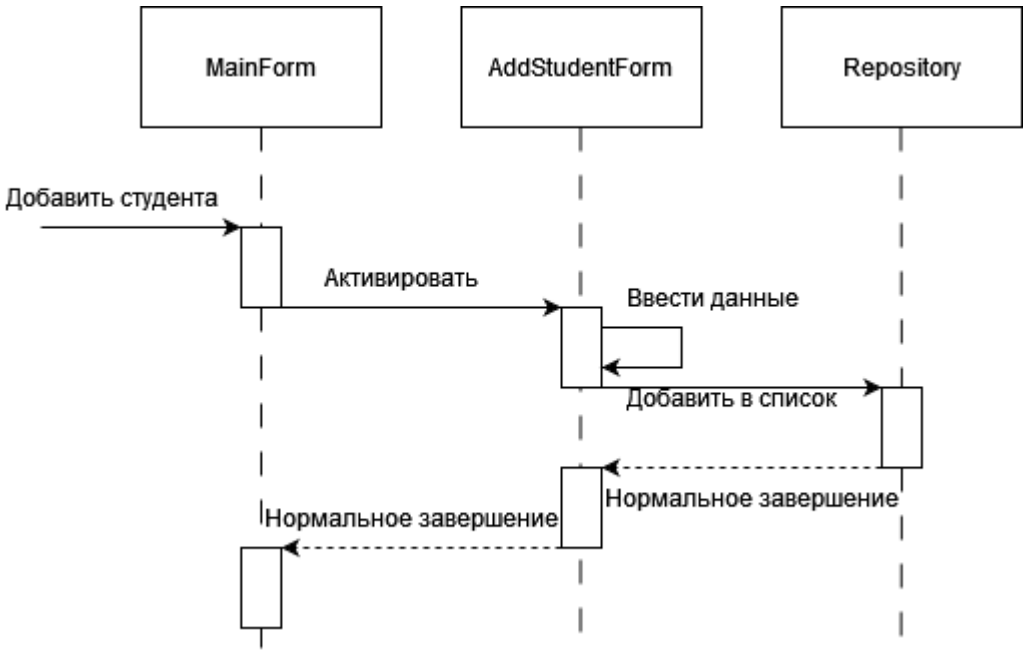

Рисунок 11 - диаграмма последовательности действий при добавлении нового студента

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

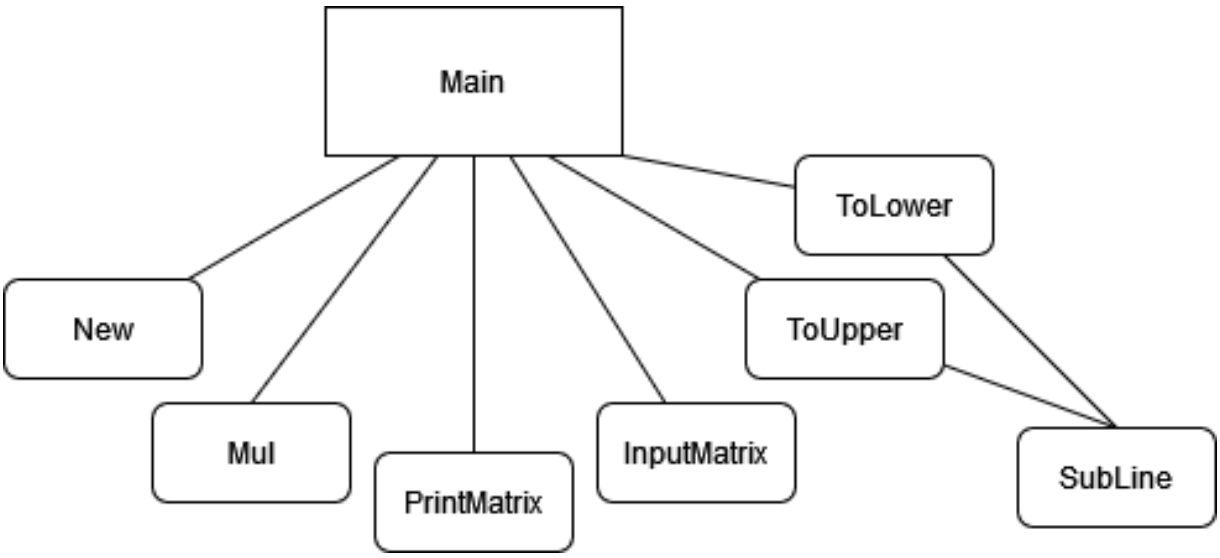

Рисунок 23 - структурная схема

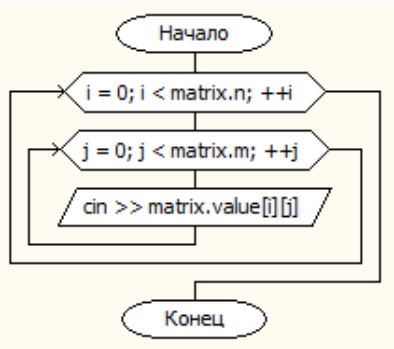

Рисунок 24 - схема алгоритма InputMatrix

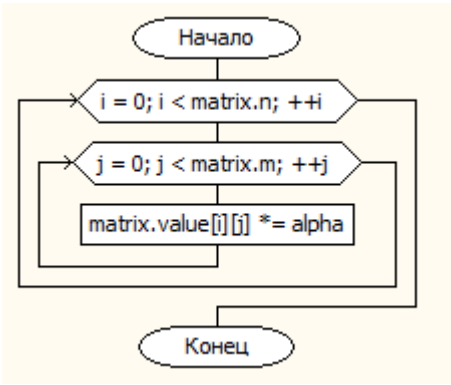

Рисунок 25 - схема алгоритма Mul

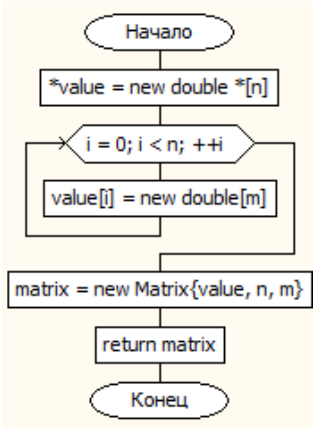

Рисунок 26 - схема алгоритма New

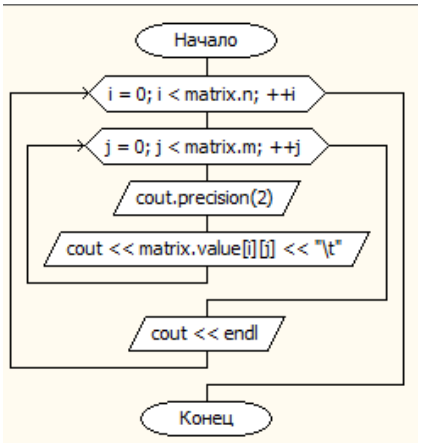

Рисунок 27 - схема алгоритма PrintMatrix

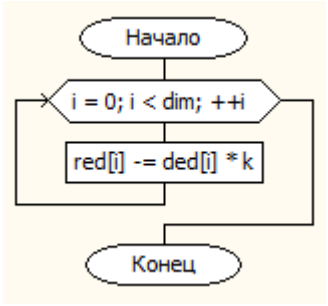

Рисунок 28 - схема алгоритма SubLine

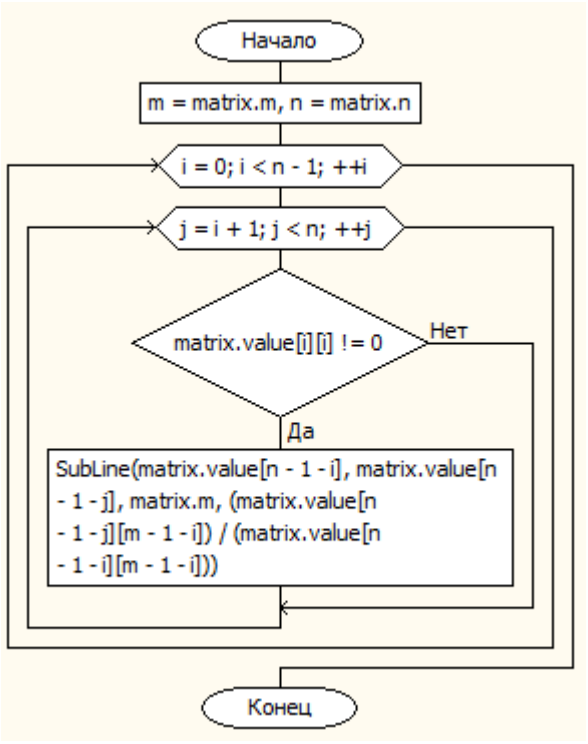

Рисунок 29 - схема алгоритма ToLower

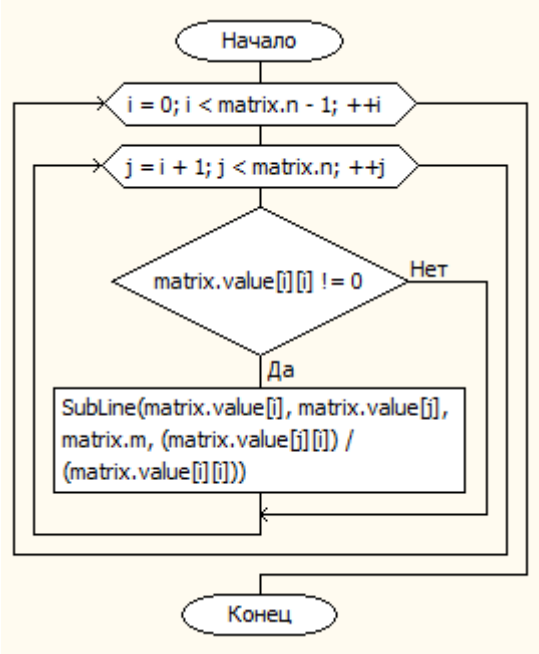

Рисунок 30 - схема алгоритма ToUpper

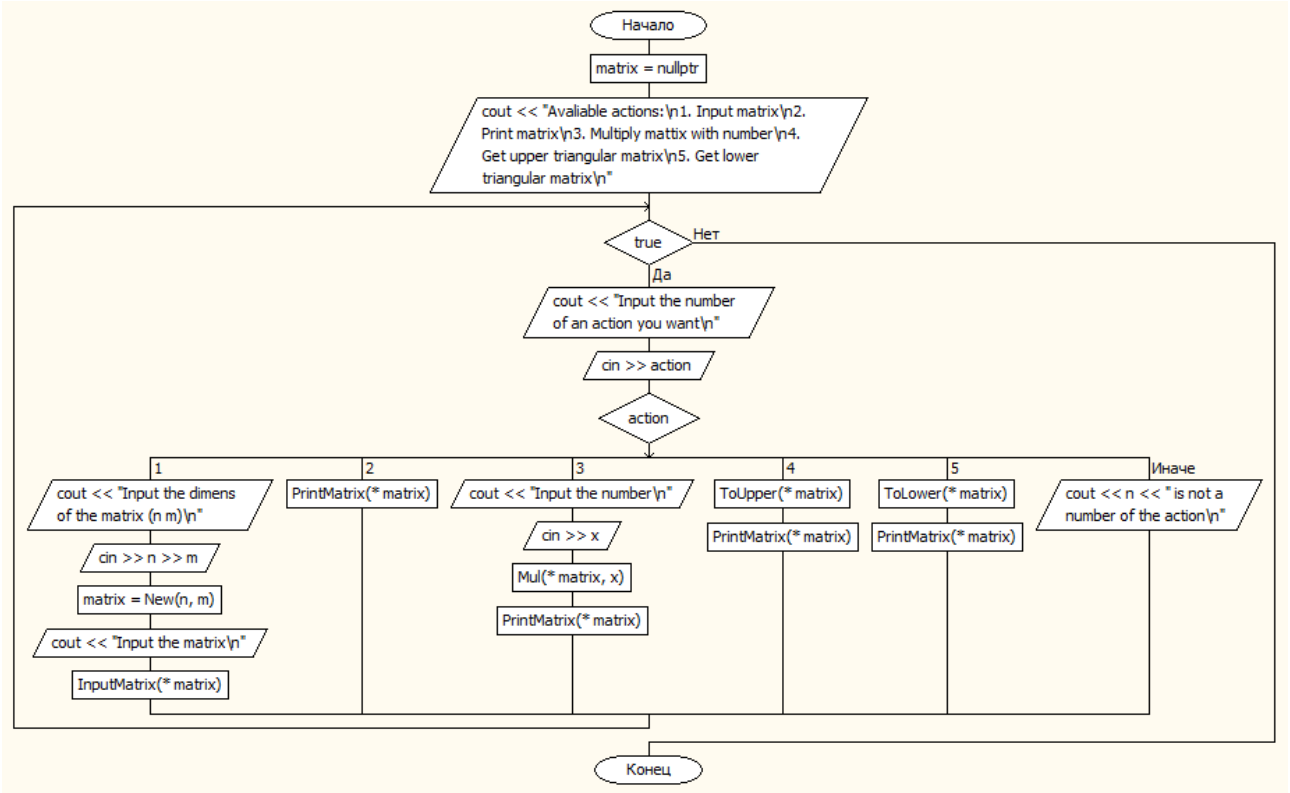

Рисунок 31 - схема алгоритма Main

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++.