

Оглавление

Введение	5
Задание 1	6
Техническое задание.....	6
Код программы.....	6
Примеры работающей версии программы	18
Объектная декомпозиция	21
Заключение	23
Задание 2	24
Техническое задание.....	24
Код программы.....	24
Пример работающей версии программы.....	29
Диаграмма состояний интерфейса	33
Заключение	33
Задание 3	34
Техническое задание.....	34
Код программы.....	34
Примеры работающей версии программы	66
Объектная декомпозиция	68
Заключение	72
Список использованных источников	73

Введение

Как бакалавры, обучающиеся по направлению «Прикладная информатика» мы обязаны знать не менее 2 языков программирования. При этом выполнение лабораторных работ, домашних задание и контрольных мероприятий по дисциплинам «Основы программирования» и «Объектно-ориентированное программирование» недостаточно для получения разработки программ различного назначения. Поэтому мы обязаны пройти специальную практику, целью которой является получения навыков создания небольших программных систем с оконными и консольными интерфейсами на языках Pascal и C++.

Практикум включает три задания, одно – на структурную декомпозицию и два – на объектную декомпозицию.

Задание 1. Создание программной системы на Object Pascal

Условие: 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.

О каждом товаре на овощной базе известны следующие сведения: наименование товара, поставщик, количество в наличии (кг), цена (за кг). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Определить, какого товара на базе больше всего.
2. Определить, сколько килограмм товара каждого наименования и поставщика можно купить на заданную сумму (показать все комбинации «товар, поставщик – масса»).
3. Определить, у какого поставщика выгоднее всего брать заданный товар.
4. Построить гистограмму, демонстрирующую среднюю цену за килограмм каждого товара.

Цель: получение навыков создания программной системы на языке Pascal с помощью среды Lazarus

Код основной программы:

```
program pr1;  
{ $mode objfpc } { $H+ }  
uses  
  { $IFDEF UNIX }  
  cthreads,  
  { $ENDIF }  
  { $IFDEF HASAMIGA }  
  athreads,  
  { $ENDIF }  
  Interfaces, Forms,
```

```

    tachart LazarusPkg, main, maxproduct, howmuchbuy, ben, table, base;
{$R *.res}
begin
    RequireDerivedFormResource:=True;
    Application.Scaled:=True;
    Application.Initialize;
    Application.CreateForm(TBased, Based);
    Application.CreateForm(TForm1, Form1);
    Application.CreateForm(TForm2, Form2);
    Application.CreateForm(TForm3, Form3);
    Application.CreateForm(TForm4, Form4);
    Application.CreateForm(TForm5, Form5);
    Application.Run;
end.

```

Код модуля main:

```

unit main;
{$mode objfpc} {$H+}
interface
uses
    Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;
type
    { TBased }
    TBased = class(TForm)
        Button1: TButton;
        maxbutton: TButton;
        muchbutton: TButton;
        benefitbutton: TButton;
        tablebutton: TButton;
        exitbutton: TButton;
        addbutton: TButton;
        deledit: TButton;
        nedit: TEdit;
        proedit: TEdit;
        amedit: TEdit;
        predit: TEdit;
        nlabel: TLabel;
        prolable: TLabel;
        amlabel: TLabel;
    end;

```

```

prlabel: TLabel;
procedure addbuttonClick(Sender: TObject);
procedure benefitbuttonClick(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure deleditClick(Sender: TObject);
procedure exitbuttonClick(Sender: TObject);
procedure maxbuttonClick(Sender: TObject);
procedure muchbuttonClick(Sender: TObject);
procedure tablebuttonClick(Sender: TObject);
end;
type zap = record
    name: string[22];
    provider: string[22];
    amount: string[22];
    price: string[22];
end;
var
    Based: TBased;
    f:file of zap;
    b:zap;
implementation
uses maxproduct, howmuchbuy, ben, table, base;
{$R *.lfm}

{ TBased }
procedure TBased.addbuttonClick(Sender: TObject);
begin
    AssignFile(f,'vegbase.dat');
    {$I-} Reset(F); {$I+}
    if ioresult=0 then seek(f,FileSize(f))
    else rewrite(f);
    b.name:=nedit.text;
    b.provider:=proedit.text;
    b.amount:=amedit.text;
    b.price:=predit.text;
    nedit.clear;
    proedit.clear;
    amedit.clear;
    predict.clear;

```

```

write(f, b);
nedit.setfocus;
closefile(f);
end;
procedure TBased.deleditClick(Sender: TObject);
var c:zap; fl:file of zap;
begin
  AssignFile(f,'vegbase.dat');
  assignfile(fl, 'trash.dat');
  Reset(F);
  rewrite(fl);
  b.name:=nedit.text;
  b.provider:=proedit.text;
  b.amount:=amedit.text;
  b.price:=predit.text;
  while not(EOF(f)) do begin
    read(f, c);
    if (b.name<>c.name) or (b.provider<>c.provider) then write(fl, c);
  end;
  reset(fl);
  rewrite(f);
  while not(EOF(fl)) do begin read(fl, c); write(f, c); end;
  nedit.clear;
  proedit.clear;
  amedit.clear;
  preedit.clear;
  nedit.setfocus;
  closefile(f);
  closefile(fl);
end;
procedure TBased.exitbuttonClick(Sender: TObject);
begin
  close;
end;
procedure TBased.maxbuttonClick(Sender: TObject);
begin
  assignfile(f, 'vegbase.dat');
  reset(f);
  form1.show;

```

```

end;
procedure TBased.muchbuttonClick(Sender: TObject);
begin
    assignfile(f, 'vegbase.dat');
    Reset(F);
    form2.show;
end;
procedure TBased.benefitbuttonClick(Sender: TObject);
begin
    assignfile(f, 'vegbase.dat');
    Reset(F);
    form3.show;
end;
procedure TBased.tablebuttonClick(Sender: TObject);
begin
    assignfile(f, 'vegbase.dat');
    Reset(F);
    form4.show;
end;
procedure TBased.Button1Click(Sender: TObject);
begin
    form5.show;
end;
end.

```

Код модуля maxproduct:

```

unit maxproduct;
{$mode ObjFPC} {$H+}
interface
uses
    Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;
type
    { TForm1 }

    TForm1 = class(TForm)
        backb: TButton;
        muchb: TButton;
        muche: TEdit;
        procedure backbClick(Sender: TObject);

```

```

    procedure muchbClick(Sender: TObject);
end;

var
    Form1: TForm1;
implementation
uses main;
{$R *.lfm}
{ TForm1 }
procedure TForm1.muchbClick(Sender: TObject);
var max:integer=0; fl:boolean; i, j:byte;
    aa: array [1..2, 1..20] of string[22]; ii, n:string[22];
begin
    i:=0;
    seek(f, 0);
    while not(eof(f)) do begin
        fl:=false;
        read(f, b);
        n:=b.name;
        for j:=1 to 20 do
            if aa[1, j]=n then begin
                str(strtoint(aa[2, j])+strtoint(b.amount), aa[2, j]);
                fl:=true;
                break;
            end;
        if fl=false then begin i:=i+1; aa[1, i]:=n; aa[2, i]:=b.amount; end;
    end;
    for j:=1 to i do
        if strtoint(aa[2, j])>max then begin
            max:= strtoint(aa[2, j]);
            ii:=aa[1, j];
        end;
    muche.Enabled:=true;
    muche.text:=ii;
end;
procedure TForm1.backbClick(Sender: TObject);
begin
    closefile(f);
    self.hide;
end;

```



```
end;  
end.
```

Код модуля howmuchbuy:

```
unit howmuchbuy;  
{ $mode ObjFPC } { $H+ }  
interface  
uses  
    Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;  
type  
  
    { TForm2 }  
  
    TForm2 = class(TForm)  
        prb: TButton;  
        Edit1: TEdit;  
        Label1: TLabel;  
        putb: TButton;  
        nextb: TButton;  
        exitb: TButton;  
        sume: TEdit;  
        namee: TEdit;  
        prove: TEdit;  
        mase: TEdit;  
        suml: TLabel;  
        namel: TLabel;  
        provl: TLabel;  
        masl: TLabel;  
        procedure exitbClick(Sender: TObject);  
        procedure nextbClick(Sender: TObject);  
        procedure prbClick(Sender: TObject);  
        procedure putbClick(Sender: TObject);  
    private  
    public  
    end;  
var  
    Form2: TForm2; j: integer = 0;  
implementation  
uses main;
```

```

{$R *.lfm}
{ TForm2 }
procedure TForm2.exitbClick(Sender: TObject);
begin
    closefile(f);
    self.hide;
end;

procedure TForm2.nextbClick(Sender: TObject);
begin
    j:=j+1;
    prb.enabled:=true;
    Putbclick(sender);
end;

procedure TForm2.prbClick(Sender: TObject);
begin
    j:=j-1;
    Putbclick(sender);
end;

procedure TForm2.putbClick(Sender: TObject);
var summa, h, k:integer; s:string[10];
begin
    seek(f, j);
    nextb.enabled:=true;
    namee.enabled:=true;
    prove.enabled:=true;
    mase.enabled:=true;
    edit1.enabled:=true;
    summa:=strtoint(sume.Text);
    read(f, b);
    namee.text:=b.name;
    prove.text:=b.provider;
    edit1.text:=b.price;
    h:=summa div strtoint(b.price);
    if h<=strtoint(b.amount) then begin str(h, s); mase.text:=s; end
    else mase.text:=b.amount;
    k:=j+1;

```

```

seek(f, k);
if eof(f) then nextb.enabled:=false;
k:=j-1;
if k<0 then prb.enabled:=false;
end;
end.

```

Код модуля ben:

```

unit ben;
{$mode ObjFPC} {$H+}
interface
uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;
type
  { TForm3 }
  TForm3 = class(TForm)
    exitb: TButton;
    putb: TButton;
    namee: TEdit;
    prove: TEdit;
    namel: TLabel;
    provl: TLabel;
    procedure exitbClick(Sender: TObject);
    procedure putbClick(Sender: TObject);
  private
  public
  end;
var
  Form3: TForm3;
implementation
uses main;
{$R *.lfm}

{ TForm3 }
procedure TForm3.exitbClick(Sender: TObject);
begin
  closefile(f);
  self.hide;
end;

```

```

procedure TForm3.putbClick(Sender: TObject);
var n, p: string[22]; j, be: word;
begin
  n:=namee.text;
  j:=0;
  seek(f, j);
  be:=10000;
  while not(eof(f)) do begin
    read(f, b);
    if b.name<>n then begin j:=j+1; seek(f, j); end
    else
      if strtoint(b.price)<be then begin
        be:=strtoint(b.price);
        p:=b.provider;
      end;
    end;
    if be=10000 then prove.text:='Ошибка'
    else prove.text:=p;
  end;
end.

```

Код модуля table:

```

unit table;
{$mode ObjFPC} {$H+}
interface
uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  TAGraph, TASeries, TASources, TACHartUtils;
type
  { TForm4 }

  TForm4 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    BarSeries1: TBarSeries;
    Chart1: TChart;
    Chart1BarSeries1: TBarSeries;
    Label1: TLabel;
    Label2: TLabel;

```

```

    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
end;
var
    Form4: TForm4;
implementation
uses main;
{$R *.lfm}
{ TForm4 }
procedure TForm4.Button1Click(Sender: TObject);
var i, j:byte; n:string[22]; fl:boolean;
    aa:array [1..4,1..20] of string[22]; ls : TListChartSource;
begin
    i:=0;
    seek(f, 0);
    while not(eof(f)) do begin
        read(f, b);
        n:=b.name;
        fl:=false;
        for j:=1 to 20 do
            if aa[1, j]=n then begin
                str(strtoint(aa[3, j])+strtoint(b.amount), aa[3, j]);
                str(strtoint(aa[2, j])+1, aa[2, j]);
                str(strtoint(aa[3, j]) div strtoint(aa[2, j]), aa[4, j]);
                fl:=true;
                break;
            end;
        if fl=false then begin
            i:=i+1;
            aa[1, i]:=n;
            aa[2, i]:='1';
            aa[3, i]:=b.amount;
            aa[4, i]:=b.amount; end;
    end;
    Chart1BarSeries1.clear;
    ls := TListChartSource.Create(Chart1);
    Chart1.BottomAxis.Marks.Source := ls;
    for j:=1 to i do begin
        ls.Add(j, strtoint(aa[4, j]), aa[1, j], clred);
    end;
end;

```

```

        Chart1BarSeries1.AddXY(j, strtoint(aa[4, j]), aa[4, j], clred);
    end;
end;

```

```

procedure TForm4.Button2Click(Sender: TObject);
begin
    closefile(f);
    self.hide;
end;
end.

```

Код модуля base:

```

unit base;
{$mode ObjFPC} {$H+}
interface
uses
    Classes, SysUtils, Forms, Controls, Graphics, Dialogs, Grids, StdCtrls;
type
    { TForm5 }

    TForm5 = class(TForm)
        Button1: TButton;
        StringGrid1: TStringGrid;
        procedure Button1Click(Sender: TObject);
    private
    public
    end;
var
    Form5: TForm5;
implementation
uses main;
{$R *.lfm}
{ TForm5 }
procedure TForm5.Button1Click(Sender: TObject);
var k:byte=0; j:byte; s:string[22];
begin
    assignfile(f, 'vegbase.dat');
    Reset(F);
    stringgrid1.rowcount:=filesize(f)+1;

```

```

stringgrid1.cells[1, 0]:='имя';
stringgrid1.cells[2, 0]:='производитель';
stringgrid1.cells[3, 0]:='наличие';
stringgrid1.cells[4, 0]:='цена за 1 кг';
for j:=1 to filesize(f) do begin
    seek(f, k);
    read(f, b);
    str(j, s);
    stringgrid1.cells[0, j]:=s;
    stringgrid1.cells[1, j]:=b.name;
    stringgrid1.cells[2, j]:=b.provider;
    stringgrid1.cells[3, j]:=b.amount;
    stringgrid1.cells[4, j]:=b.price;
    inc(k);
end;
closefile(f);
end;
end.

```

Рисунок 1-6 – работающая версия программы:

Рис. 1

Какого товара больше всего?

Посчитать

potato

Меню

Рис.2

Сколько кг можно купить на сумму

Введите сумму 200

Товар tomato

Поставщик jane

цена за 1 кг 60

Масса 3

Предыдущий Вывести Следующий

Меню

Рис.3

Поиск самого выгодного поставщика

Наименование товара tomato

Самый выгодный поставщик karen

Меню Вывести

Рис.4

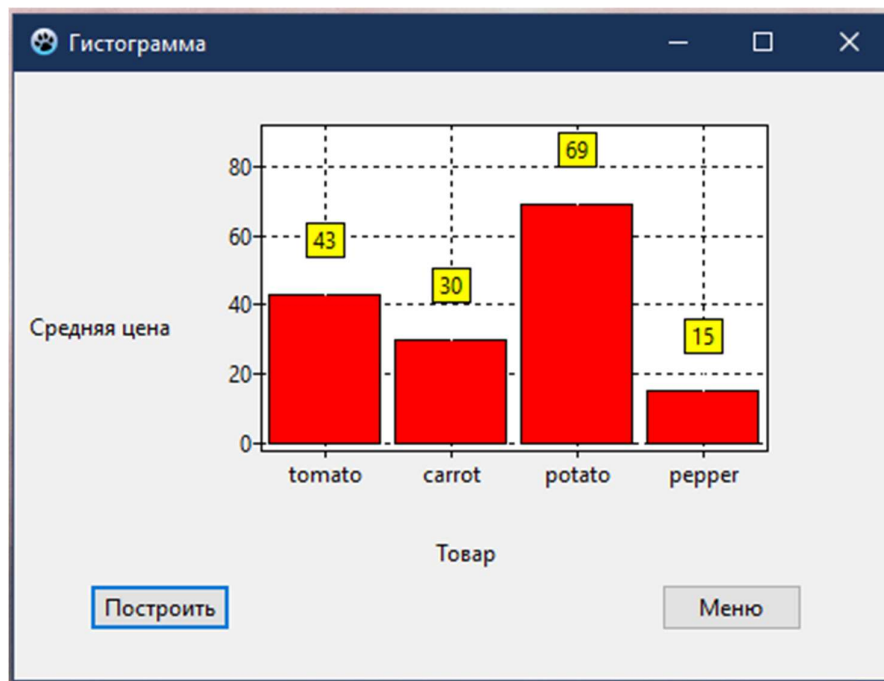


Рис.5

Таблица

	имя	производитель	наличие	цена за 1 кг
1	tomato	jane	35	60
2	tomato	karen	80	20
3	tomato	ivan	15	78
4	carrot	arsen	10	56
5	carrot	rane	45	83
6	carrot	anton	37	54
7	potato	dima	65	34
8	potato	sergey	100	40
9	potato	kate	43	68
10	pepper	olga	15	32

Показать

Рис.6

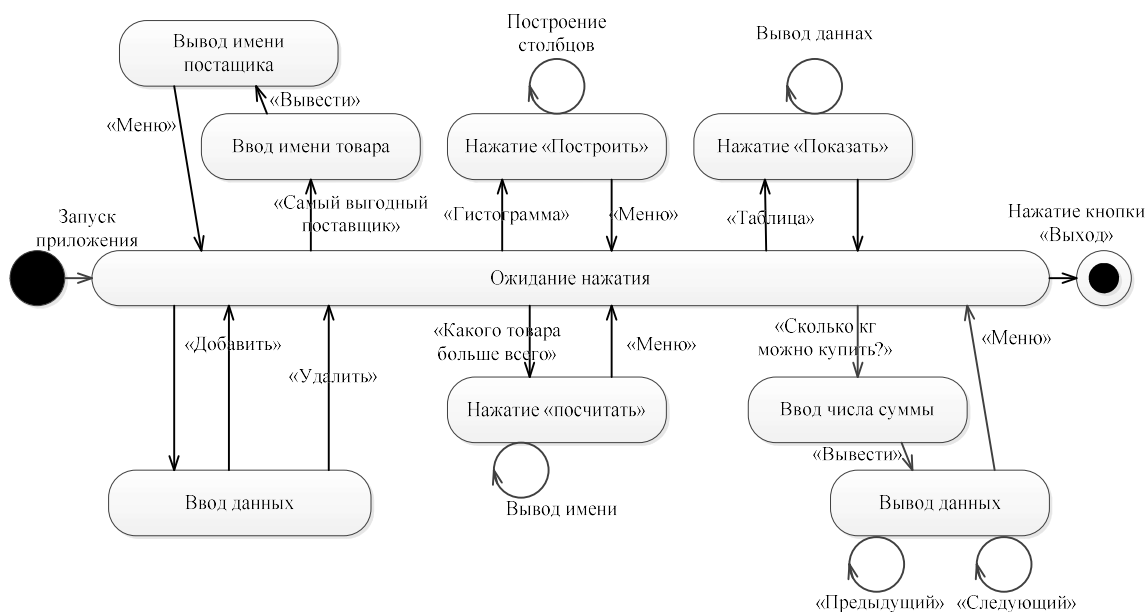


Рисунок 7 – диаграмма состояний интерфейса

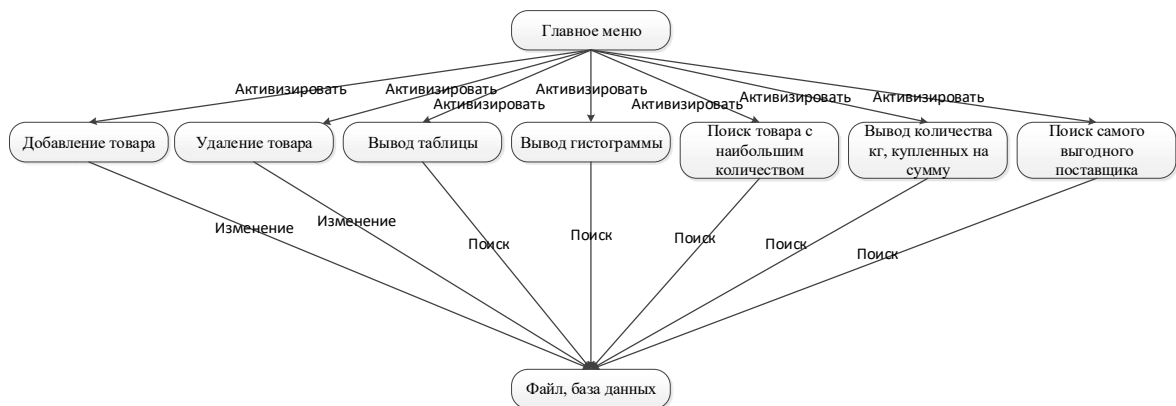


Рисунок 8 – объектная декомпозиция приложения

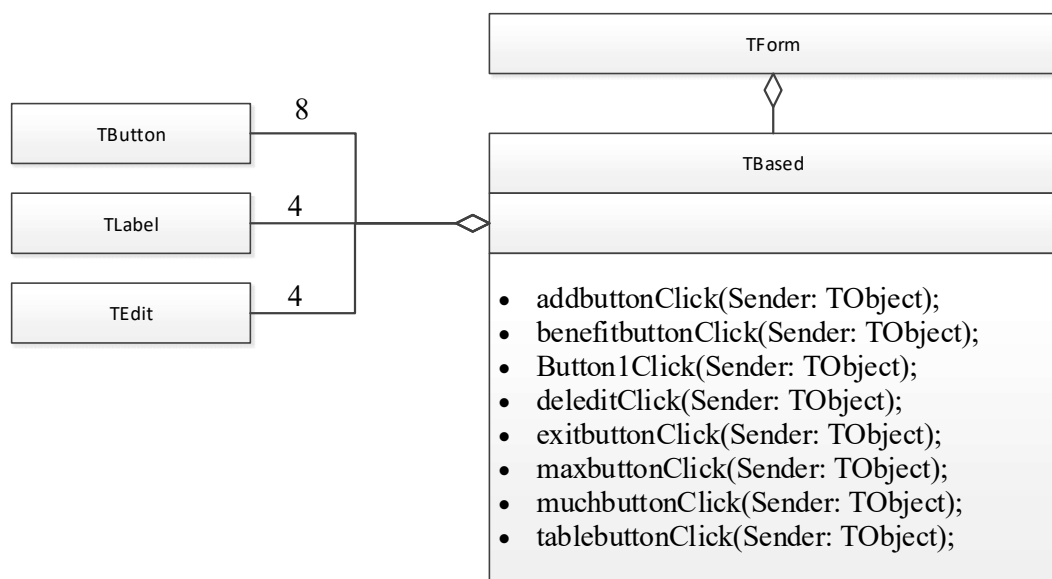


Рисунок 9 – диаграмма класса TBased

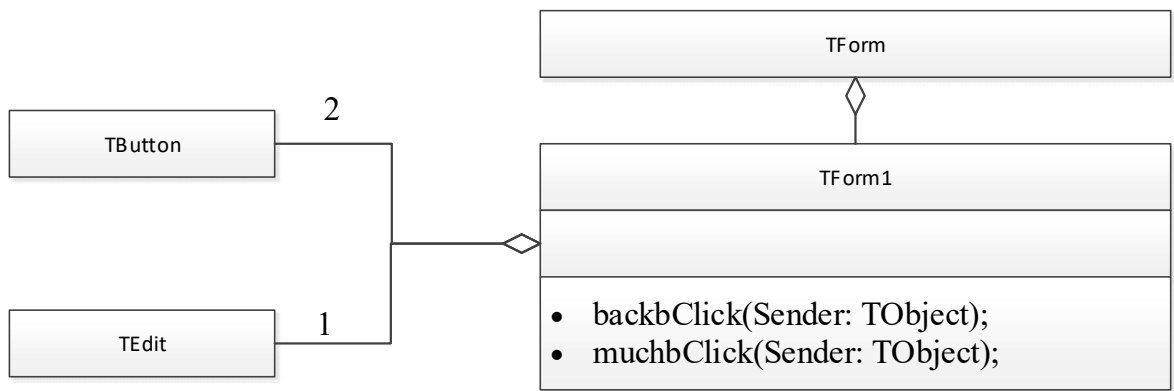


Рисунок 10 – диаграмма класса TForm1

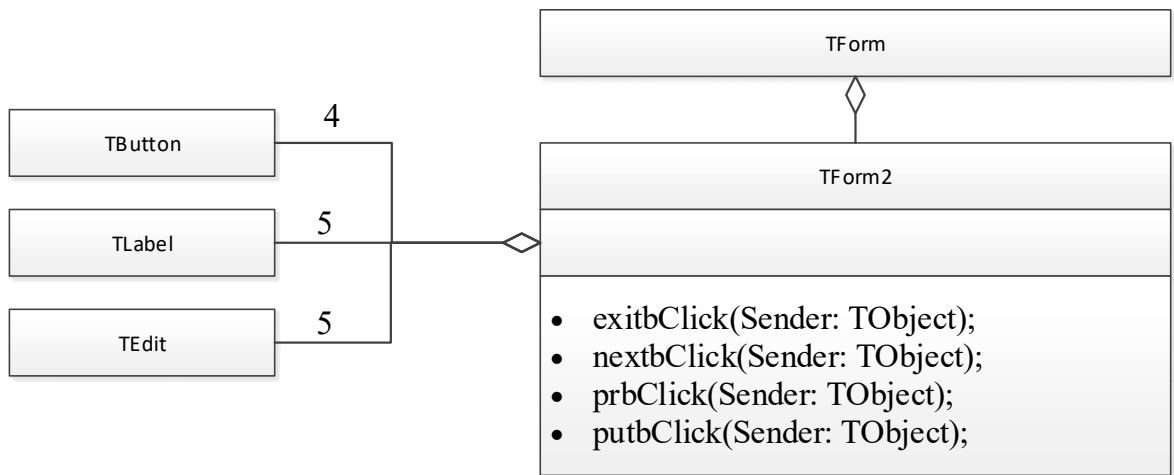


Рисунок 11 – диаграмма класса TForm2

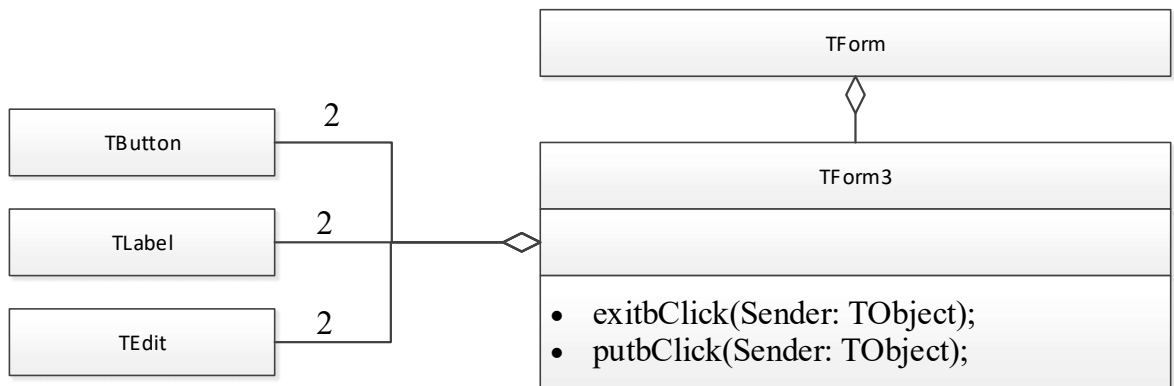


Рисунок 12 – диаграмма класса TForm3

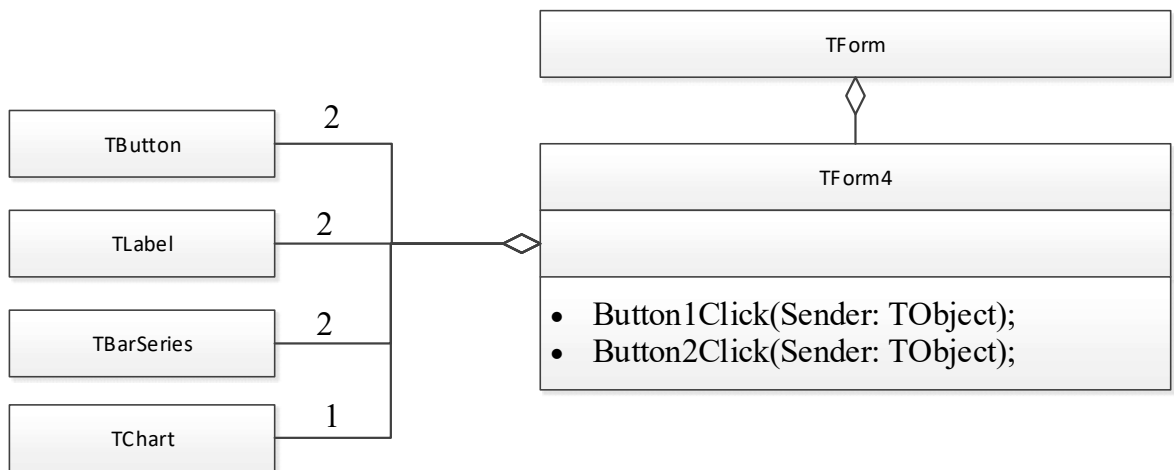


Рисунок 13 – диаграмма класса TForm4

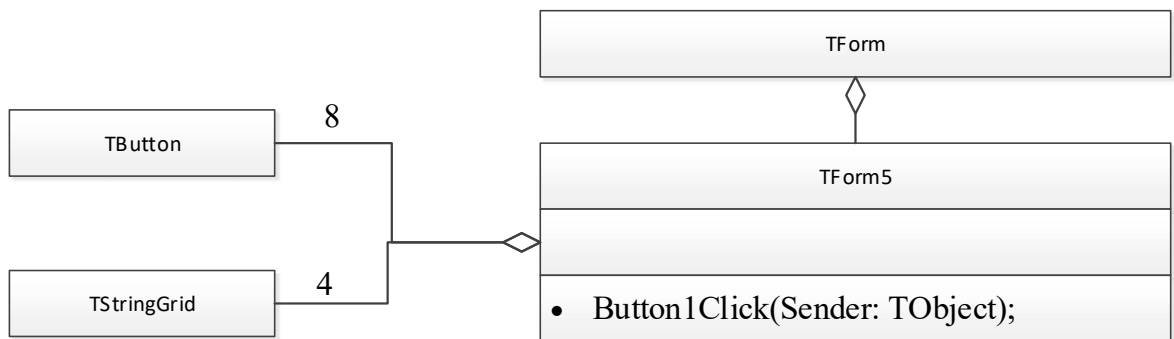


Рисунок 14 – диаграмма класса TForm5

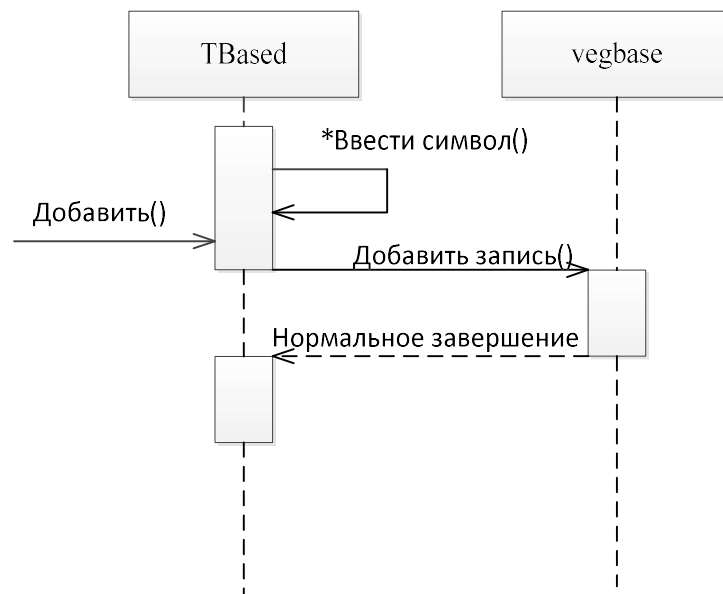


Рисунок 15 – диаграмма последовательностей действий при выполнении функции «Добавление записей»

Вывод: в соответствии с условием задачи было создано многооконное приложение, работающее с базой данных.

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

Условие: 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 выполняет простейшее исследование функций одной переменной. Реализовать следующие операции: ввод функции, преобразование ее во внутреннее представление (дерево), вычисление значения функции от заданного аргумента, а также вывод результатов на экран.

Цель: получение навыков создания программной системы на языке Pascal с помощью среды Lazarus

Код основной программы:

```
#include <iostream>
#include <math.h>
#include <cmath>
using namespace std;
const int N = 256;
struct core {
    char oper[5]; //знак
    int value; //константа
    core* left;
    core* right;
};
//прототипы функций
void print();
void tree(core* r, char str[]);
int position(char st[], char ops[]);
```

```
float solve(core* r, bool&key);
```

```
char function[N];
```

```
core* root;
```

```
bool key = true;
```

```
int x;
```

```
int main ()
```

```
{
```

```
    int n; float y;
```

```
    cout << "Choose the function:\n";
```

```
    print();
```

```
    cin >> n;
```

```
    switch (n) {
```

```
        case 1: strcpy_s(function, "2*x"); break;
```

```
        case 2: strcpy_s(function, "4*sin(-x)+(x^2-3)"); break;
```

```
        case 3: strcpy_s(function, "cos(7*x+2)"); break;
```

```
        case 4: strcpy_s(function, "(4-x)*(1+x)"); break;
```

```
        case 5: strcpy_s(function, "1/x"); break;
```

```
    }
```

```
    root = new core;
```

```
    tree(root, function);
```

```
    cout << "\nInput the value of x:";
```

```
    cin >> x;
```

```
    y = solve(root, key);
```

```
    if (key != false)cout<<y;
```

```
    else cout << "Impossible";
```

```
    return 0;
```

```
}
```

```
void print() {
```

```

        cout << "1)y=2*x\n2)y=4*sin(-x)+(x^2-3)\n3)y=cos(7*x+2)\n4)y=(4-
x)*(1+x)\n5)y=1/x\n";
    }

```

//есть ли "главная" операция

```

int position(char st[], char ops[]) {

```

```

    int i = 0, j = 0, k = 0; int p;

```

```

    p = 0;

```

```

    while (i < strlen(st) && p == 0) {

```

```

        if (st[i] == '(') j++;

```

```

        else if (st[i] == ')') k++;

```

```

        else if (j == k && strchr(ops, st[i]) != nullptr) p = i;

```

```

        i++;

```

```

    }

```

```

    return p;

```

```

}

```

```

void tree(core* r, char str[]) {

```

```

    char op[6], o[2] = { ' ', '\0' }, o2[3]{ ' ', ' ', '\0' }, strr[N], strl[N], str1[N], *

```

```

    ptr1;

```

```

    int nmain, nnn;

```

```

    op[0] = '+'; op[1] = '-'; op[2] = '\0';

```

```

    nmain = position(str, op);

```

```

    op[0] = '*'; op[1] = '/'; op[2] = '\0';

```

```

    if (nmain == 0) nmain = position(str, op);

```

```

    op[0] = '^'; op[1] = '\0';

```

```

    if (nmain == 0) nmain = position(str, op);

```

```

    op[0] = '*'; op[1] = '/'; op[2] = '+';

```

```

    op[3] = '-'; op[4] = '^'; op[5] = '\0';

```

```

    if (nmain != 0) {

```

```

        o[0] = str[nmain];

```

```

strcpy_s(r->oper, o);
strncpy_s(strl, str, nmain);
strl[nmain] = '\0';
if (strl[0] == '(' && position(strl, op) == NULL) {
    strcpy_s(strl, &strl[0]+1);
    strl[strlen(strl)-1] = '\0';
}
ptr1 = &str[nmain] + 1;
strcpy_s(strr, ptr1);
if (strr[0] == '(' && position(strr, op) == NULL) {
    strcpy_s(strr, &strr[0] + 1);
    strr[strlen(strr)-1] = '\0';
}
r->left = new core;
tree(r->left, strl);
r->right = new core;
tree(r->right, strr);
}
else
    if (str[0] == 'x' || strcmp(str, "-x")==0 ){           //переменная
        if (str[0] == 'x') {
            o[0] = 'x';
            strcpy_s(r->oper, o);
            r->left = nullptr;;
            r->right = nullptr;
        }
        else {
            o[0] = '-';
            strcpy_s(r->oper, o);
            strcpy_s(strl, "0");

```



```

        strcpy_s(strr, "x");
        r->left = new core;
        tree(r->left, strl);
        r->right = new core;
        tree(r->right, str);
    }
}
else
    if (atoi(str)!=0 || strcmp(str, "0") == 0) {           //константа
        o[0] = 'n';
        strcpy_s(r->oper, o);
        r->left = nullptr;
        r->right = nullptr;
        r->value = atoi(str);
    }
    else {           //функция
        ptr1 = strchr(str, '(');
        nnn = strlen(str) - strlen(ptr1);
        strncpy_s(str1, str, nnn);
        str1[nnn] = '\0';
        strcpy_s(r->oper, str1);
        r->right = nullptr;
        strcpy_s(strl, ptr1 + 1);
        strl[strlen(strl) - 1] = '\0';
        r->left = new core;
        tree(r->left, strl);
    }
}

```

```

Choose the function:
1)y=2*x
2)y=4*sin(-x)+(x^2-3)
3)y=cos(7*x+2)
4)y=(4-x)*(1+x)
5)y=1/x
4

Input the value of x:13
-126

```

Рисунок 1 – работающая версия программы

Блок-схема алгоритма:

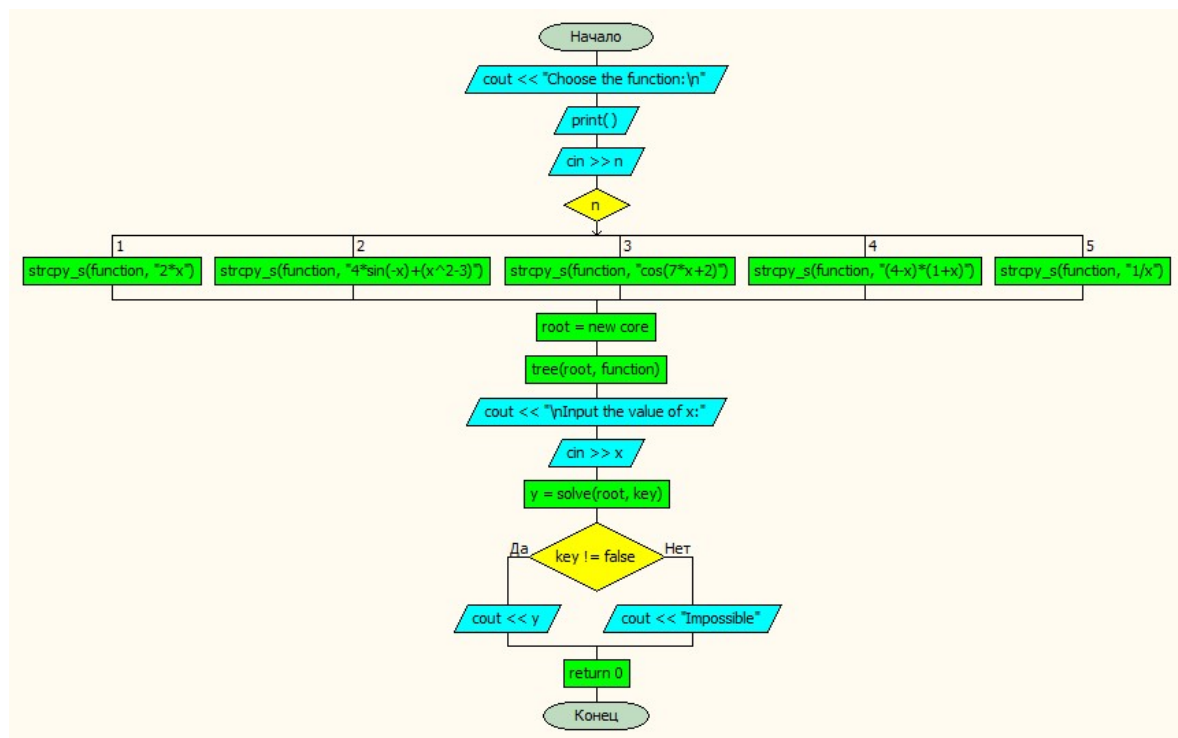


Рисунок 2 – основная программа

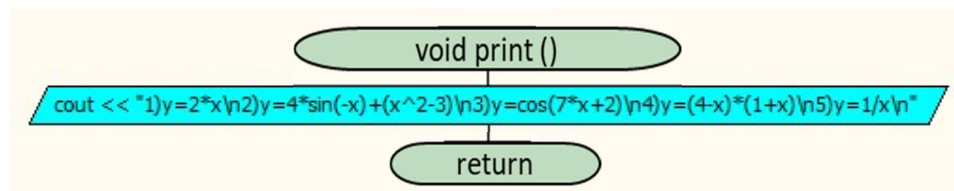


Рисунок 3 – печать меню функций

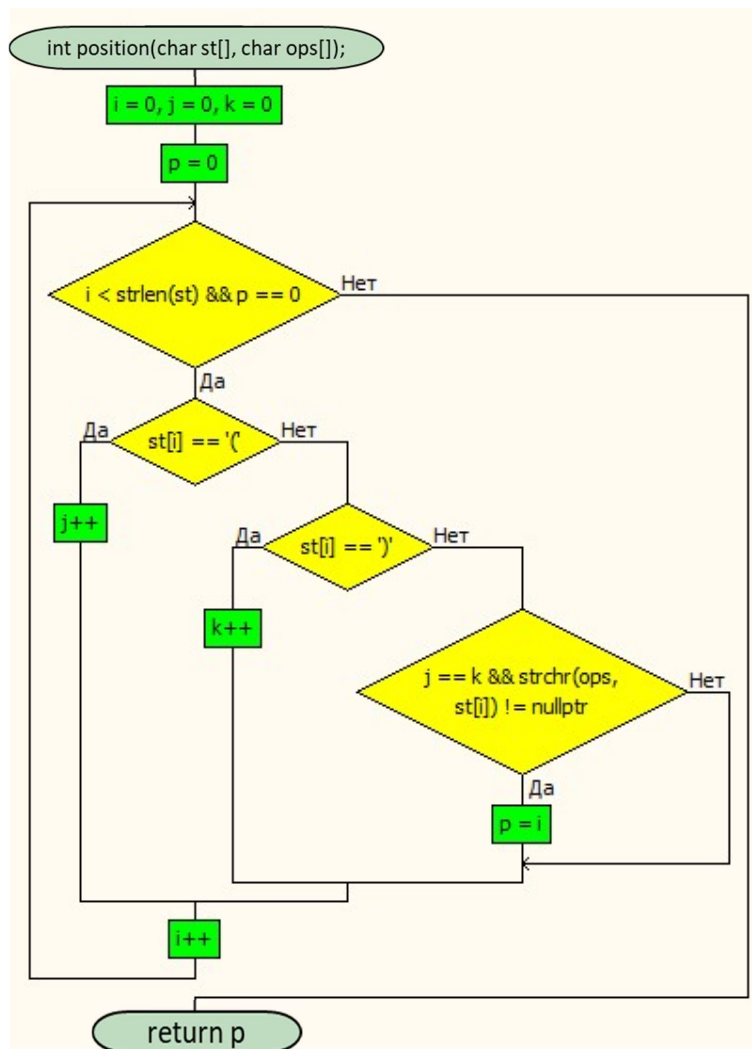
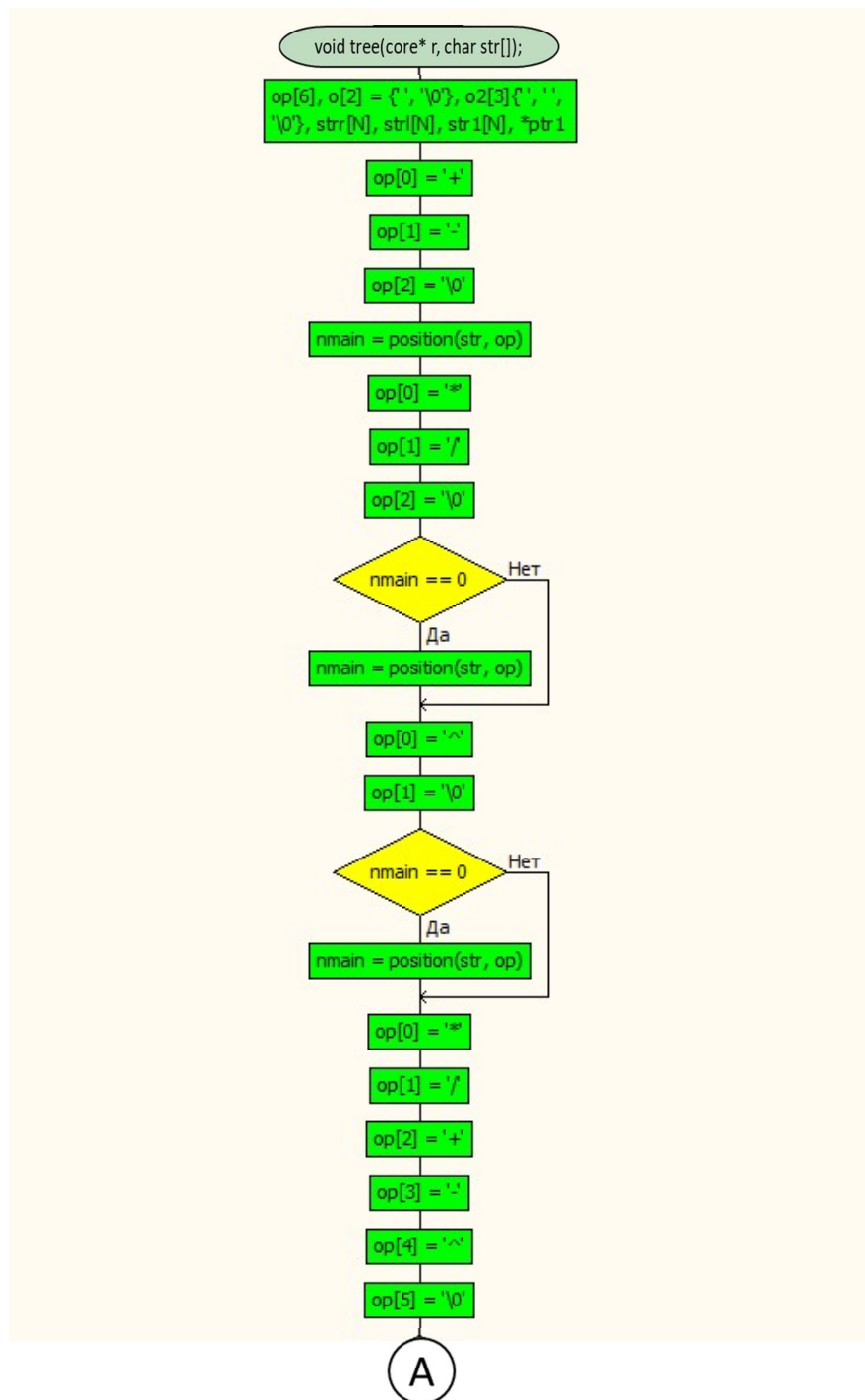


Рисунок 4 – определение последней операции



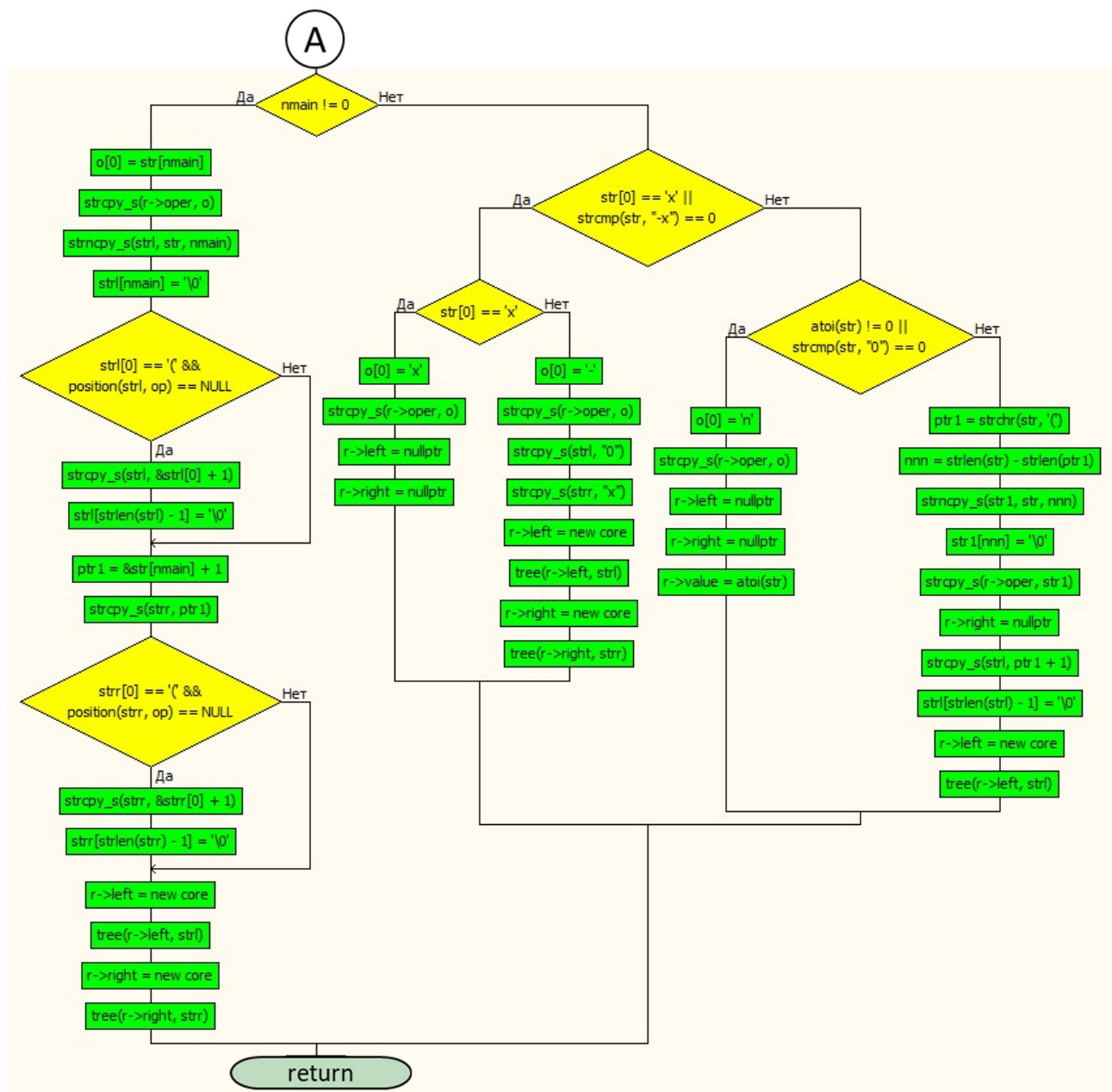


Рисунок 5 – составление дерева

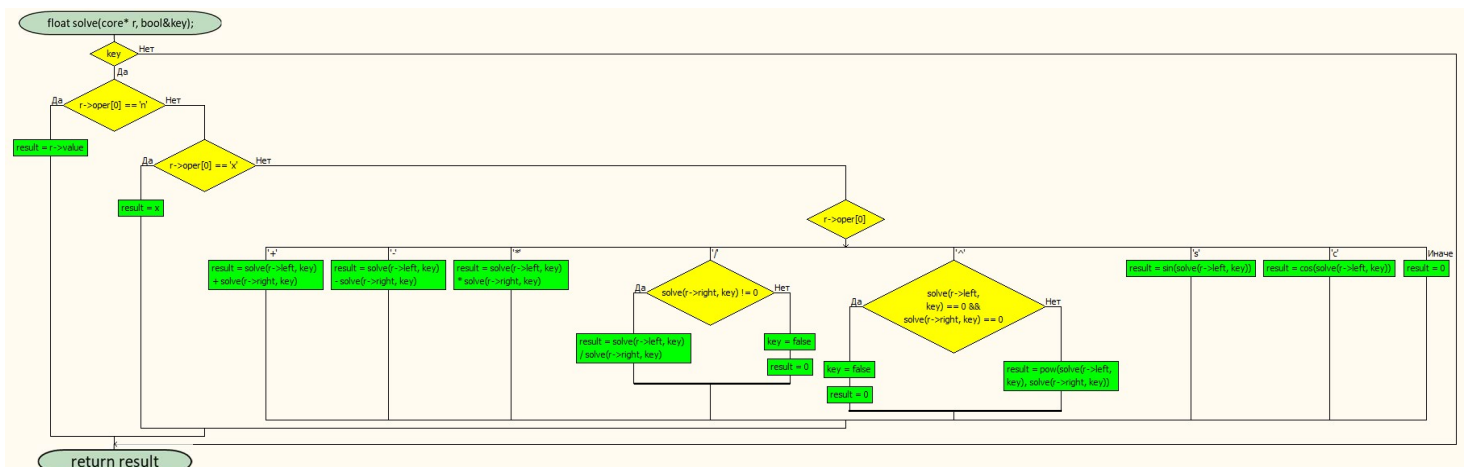


Рисунок 6 – вычисление значения функции

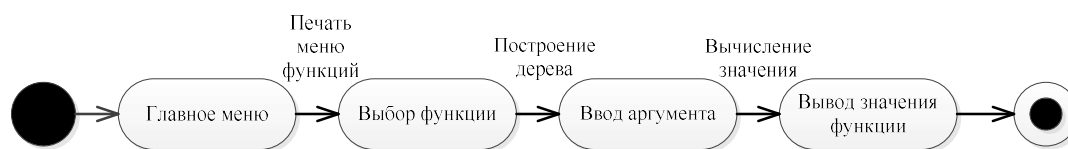


Рисунок 7 – диаграмма состояний интерфейса

Вывод: в соответствии с условием задачи была создана программа, которая предлагает пользователю на выбор функции и определяет её значение в зависимости от введённого аргумента.

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

Условие: 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.

О каждом товаре на овощной базе известны следующие сведения: наименование товара, поставщик, количество в наличии (кг), цена (за кг). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Определить, какого товара на базе больше всего.
2. Определить, сколько килограмм товара каждого наименования и поставщика можно купить на заданную сумму (показать все комбинации «товар, поставщик – масса»).
3. Определить, у какого поставщика выгоднее всего брать заданный товар.
4. Построить гистограмму, демонстрирующую среднюю цену за килограмм каждого товара.

Цель: получение навыков создания программной системы на языке C++ с помощью среды QtCreator

Код модуля basefile.h:

```
#ifndef BASEFILE_H
#define BASEFILE_H
#include <QFile>
struct Spis{
    QString nam; QString pro, am, pr;
    Spis* p, *s;
};
```

```

struct Spisok{
    QString name; int k, allpr, count;//количество, сумма всех цен,
количество производителей
    //(2 последних - ддля гистограммы и средней
цены
    Spisok *p;
};
struct recType{
    QString name, prov, amount, price;
};

class baseFile
{
    QFile *f;
    bool k1,//в запросе присутствует фамилия
    k2,//в запросе присутствует имя
    k3,//найдена фамилия
    k4,//найдено имя
    ff;//найдена запись
public:
    recType r;
    baseFile();
    ~baseFile();
    bool addRec(recType r);
    bool delRec(recType r);
    bool readRec();
    bool findFirst(const recType r1);
    bool findNext(const recType r1);
    bool FromFirst();
};

```



```
#endif // BASEFILE_H
```

Код модуля basefile.cpp:

```
#include "basefile.h"
```

```
#include <QMessageBox>
```

```
baseFile::baseFile()
```

```
{  
    f=new QFile("base.txt");  
    if(!f->exists()) // если файл не существует, то  
    {    // формируем сообщение  
        QMessageBox msg(QMessageBox::Critical, "Файл не найден",  
                        "Файл base.txt не создан",  
                        QMessageBox::Ok,0);  
        msg.exec();// выводим сообщение на экран  
    }  
    f->open(QFile::ReadWrite); // открываем файл для ввода-вывода  
}
```

```
baseFile::~baseFile() // деструктор
```

```
{  
    f->close();    // закрываем файл  
    delete f;      // освобождаем память под указатель  
}
```

```
bool baseFile::addRec(recType r)
```

```
{  
    f->seek(f->size()); // переходим на конец файла  
    QDataStream out(f); // связываем с файлом поток вывода  
    out<<r.name<<r.prov<<r.amount<<r.price;// выводим данные в файл  
    return true;  
}
```

```

bool baseFile::delRec(recType r1){
    f->seek(0);
    QFile* fl=new QFile("trash.txt");
    fl->open(QFile::ReadWrite);
    k1=(r1.name=="");          // устанавливаем два ключа поиска,
    присутствует имя
    k2=(r1.prov=="");//присутствует фамилия
    f->reset();
    fl->reset();
    bool fff = FromFirst();
    while(fff)
    {
        k3=(r1.name==r.name); //строим еще два ключа поиска
        k4=(r1.prov==r.prov);
        QDataStream out1(fl);
        if ((!k1 && !k2 && (!k3 || !k4))||
            (!k1 && k2 && !k3)||
            (k1 && !k2 && !k4)){
            out1<<r.name<<r.prov<<r.amount<<r.price;
            int k=sizeof(r);
            k=fl->size();
        }
        fff=readRec();
    }
    f->resize(0);
    fl->reset();
    fff=true;
    while (fff){
        QDataStream in1(fl); // связываем с файлом поток ввода
        if (in1.atEnd()||r.name=="")fff=false;
    }
}

```

```

        else
        {
            in1>>r.name>>r.prov>>r.amount>>r.price;
            addRec(r);
        }
    }
    fl->resize(0);
    fl->close();
    delete fl;
    return true;
}

bool baseFile::readRec()//для таблицы
{
    QDataStream in(f); // связываем с файлом поток ввода
    if (in.atEnd())return false;
    else
    {
        in>>r.name>>r.prov>>r.amount>>r.price;
        return true;
    }
}

bool baseFile::findFirst(const recType r1)//первое совпадение
{
    k1=(r1.name=="");          // устанавливаем два ключа поиска,
    присутствует имя
    k2=(r1.prov=="");//присутствует фамилия
    ff=false; // устанавливаем ключ поиска «запись не найдена»
    f->reset();
    bool fff = readRec();
    while(fff &&(!ff))

```

```

    {
        k3=(r1.name==r.name); //строим еще два ключа поиска
        k4=(r1.prov==r.prov);
        if ((!k1 && !k2 && k3 && k4)||
            (!k1 && k2 && k3)||//найдено имя
            (k1 && !k2 && k4)) //найдена фамилия
            ff=true; // ключ поиска «запись найдена»
        else fff=readRec();
    }
    return ff; // возвращаем ключ поиска
}

bool baseFile::findNext(const recType r1)
{
    ff=false; // ключ поиска «запись не найдена»
    bool fff = readRec();
    while((!ff) && fff)
    {
        k3=(r1.name==r.name);//строим еще два ключа поиска
        k4=(r1.prov==r.prov);
        if ((!k1 && !k2 && k3 && k4)|| //если имя и пров. не пустые и они
равны нужному
            (!k1 && k2 && k3)||
            (k1 && !k2 && k4))
            ff=true; // ключ поиска «запись найдена»
        else fff=readRec();
    }
    return ff; // возвращаем ключ поиска
}

bool baseFile::FromFirst(){
    bool ff;

```

```

        f->seek(0);
        ff=readRec();
        return ff;
    }

```

Код модуля tableform.h:

```

#ifndef TABLEFORM_H
#define TABLEFORM_H

```

```

#include <QObject>
#include <QWidget>
#include <basefile.h>
#include <QTableWidget>
#include <QPushButton>

```

```

class TableForm:public QWidget
{
    QTableWidget *table;
    QPushButton *Exitbtn;
public:
    TableForm();
    void showAll();//показать все записи
    void showRow(int i,recType r);
    void showResults(recType r1);
};

```

```

#endif // TABLEFORM_H

```

Код модуля tableform.cpp:

```

#include "mainform.h"
#include "tableform.h"

```

```

#include <QHBoxLayout>
#include <QVBoxLayout>
#include <QMessageBox>
TableForm::TableForm()
{
    this->setWindowTitle("Все записи");
    QStringList strlist;
    strlist<<("Имя
товара")<<"Производитель"<<"Количество"<<"Цена";
    table=new QTableWidgetItem(20, 4, this);
    table->setHorizontalHeaderLabels(strlist);
    QHBoxLayout*layout=new QHBoxLayout();
    Exitbtn=new QPushButton("Меню");
    layout->addWidget(Exitbtn);
    QVBoxLayout*layout1=new QVBoxLayout();
    layout1->addWidget(table);
    layout1->addLayout(layout);
    setLayout(layout1);
    connect(Exitbtn, SIGNAL(clicked(bool)), this, SLOT(close()));
}
void TableForm::showRow(int i,recType r)
{
    QTableWidgetItem *item; // элемент таблицы
    item = new QTableWidgetItem(); // создаем элемент
    item->setFlags(Qt::NoItemFlags);//запрещаем выделение
    item->setText(r.name); // записываем текст
    table->setItem(i,0,item);// привязываем элемент к таблице
    item = new QTableWidgetItem();// создаем элемент
    item->setFlags(Qt::NoItemFlags); //запрещаем выделение
    item->setText(r.prov);

```

```

table->setItem(i,1,item); // привязываем элемент
item = new QTableWidgetItem();// создаем элемент
item->setFlags(Qt::NoItemFlags); //запрещаем выделение
item->setText(r.amount); // записываем текст
table->setItem(i,2,item); // привязываем элемент
item = new QTableWidgetItem();// создаем элемент
item->setFlags(Qt::NoItemFlags); //запрещаем выделение
item->setText(r.price); // записываем текст
table->setItem(i,3,item);
}
void TableForm::showAll()
{
    baseFile base;
    if (!base.FromFirst())
    { // если файл пустой , то создаем сообщение
        QMessageBox msg(QMessageBox::Critical,
            "Нет данных",
            "База пуста",
            QMessageBox::Ok,0);
        msg.exec(); // выводим сообщение
    }
    else
    { // иначе - выводим таблицу по строкам
        showRow(0,base.r);
        int i=0;
        while (base.readRec())
            showRow(++i,base.r);
        table->setRowCount(i+1);
        resize(600,400);
        show();
    }
}

```

```

    }
    base.FromFirst();
}
void TableForm::showResults(recType r1)
{
    baseFile base;
    //base->reset();
    if (!base.findFirst(r1))
    { // если данные не найдены, то создаем сообщение
        QMessageBox msg(QMessageBox::Critical,
            "Нет данных",
            "Данные не найдены",
            QMessageBox::Ok,0);
        msg.exec();
    }
    else
    { // иначе - выводим результаты по строкам
        showRow(0,base.r);
        int i=0;
        while (base.findNext(r1))
            showRow(++i,base.r);
        table->setRowCount(i+1);
        resize(350,200);
        show();
    }
}

```

Код модуля themostform.h:

```

#ifndef THEMOSTFORM_H

```



```

#define THEMOSTFORM_H

#include <QWidget>
#include <QObject>
#include <basefile.h>

#include <QPushButton>
#include <QLineEdit>

class themostForm:public QWidget
{
    Q_OBJECT//!!!!работает с сигналами и слотами
    QPushButton* countbtn;
    QPushButton* exitbtn;
    QLineEdit* resulte;
public:
    themostForm();
public slots:
    void prCount();
};

```

```

#endif // THEMOSTFORM_H

```

Код модуля themostform.cpp:

```

#include "themostform.h"

```

```

#include "QVBoxLayout"

```

```

themostForm::themostForm()

```

```

{
    this->setWindowTitle("Какого товара больше всего?");
    countbtn=new QPushButton("Посчитать");
    exitbtn=new QPushButton("Меню");
    resulte=new QLineEdit;
    resulte->setReadOnly(true);
    QVBoxLayout* layout=new QVBoxLayout();
    layout->addWidget(countbtn);
    layout->addWidget(resulte);
    layout->addWidget(exitbtn);
    setLayout(layout);
    resize(300,150);
    resulte->clear();
    connect(countbtn, SIGNAL(clicked(bool)), this, SLOT(prCount()));
    connect(exitbtn, SIGNAL(clicked(bool)), this, SLOT(close()));
}

void themostForm::prCount() {
    int max = 0;
    QString str;
    baseFile base;
    bool fff = base.FromFirst();
    if (fff==false) resulte->setText("0");
    else {
        Spisok *first=new Spisok;
        first->name=base.r.name;
        str=base.r.amount;
        first->k=str.toInt();
        //first->allpr=base.r.price.toInt();
        ///first->count=1;
        first->p=nullptr;
    }
}

```

```

Spisok*r, *r1;
while (fff=base.readRec(), str=base.r.amount, fff){
    r=first;
    bool find=false;
    while (r!=nullptr&&find==false){
        if (base.r.name==r->name) {
            r->k+=str.toInt();
            //r->allpr+=base.r.price.toInt();
            //r->count+=1;
            find=true;}
        else{
            if(r->p!=nullptr)r=r->p;
            else{
                r1=new Spisok;
                r1->name=base.r.name;
                r1->k=str.toInt();
                //r->allpr=base.r.price.toInt();
                //r->count=1;
                r1->p=nullptr;
                r->p=r1;
                r=r1;
                find=true;}
            }
        }
    }
    r=first;r1=first;
    while (r!=nullptr){
        if (r->k>max){
            str=r->name;
            max=r->k;

```

```

    }
    r=r->p;
    delete r1;
    r1=r;
}
resulte->setText(str);
}
}

```

Код модуля muchform.h:

```

#ifndef MUCHFORM_H
#define MUCHFORM_H

#include <QObject>
#include <QWidget>
#include <QPushButton>
#include <QLineEdit>
#include <QLabel>

#include <basefile.h>

class muchForm:public QWidget
{
    Q_OBJECT
    int j=0;
    Spis* first;
    QPushButton* prevbtn;
    QPushButton* inputbtn;

```

```

        QPushButton* sucbtn;
        QPushButton* exitbtn;
        QLabel* inputl;
        QLineEdit* inpute;
        QLabel* namel;
        QLineEdit* namee;
        QLabel* provl;
        QLineEdit* prove;
        QLabel* pricel;
        QLineEdit* pricee;
        QLabel* massl;
        QLineEdit* masse;
        friend Spis* createsp();
public:
        muchForm();
public slots:
        void printf();
        void printsuc();
        void printprev();
        void closeform();
};
        Spis* createsp();
#endif // MUCHFORM_H

```

Код модуля muchform.cpp:

```

#include "muchform.h"
#include <QGridLayout>
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QMessageBox>

```

```

muchForm::muchForm()
{
    this->setWindowTitle("Сколько кг можно купить на сумму");
    prevbtn = new QPushButton("Предыдущий");
    inputbtn= new QPushButton("Вывести");
    sucbtn=new QPushButton("Следующий");
    exitbtn=new QPushButton("Меню");
    inputl=new QLabel("Введите сумму");
    inpute=new QLineEdit;
    namel=new QLabel("Товар");
    namee=new QLineEdit;
    namee->setReadOnly(true);
    provl=new QLabel("Поставщик");
    prove=new QLineEdit;
    prove->setReadOnly(true);
    pricel=new QLabel("Цена за 1 кг");
    pricee=new QLineEdit;
    pricee->setReadOnly(true);
    massl=new QLabel("Масса");
    masse=new QLineEdit;
    masse->setReadOnly(true);
    prevbtn->setEnabled(false);
    QGridLayout*layout=new QGridLayout();
    layout->addWidget(namel, 1, 1);
    layout->addWidget(namee, 1, 2);
    layout->addWidget(provl, 2, 1);
    layout->addWidget(prove, 2, 2);
    layout->addWidget(pricel, 3, 1);
    layout->addWidget(pricee, 3, 2);

```

```

layout->addWidget(massl, 4, 1);
layout->addWidget(masse, 4, 2);
layout->addWidget(prevbtn, 5, 1);
layout->addWidget(inputbtn, 5, 2);
layout->addWidget(sucbtn, 5, 3);
layout->addWidget(exitbtn, 6, 2);
QHBoxLayout*layouttop=new QHBoxLayout();
layouttop->addWidget(inputl);
layouttop->addWidget(inpute);
QVBoxLayout*all=new QVBoxLayout();
all->insertLayout(0, layouttop, 0);
all->insertLayout(1, layout, 0);
setLayout(all);

connect(exitbtn, SIGNAL(clicked(bool)), this, SLOT(closeform()));
connect(prevbtn, SIGNAL(clicked(bool)), this, SLOT(printprev()));
connect(sucbtn, SIGNAL(clicked(bool)), this, SLOT(printsuc()));
connect(inputbtn, SIGNAL(clicked(bool)), this, SLOT(printinf()));
}

void muchForm::closeform(){
    Spis*r=first;
    while (r=first, first!=nullptr){
        first=first->p;
        delete r;
    }
    close();
}

void muchForm::printsuc(){
    j+=1;
    prevbtn->setEnabled(true);
    printf();
}

```

```

    }
    void muchForm::printprev() {
        j-=1;
        sucbtn->setEnabled(true);
        printf();
    }
    void muchForm::printf() {
        int summa=input->text().toInt(), h;
        first=createsp(); Spis *r;
        r=first;
        for (int i=0; i<j; i++) r=r->s;
        namee->setText(r->nam);
        prove->setText(r->pro);
        pricee->setText(r->pr);
        h=summa/r->pr.toInt();
        if (h<=r->am.toInt()) {
            QString str=QString::number(h);
            masse->setText(str);
        }
        else masse->setText(r->am);
        if (r->s==nullptr) sucbtn->setEnabled(false);
        if (r->p==nullptr) prevbtn->setEnabled(false);
    }
    Spis* createsp() {
        baseFile base;
        bool fff = base.FromFirst();
        Spis* first, *r, *r1;
        if (fff==false){QMessageBox msg(QMessageBox::Critical, "Файл не
найден",
                                "Файл base.txt не создан",

```



```

        QMessageBox::Ok,0);
    msg.exec();
}
else {
    first=new Spis;
    first->nam=base.r.name;
    first->pro=base.r.prov;
    first->am=base.r.amount;
    first->pr=base.r.price;
    first->p=nullptr;
    first->s=nullptr;
    r=first;
    while (fff=base.readRec(), fff){
        r1=new Spis;
        r1->nam=base.r.name;
        r1->pro=base.r.prov;
        r1->am=base.r.amount;
        r1->pr=base.r.price;
        r1->p=r;
        r1->s=nullptr;
        r->s=r1;
        r=r1;
    }
}
r=first;
return first;
}

```

Код модуля benproform.h:

```

#ifndef BENPROFORM_H

```

```

#define BENPROFORM_H

#include <QObject>
#include <QWidget>
#include <QPushButton>
#include <QLineEdit>
#include <QLabel>

#include <basefile.h>
class BenProForm:public QWidget
{
    Q_OBJECT
    QPushButton* inputbtn;
    QPushButton* exitbtn;
    QLabel* inputNl;//имя товара
    QLineEdit* inputNe;
    QLabel* outputPl;//имя производителя
    QLineEdit* outputPe;
public:
    BenProForm();
public slots:
    void printBen();
};

#endif // BENPROFORM_H
Код модуля benproform.cpp:

#include "benproform.h"
//#include "muchform.h"

```

```

#include <QGridLayout>
#include <QVBoxLayout>
#include <QHBoxLayout>
BenProForm::BenProForm()
{
    this->setWindowTitle("Поиск самого выгодного поставщикф");
    inputbtn= new QPushButton("Вывести");
    exitbtn=new QPushButton("Меню");
    inputNl=new QLabel("Введите наименование товара");
    inputNe=new QLineEdit;
    outputPl=new QLabel("Самый выгодный поставщик");
    outputPe=new QLineEdit;
    inputNe->clear();
    outputPe->clear();
    inputNe->setFocus();
    outputPe->setReadOnly(true);
    QGridLayout*layout=new QGridLayout();
    layout->addWidget(inputNl, 1, 1);
    layout->addWidget(inputNe, 1, 2);
    layout->addWidget(outputPl, 2, 1);
    layout->addWidget(outputPe, 2, 2);
    layout->addWidget(inputbtn, 3, 1);
    layout->addWidget(exitbtn, 3, 2);
    setLayout(layout);
    connect(exitbtn, SIGNAL(clicked(bool)), this, SLOT(close()));
    connect(inputbtn, SIGNAL(clicked(bool)), this, SLOT(printBen()));
}
void BenProForm::printBen(){
    QString str=inputNe->text(), result;int min=1000;
    baseFile base;

```

```

bool fff=base.FromFirst();
while(fff){
    if (base.r.name==str&&base.r.price.toInt()<min){
        result=base.r.prov;
        min=base.r.price.toInt();}
    fff=base.readRec();
}
outputPe->setText(result);
}

```

Код модуля gystform.h:

```

#ifndef GYSTFORM_H
#define GYSTFORM_H

#include <QObject>
#include <QWidget>
#include <QPushButton>
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QGridLayout>

#include "basefile.h"

#include <QtCharts/QtCharts>
class gystForm:public QWidget
{
    Q_OBJECT
    bool twice=false;
    QPushButton* exitbtn;

```

```

QPushButton* printbtn;
QChart* Gystog;
QChartView* chartview;
QBarSeries *series;
QValueAxis *axisY;
QBarCategoryAxis *axisX;
QBarSet*srpr;
QStringList categories;//имена продуктов
public:
    gystForm();
public slots:
    void printGyst();
    //void closeform();
};

```

```

#endif // GYSTFORM_H

```

Код модуля gystform.cpp:

```

#include "gystform.h"
gystForm::gystForm()
{
    this->setWindowTitle("Гистограмма");
    exitbtn=new QPushButton("Меню");
    printbtn=new QPushButton("Вывести");
    Gystog=new QChart();
    chartview = new QChartView(Gystog);
    axisY= new QValueAxis();
    axisX= new QBarCategoryAxis();
    series= new QBarSeries();
    srpr=new QBarSet("Средняя цена");
}

```

```

QHBoxLayout* layout=new QHBoxLayout();
layout->addWidget(printbtn);
layout->addWidget(exitbtn);
QVBoxLayout* layout1=new QVBoxLayout();
layout1->addWidget(chartview);
layout1->addLayout(layout);
setLayout(layout1);
resize(600, 400);
connect(exitbtn, SIGNAL(clicked(bool)), this, SLOT(close()));
connect(printbtn, SIGNAL(clicked(bool)), this, SLOT(printGyst()));
}

void gystForm::printGyst() {
    baseFile base;float max=0;
    srpr->remove(0, categories.size());
    axisX->clear();
    categories.resize(0);
    bool fff = base.FromFirst();
    if (fff==false) {QMessageBox msg(QMessageBox::Critical, "Файл не
найден",
                                "Файл base.txt не создан",
                                QMessageBox::Ok,0);
        msg.exec();
    }
    else {
        Spisok *first=new Spisok;
        first->name=base.r.name;
        QString str=base.r.amount;
        //first->k=str.toInt();
        first->allpr=base.r.price.toInt();
        first->count=1;
    }
}

```

```

first->p=nullptr;
Spisok*r, *r1;
while (fff=base.readRec(), str=base.r.amount, fff){
r=first;
bool find=false;
while (r!=nullptr&&find==false){
    if (base.r.name==r->name) {
        //r->k+=str.toInt();
        r->allpr+=base.r.price.toInt();
        r->count+=1;
        find=true;}
    else{
        if(r->p!=nullptr)r=r->p;
        else{
            r1=new Spisok;
            r1->name=base.r.name;
            //r1->k=str.toInt();
            r1->allpr=base.r.price.toInt();
            r1->count=1;
            r1->p=nullptr;
            r->p=r1;
            r=r1;
            find=true;
        }
    }
}
}
r=first;r1=first;
while (r!=nullptr){
    srpr->append(r->allpr/r->count);

```

```

        categories<<r->name;
        if (r->allpr/r->count>max) max=r->allpr/r->count;
        r=r->p;
        delete r1;
        r1=r;
    }
}
series->append(srpr);
//-
Gystog->addSeries(series);
//про y
axisY->setRange(0,max);
Gystog->addAxis(axisY, Qt::AlignLeft);
series->attachAxis(axisY);
//про x
axisX->append(categories);
Gystog->addAxis(axisX, Qt::AlignBottom);
series->attachAxis(axisX);
//--
Gystog->legend()->setVisible(true);
Gystog->legend()->setAlignment(Qt::AlignBottom);
chartview->setRenderHint(QPainter::Antialiasing);
twice=true;
}

```

Код модуля mainform.h:

```

#ifndef MAINFORM_H
#define MAINFORM_H

#include "gystform.h"

```



```

#include <QWidget>
#include <QPushButton>
#include <QLabel>
#include <QLineEdit>

#include <TableForm.h>
#include <themostform.h>
#include <muchform.h>
#include <benproform.h>
#include <gystForm.h>

class mainForm : public QWidget
{
    Q_OBJECT
    QPushButton* Exitbtn;
    QPushButton* addbtn;
    QPushButton* delbtn;
    QPushButton* mostbtn;
    QPushButton* muchbtn;
    QPushButton* benbtn;
    QPushButton* gystbtn;
    QPushButton* tablebtn;
    QLabel* namel;
    QLineEdit* namee;
    QLabel* provl;
    QLineEdit* prove;
    QLabel* amountl;
    QLineEdit* amounte;
    QLabel* pricel;
    QLineEdit* pricee;

```

```

    TableForm winTable;
    themostForm winMost;
    muchForm winMuch;
    BenProForm winBen;
    gystForm winGyst;
public:
    mainForm();
public slots:
    void addRecord();
    void delRecord();
    void mostAmount();
    void muchBuy();
    void benProvider();
    void printGyst();
    void printTable();
};

```

```
#endif // MAINFORM_H
```

Код модуля mainform.cpp:

```

#include "mainform.h"
#include "basefile.h"
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QGridLayout>
mainForm::mainForm()
{
    this->setWindowTitle("Овощная база");
    Exitbtn= new QPushButton("Выйти");
    mostbtn=new QPushButton("Какого товара больше всего?");

```

```

muchbtn=new QPushButton("Сколько кг можно купить");
benbtn=new QPushButton("Самый выгодный пооставщик");
gystbtn=new QPushButton("Гистограмма");
namel=new QLabel("Имя товара");
namee=new QLineEdit;
provl=new QLabel("Поставщик");
prove=new QLineEdit;
amountl=new QLabel("Количество в наличии (кг)");
amounte=new QLineEdit;
pricel=new QLabel("Цена (за 1 кг)");
pricee=new QLineEdit;
addbtn=new QPushButton("Добавить");
delbtn=new QPushButton("Удалить");
tablebtn= new QPushButton("Таблица");
namee->setFocus();
QVBoxLayout* right = new QVBoxLayout ();
right->addWidget(mostbtn);
right->addWidget(muchbtn);
right->addWidget(benbtn);
right->addWidget(gystbtn);
right->addWidget(Exitbtn);
QHBoxLayout* adde =new QHBoxLayout();
adde->addWidget(addbtn);
adde->addWidget(delbtn);
QGridLayout*left = new QGridLayout();
left->addWidget(namel, 1, 1);
left->addWidget(provl, 2, 1);
left->addWidget(amountl, 3, 1);
left->addWidget(pricel, 4, 1);
left->addWidget(namee, 1, 2);

```

```

left->addWidget(prove, 2, 2);
left->addWidget(amounte, 3, 2);
left->addWidget(pricee, 4, 2);
left->addLayout(adde, 5, 2);
left->addWidget(tablebtn, 6, 2);
QHBoxLayout* layout2=new QHBoxLayout();
layout2->insertLayout(0, right, 0);
layout2->insertStretch(1, 0);
layout2->insertLayout(2, left, 0);
setLayout(layout2);
resize(600, 300);//размеры окна
connect(Exitbtn, SIGNAL(clicked(bool)), this, SLOT(close()));
connect(addbtn, SIGNAL(clicked(bool)),this,SLOT(addRecord()));
connect(delbtn, SIGNAL(clicked(bool)),this,SLOT(delRecord()));
//другие формы:
connect(mostbtn, SIGNAL(clicked(bool)),this,SLOT(mostAmount()));
connect(muchbtn, SIGNAL(clicked(bool)),this,SLOT(muchBuy()));
connect(benbtn, SIGNAL(clicked(bool)),this,SLOT(benProvider()));
connect(gystbtn, SIGNAL(clicked(bool)),this,SLOT(printGyst()));
connect(tablebtn, SIGNAL(clicked(bool)),this,SLOT(printTable()));
}

void MainForm::addRecord(){
    baseFile base;
    recType r;
    r.name=namee->text();
    r.prov=prove->text();
    r.amount=amounte->text();
    r.price=pricee->text();
    namee->clear();
    prove->clear();
}

```

```

    amounte->clear();
    pricee->clear();
    base.addRec(r);
    namee->setFocus();
}

void MainForm::delRecord(){
    baseFile base;
    recType r;
    r.name=namee->text();
    r.prov=prove->text();
    //r.amount=amounte->text();
    //r.price=pricee->text();
    namee->clear();
    prove->clear();
    amounte->clear();
    pricee->clear();
    base.delRec(r);
    namee->setFocus();
}

//показать другие формы
void MainForm::mostAmount(){
    winMost.show();
}

void MainForm::muchBuy(){
    winMuch.show();
}

void MainForm::benProvider(){
    winBen.show();
}

void MainForm::printGyst(){

```

```

        winGyst.show();
    }
    void MainForm::printTable(){
        winTable.showAll();
    }

```

Код модуля main.cpp:

```

#include "mainform.h"
#include <QApplication>

#include <QLocale>
#include <QTranslator>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QTranslator translator;
    const QStringList uiLanguages = QLocale::system().uiLanguages();
    for (const QString &locale : uiLanguages) {
        const QString baseName = "practice3_" + QLocale(locale).name();
        if (translator.load(":/i18n/" + baseName)) {
            a.installTranslator(&translator);
            break;
        }
    }
    MainForm w;
    w.show();
    return a.exec();
}

```

Рисунок 1-6 – работающая версия программы:

Овощная база

Какого товара больше всего?

Сколько кг можно купить

Самый выгодный пооставщик

Гистограмма

Выйти

Имя товара

Поставщик

Количество в наличии (кг)

Цена (за 1 кг)

Добавить

Удалить

Таблица

Рис. 1

Какого товара больше...

Посчитать

carrot

Меню

Рис.2

Сколько кг можно купить...

Введите сумму 300

Товар carrot

Поставщик olga

Цена за 1 кг 73

Масса 4

Предыдущий

Вывести

Следующий

Меню

Рис.3

Поиск самого выгодного п...

Введите наименование товара potato

Самый выгодный поставщик sergey

Вывести Меню

Рис.4

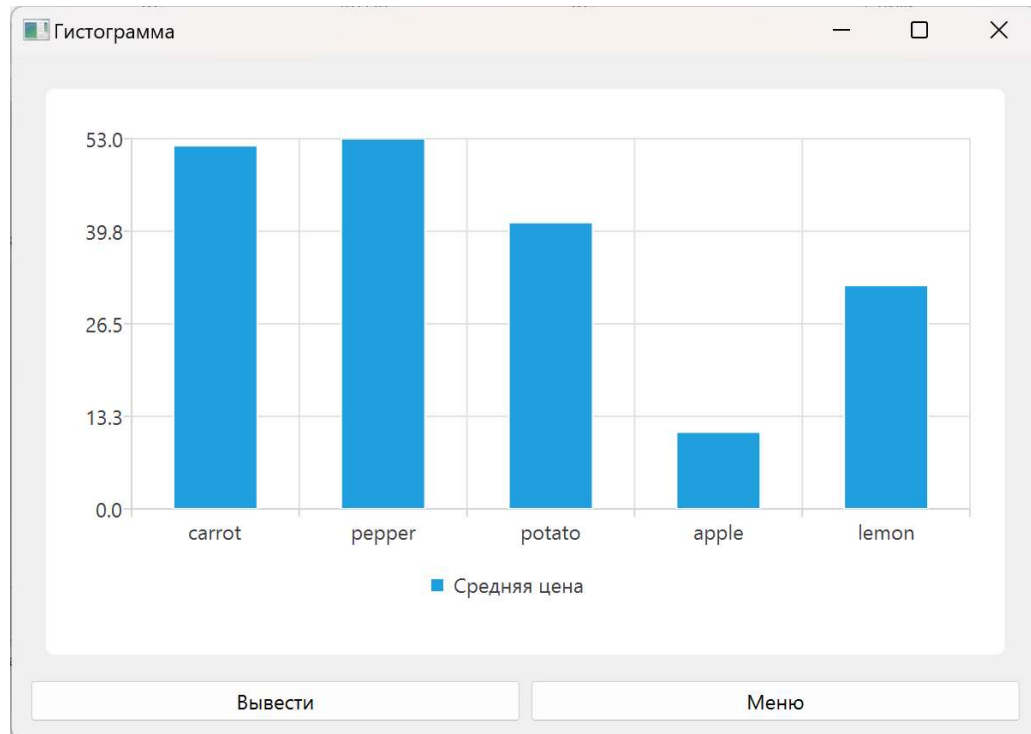


Рис.5

Все записи				
	Имя товара	Производитель	Количество	Цена
1	carrot	peter	34	46
2	carrot	dima	35	54
3	carrot	olga	27	73
4	carrot	masha	13	37
5	pepper	anton	3	56
6	pepper	85	45	65
7	pepper	kolya	7	35
8	potato	sergey	23	41
9	pepper	tom	12	12
10	pepper	tom	15	100
11	apple	peter	11	11
12	lemon	tom	45	32

Меню

Рис.6

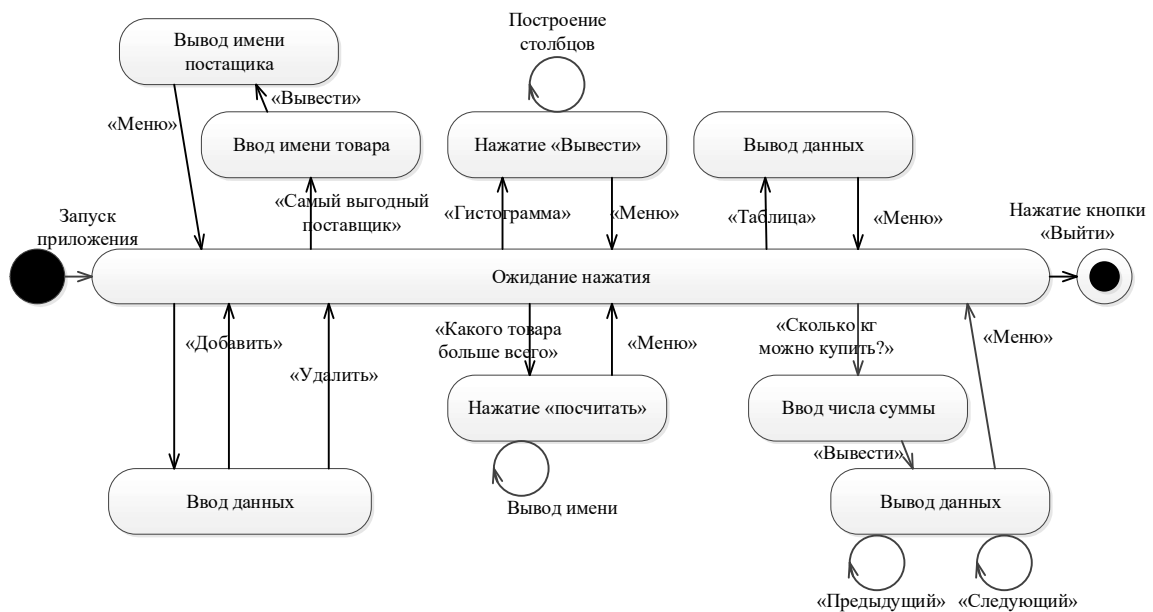


Рисунок 7 – диаграмма состояний интерфейса

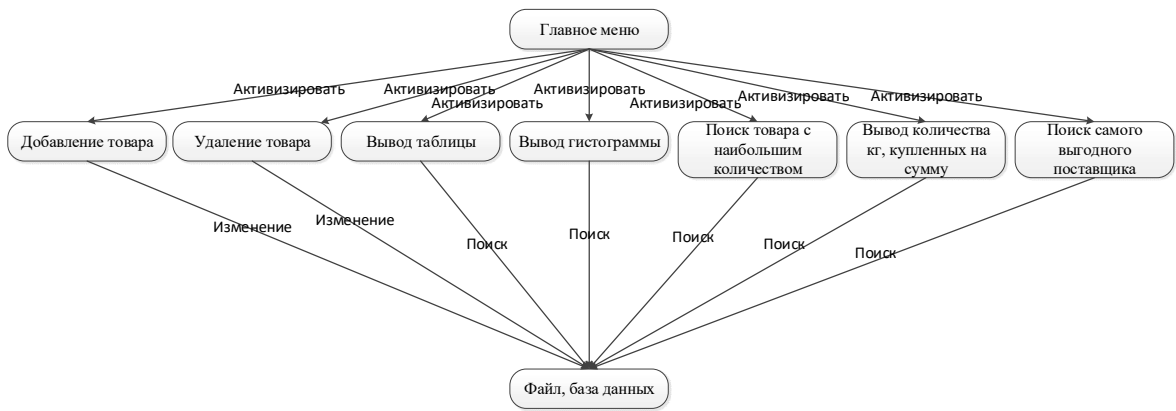


Рисунок 8 – объектная декомпозиция приложения

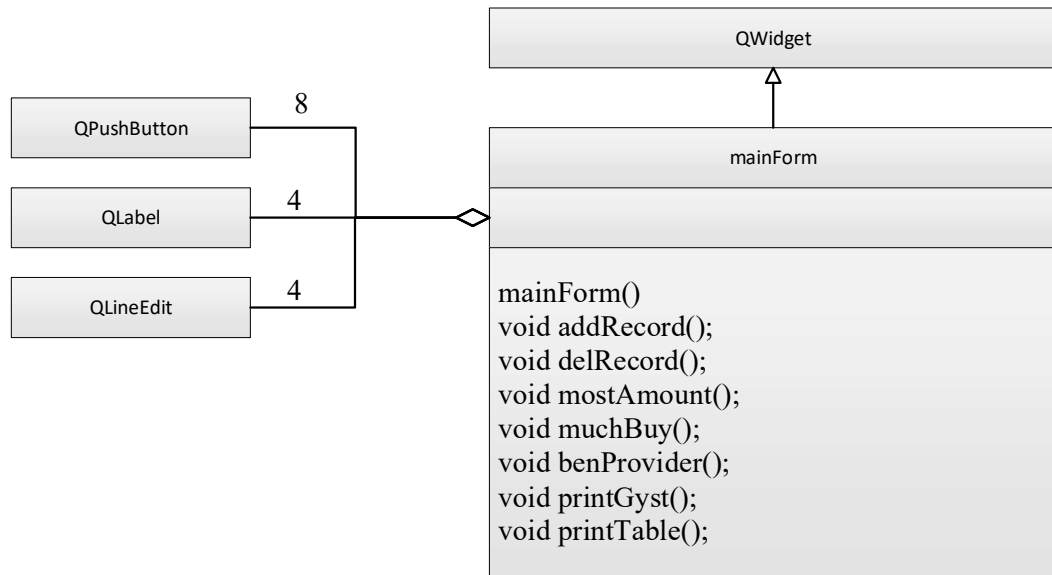


Рисунок 9 – диаграмма класса mainForm

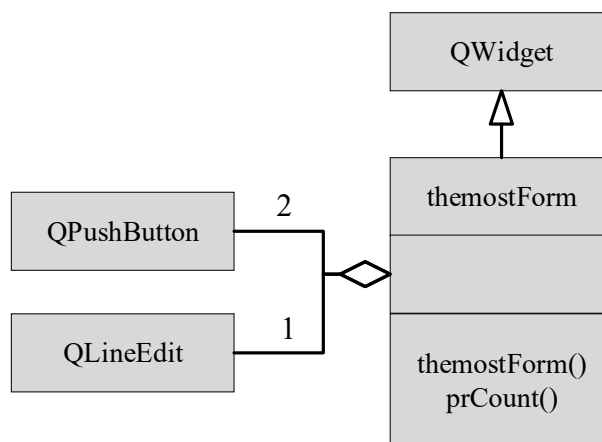


Рисунок 10 – диаграмма класса themostForm

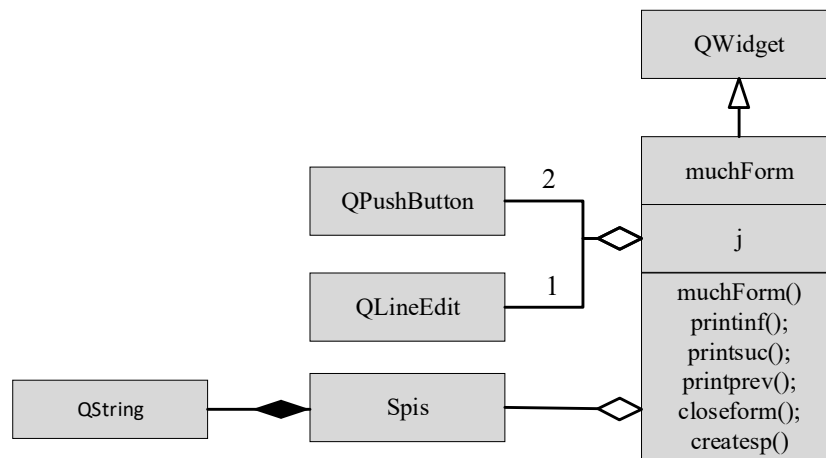


Рисунок 11 – диаграмма класса `muchForm`

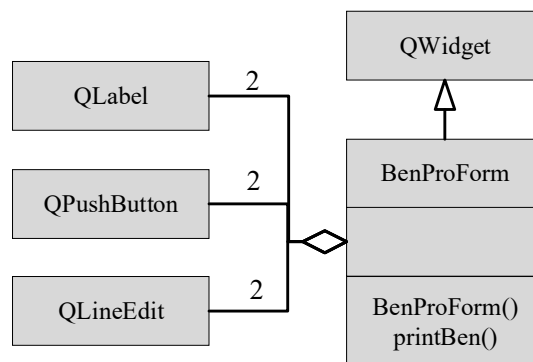


Рисунок 12 – диаграмма класса `BenProForm`

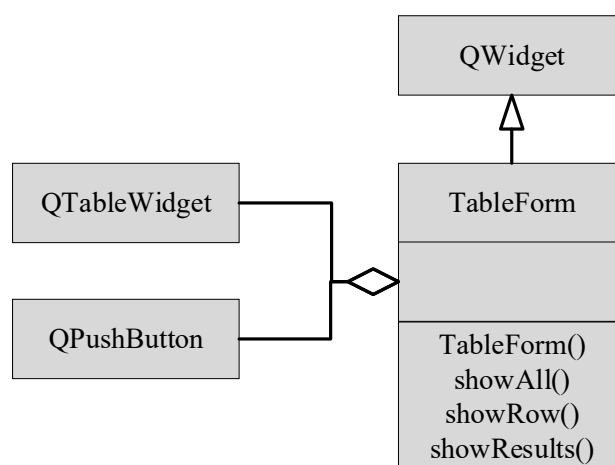


Рисунок 13 – диаграмма класса `TableForm`

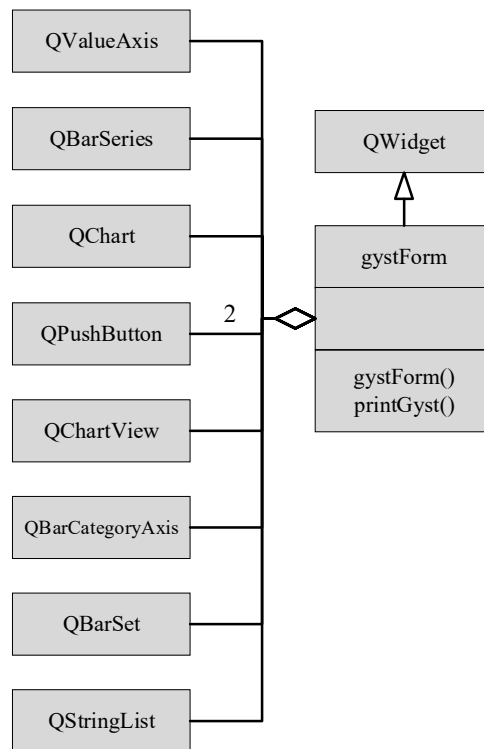


Рисунок 14 – диаграмма класса gystForm

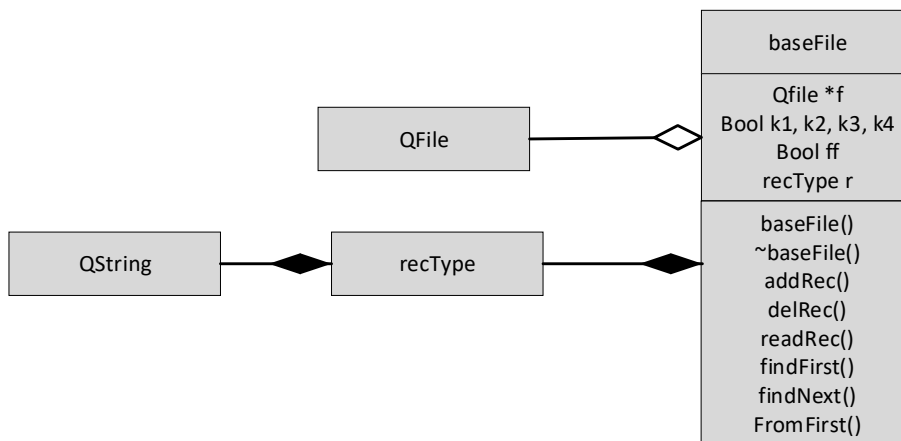


Рисунок 15 – диаграмма класса baseFile

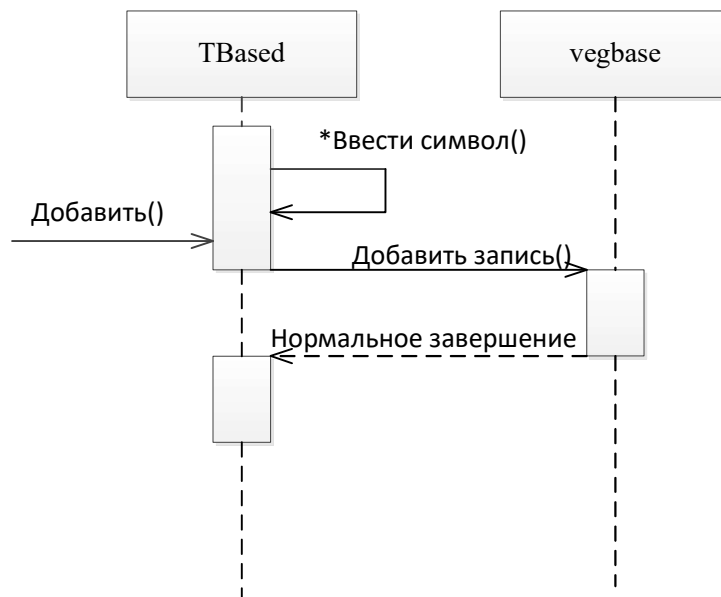


Рисунок 16 – диаграмма последовательностей действий при выполнении функции «Добавление записей»

Вывод: в соответствии с условием задачи было создано многооконное приложение, работающее с базой данных.

Список использованных источников:

1. Г.С. Иванова. Лекции по курсу «Объектно-ориентированное программирование», 2023
2. Г. С. Иванова, Т.Н. Ничушкина, О.В. Платонова. Программирование под Windows в среде Turbo DELPHI 2006. Методические указания по выполнению лабораторной работы по дисциплине Основы программирования, 2011
3. Г.С. Иванова. Создание многооконных приложений с использованием библиотеки Qt, 2013
4. TChart Tutorial: BarSeries [Электронный ресурс]. URL: https://wiki.lazarus.freepascal.org/TChart_Tutorial:_BarSeries/ru
5. Введение в работу с гистограммами в Qt [Электронный ресурс]. URL: <https://radioprogram.ru/post/997>