|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ПО УЧЕБНОЙ ПРАКТИКЕ**

|  |  |
| --- | --- |
| Студент | Люляев Иван Александрович |
|  |  |

|  |  |
| --- | --- |
| Группа | ИУ6-22Б |

|  |  |
| --- | --- |
| Тип практики | Проектно-технологическая практика |

|  |  |
| --- | --- |
| Название  предприятия | НУК ИУ МГТУ им. Н.Э. Баумана |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  | 12.03.2020 | И.А. Люляев |
|  |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |
| Руководитель практики |  |  |  |
|  |  | (Подпись, дата) | (И.О. Фамилия) |

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2020 г.*

**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)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_ИУ6\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Пролетарский

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**ЗАДАНИЕ**

**на учебную практику**

по теме \_\_\_\_Проектирование и реализация программного обеспечения с использованием структурного и объектного подходов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы ИУ6-22Б

Люляев Иван Александрович .

Тип практики Проектно-технологическая практика

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

***Техническое задание:***

**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.

База данных (файл) продуктового магазина содержит сведения о товарах: номер партии, наименование товара, количество (штук), дата поставки, срок годности (в днях). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Показать все сведения о товарах, поступивших с указанной даты.

2. Определить наименования товаров, срок годности которых истекает сегодня.

3. Определить самый скоропортящийся товар (если их несколько, показать все).

4. Построить график изменения объема поступлений заданного товара по датам.

**Код:**

unit Unit3;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Grids, StdCtrls, ExtCtrls;

type

TForm3 = class(TForm)

StringGrid1: TStringGrid;

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Label1: TLabel;

Edit2: TEdit;

Label2: TLabel;

Edit3: TEdit;

Label3: TLabel;

Edit4: TEdit;

Label4: TLabel;

Label5: TLabel;

Edit5: TEdit;

Label6: TLabel;

Edit6: TEdit;

Label7: TLabel;

Edit7: TEdit;

Label8: TLabel;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Button6: TButton;

Button7: TButton;

Button8: TButton;

Button9: TButton;

Image1: TImage;

Button10: TButton;

procedure Button10Click(Sender: TObject);

procedure Button9Click(Sender: TObject);

procedure Button8Click(Sender: TObject);

procedure Button7Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

public

end;

var

Form3: TForm3;

implementation

type

date = record

year, month, day: integer;

end;

product = record

number, quantity, expDate: integer;

name: string[50];

date: date;

end;

f = file of product;

var f1: f;

// glBuf: product;

{$R \*.dfm}

{TForm3}

procedure TForm3.Button2Click(Sender: TObject);

var buf: product;

code: integer;

begin

AssignFile(f1, 'database.dat');

reset(f1);

val(Form3.Edit1.Text, buf.number, code);

buf.name := Form3.Edit2.Text;

val(Form3.Edit3.Text, buf.quantity, code);

val(Form3.Edit4.Text, buf.expDate, code);

val(Form3.Edit5.Text, buf.date.year, code);

val(Form3.Edit6.Text, buf.date.month, code);

val(Form3.Edit7.Text, buf.date.day, code);

Seek(f1, FileSize(f1));

Write(f1, buf);

CloseFile(f1);

Form3.Button3Click(Form3);

end;

procedure TForm3.Button3Click(Sender: TObject);

var s1, s2, s3, s4, s5, s6, s7: string;

cnt: integer;

buf: product;

begin

cnt := 1;

AssignFile(f1, 'database.dat');

reset(f1);

StringGrid1.RowCount := 2;

while not EOF(f1) do begin

Read(f1, buf);

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1;

end;

CloseFile(f1);

end;

procedure TForm3.Button4Click(Sender: TObject);

begin

Close;

end;

procedure TForm3.Button5Click(Sender: TObject);

var curPos, oldPos, cnt, delNum, code: integer;

buf, temp: product;

begin

AssignFile(f1, 'database.dat');

reset(f1);

Seek(f1, filesize(f1)-1);

Truncate(f1);

CloseFile(f1);

Form3.Button3Click(Form3);

end;

procedure TForm3.Button6Click(Sender: TObject);

begin

AssignFile(f1, 'database.dat');

ReWrite(f1);

CloseFile(f1);

Form3.Button3Click(Form3); // ‚˚‚ÂÒÚË Ú‡·ÎËˆÛ

end;

procedure TForm3.Button7Click(Sender: TObject);

var s1, s2, s3, s4, s5, s6, s7: string;

day, month, year, code, cnt: integer;

buf: product;

begin

cnt := 1;

val(Form3.Edit5.Text, year, code);

val(Form3.Edit6.Text, month, code);

val(Form3.Edit7.Text, day, code);

StringGrid1.RowCount := 2;

AssignFile(f1, 'database.dat');

Reset(f1);

while not EOF(f1) do begin

Read(f1, buf);

if (buf.date.year > year) then begin

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1;

end else if (buf.date.year = year) then begin

if (buf.date.month > month) then begin

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1;

end else if (buf.date.month = month) then begin

if (buf.date.day >= day) then begin

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1;

end;

end;

end;

end;

CloseFile(f1);

end;

procedure TForm3.Button8Click(Sender: TObject);

var s1, s2, s3, s4, s5, s6, s7: string;

cnt, minDay: integer;

buf: product;

begin

cnt := 1;

AssignFile(f1, 'database.dat');

reset(f1);

StringGrid1.RowCount := 2;

minDay := 10000;

while not EOF(f1) do begin

Read(f1, buf);

if (buf.expDate < minDay) then minDay := buf.expDate;

end;

Seek(f1, 0);

while not EOF(f1) do begin

Read(f1, buf);

if (buf.expDate = minDay) then begin

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1;

end;

end;

CloseFile(f1);

end;

procedure TForm3.Button9Click(Sender: TObject);

var today, exp: date;

buf: product;

cnt: integer;

s1, s2, s3, s4, s5, s6, s7: string;

begin

today.year := 2020; today.month := 3; today.day := 14;

cnt := 1;

AssignFile(f1, 'database.dat');

reset(f1);

StringGrid1.RowCount := 2;

while not EOF(f1) do begin

Read(f1, buf);

exp.year := buf.date.year;

exp.month := buf.date.month;

if (buf.date.day + buf.expDate) > 31 then begin

exp.month := exp.month + (buf.date.day + buf.expDate) div 31;

if exp.month > 12 then begin

exp.year := exp.year + exp.month div 12;

exp.month := exp.month mod 12;

end;

exp.day := (buf.date.day + buf.expDate) mod 31;

end else exp.day := (buf.date.day + buf.expDate);

if (exp.year = today.year) and (exp.month = today.month) and (exp.day = today.day) then begin

cnt := cnt + 1;

str(buf.number, s1);

s2 := buf.name;

str(buf.quantity, s3);

str(buf.expDate, s4);

str(buf.date.year, s5);

str(buf.date.month, s6);

str(buf.date.day, s7);

StringGrid1.Cells[0, cnt] := s1;

StringGrid1.Cells[1, cnt] := s2;

StringGrid1.Cells[2, cnt] := s3;

StringGrid1.Cells[3, cnt] := s4;

StringGrid1.Cells[4, cnt] := s5;

StringGrid1.Cells[5, cnt] := s6;

StringGrid1.Cells[6, cnt] := s7;

StringGrid1.RowCount := StringGrid1.RowCount + 1; // ‰Ó·‡‚Îˇ˛ ÔÛÒÚÛ˛ ÒÚÓÍÛ

end;

end;

CloseFile(f1);

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

StringGrid1.ColCount := 7; StringGrid1.RowCount := 2;

StringGrid1.RowHeights[1] := 0;

StringGrid1.FixedCols := 0; // Û·Ë‡˛ ÙËÍÒËÓ‚‡ÌÌ˚Â ÒÚÓÎ·ˆ˚

StringGrid1.Cells[0,0] := 'œ‡ÚËˇ'; StringGrid1.ColWidths[0] := 42;

StringGrid1.Cells[1,0] := '“Ó‚‡'; StringGrid1.ColWidths[1] := 80;

StringGrid1.Cells[2,0] := ' ÓÎË˜ÂÒÚ‚Ó';

StringGrid1.Cells[3,0] := '—ÓÍ „Ó‰ÌÓÒÚË'; StringGrid1.ColWidths[3] := 80;

StringGrid1.Cells[4,0] := '√Ó‰'; StringGrid1.ColWidths[4] := 30;

StringGrid1.Cells[5,0] := 'ÃÂÒˇˆ'; StringGrid1.ColWidths[5] := 37;

StringGrid1.Cells[6,0] := 'ƒÂÌ¸'; StringGrid1.ColWidths[6] := 33;

Image1.Canvas.Brush.Color := clWhite;

Image1.Canvas.MoveTo(31,12);

Image1.Canvas.LineTo(25,6);

Image1.Canvas.MoveTo(19,12);

Image1.Canvas.LineTo(25,6);

Image1.Canvas.LineTo(25,260);

Image1.Canvas.LineTo(535,260);

Image1.Canvas.LineTo(529,254);

Image1.Canvas.MoveTo(535,260);

Image1.Canvas.LineTo(529,266);

end;

procedure TForm3.Button10Click(Sender: TObject);

var name, s1, s0, st: string;

i, dx, dy, max: integer;

today: date;

buf: product;

a: array[1..7] of integer;

const xmax = 520;

const maxAxisx = 248;

const delx = 63;

begin

Image1.Canvas.Rectangle(-10, -10, 600, 300);

Image1.Canvas.Brush.Color := clWhite;

Image1.Canvas.MoveTo(31,12);

Image1.Canvas.LineTo(25,6);

Image1.Canvas.MoveTo(19,12);

Image1.Canvas.LineTo(25,6);

Image1.Canvas.LineTo(25,260);

Image1.Canvas.LineTo(535,260);

Image1.Canvas.LineTo(529,254);

Image1.Canvas.MoveTo(535,260);

Image1.Canvas.LineTo(529,266);

today.year := 2020; today.month := 3; today.day := 14;

s0 := '20/03/';

dx := delx;

for i := 1 to 7 do begin

str(today.day, s1);

Image1.Canvas.TextOut(xmax - dx - 23, 265, s0+s1);

Image1.Canvas.MoveTo(xmax - dx, 260);

Image1.Canvas.LineTo(xmax - dx, 255);

dx := dx + delx;

today.day := today.day - 1;

end;

name := Form3.Edit2.Text;

for i := 1 to 7 do a[i] := 0;

AssignFile(f1, 'database.dat');

reset(f1);

while not EOF(f1) do begin

Read(f1, buf);

today.day := 7;

if buf.name = name then begin

for i := 1 to 7 do begin

today.day := today.day + 1;

if buf.date.day = today.day then begin

a[i] := a[i] + buf.quantity;

end;

end;

end;

end;

max := -1;

for i := 1 to 7 do begin

if a[i] > max then max := a[i];

end;

if max = 0 then max := 1; // Á‡˘ËÚ‡ ÓÚ Ó¯Ë·ÍË ‰ÂÎÂÌËˇ Ì‡ ÌÓÎ¸

Image1.Canvas.TextOut(1, 260, '0');

Image1.Canvas.MoveTo(25,260); // 25,260 - Ì.Í. ‚ ÔËÍÒÂÎˇı

dx := 0; // ÒÚ‡Ú ÔÓ ËÍÒÛ

for i := 1 to 7 do begin

dx := dx + delx;

dy := trunc(maxAxisx\*(a[i]/max));

Image1.Canvas.LineTo(15+dx, 260-dy);

Image1.Canvas.Pen.Width := 3;

Image1.Canvas.Ellipse(15+dx-2, 260-(dy-2),15+dx+2, 260-(dy+2));

Image1.Canvas.Pen.Width := 1;

str(a[i], st);

Image1.Canvas.TextOut(1, 260-dy, st);

Image1.Canvas.MoveTo(15+dx, 260-dy);

end;

CloseFile(f1);

end;

procedure TForm3.Button1Click(Sender: TObject);

begin

AssignFile(f1, 'database.dat'); // 'C:\Practice1\database.dat'

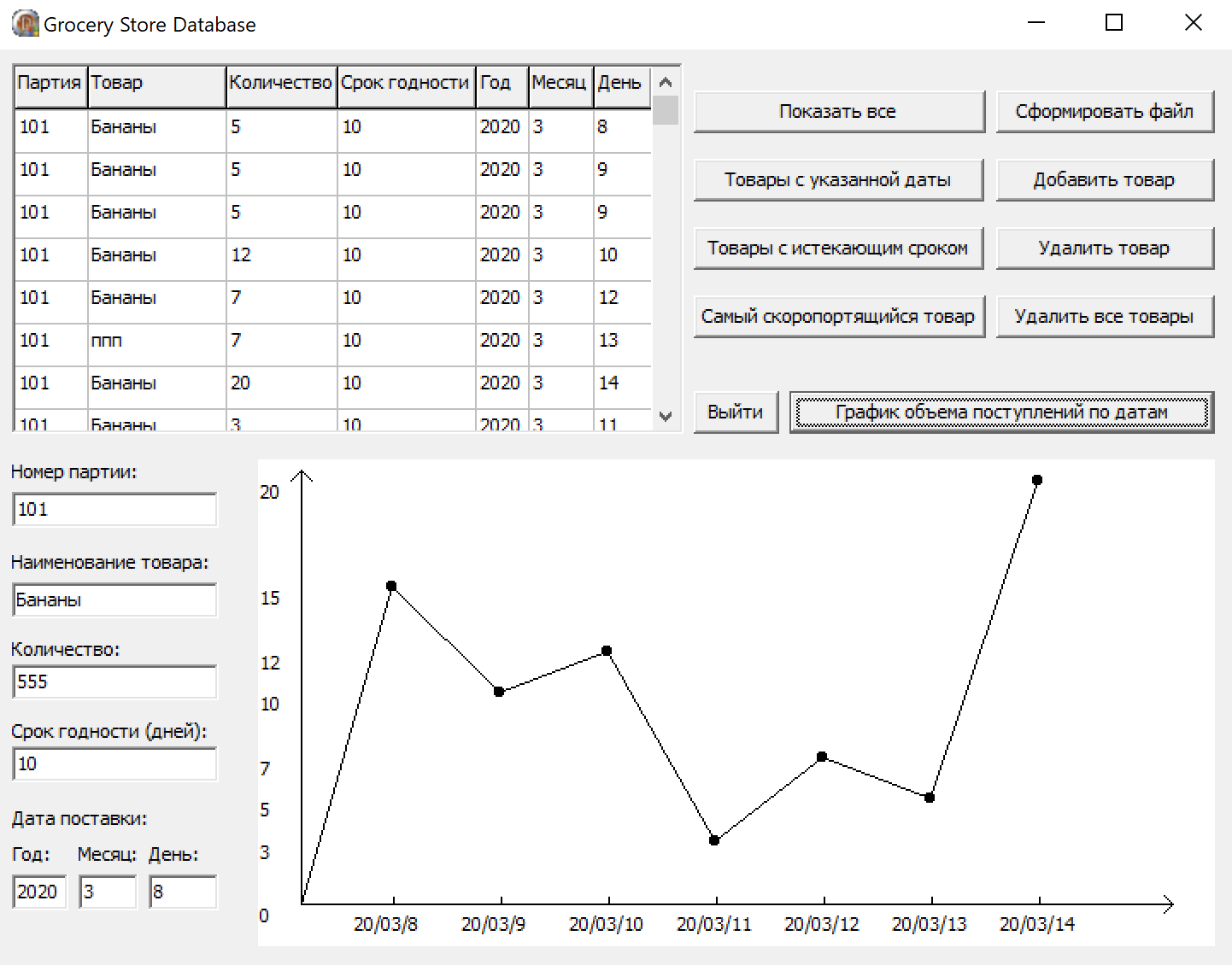
ReWrite(f1);

CloseFile(f1);

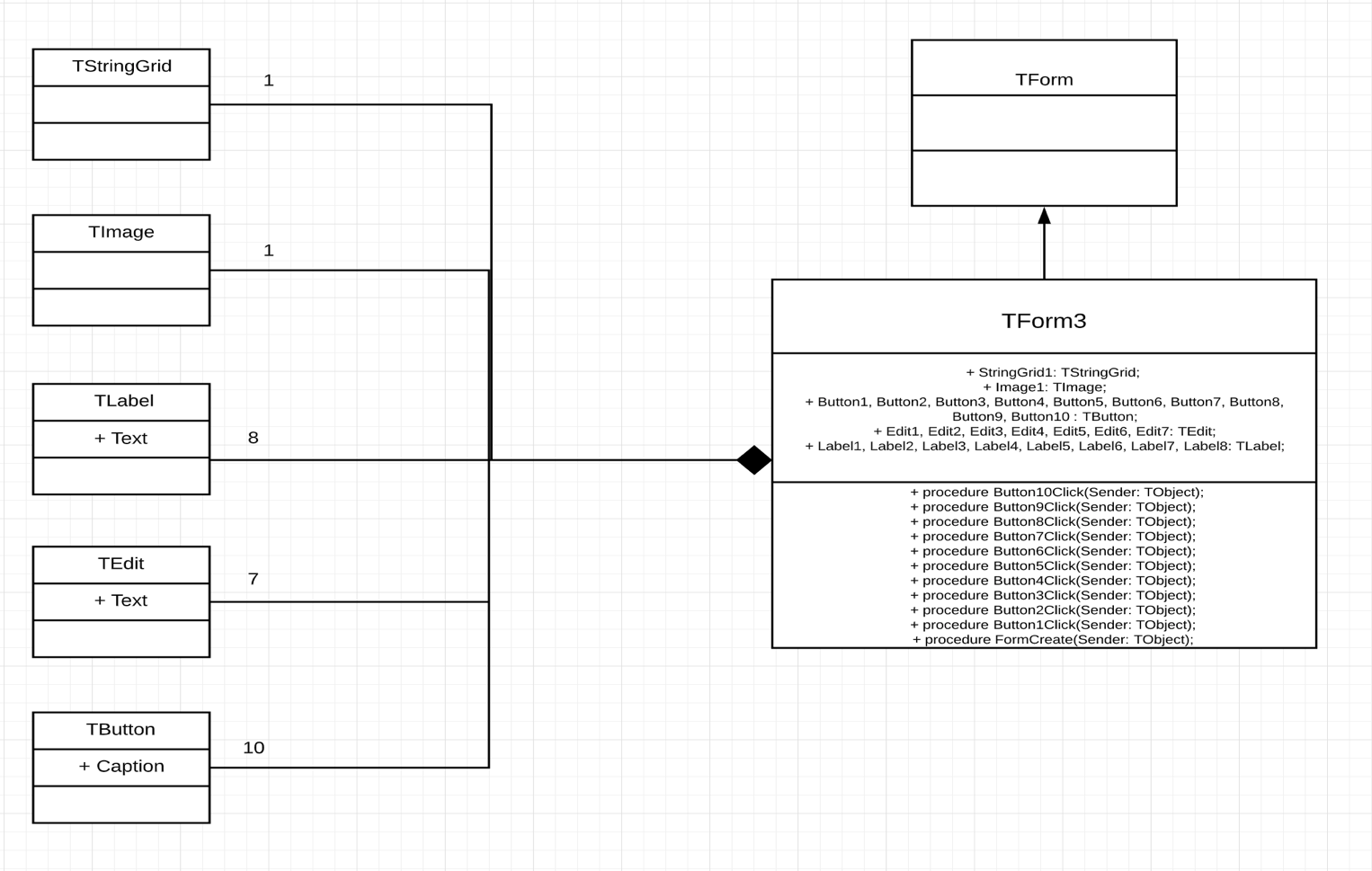
Form3.Button3Click(Form3);

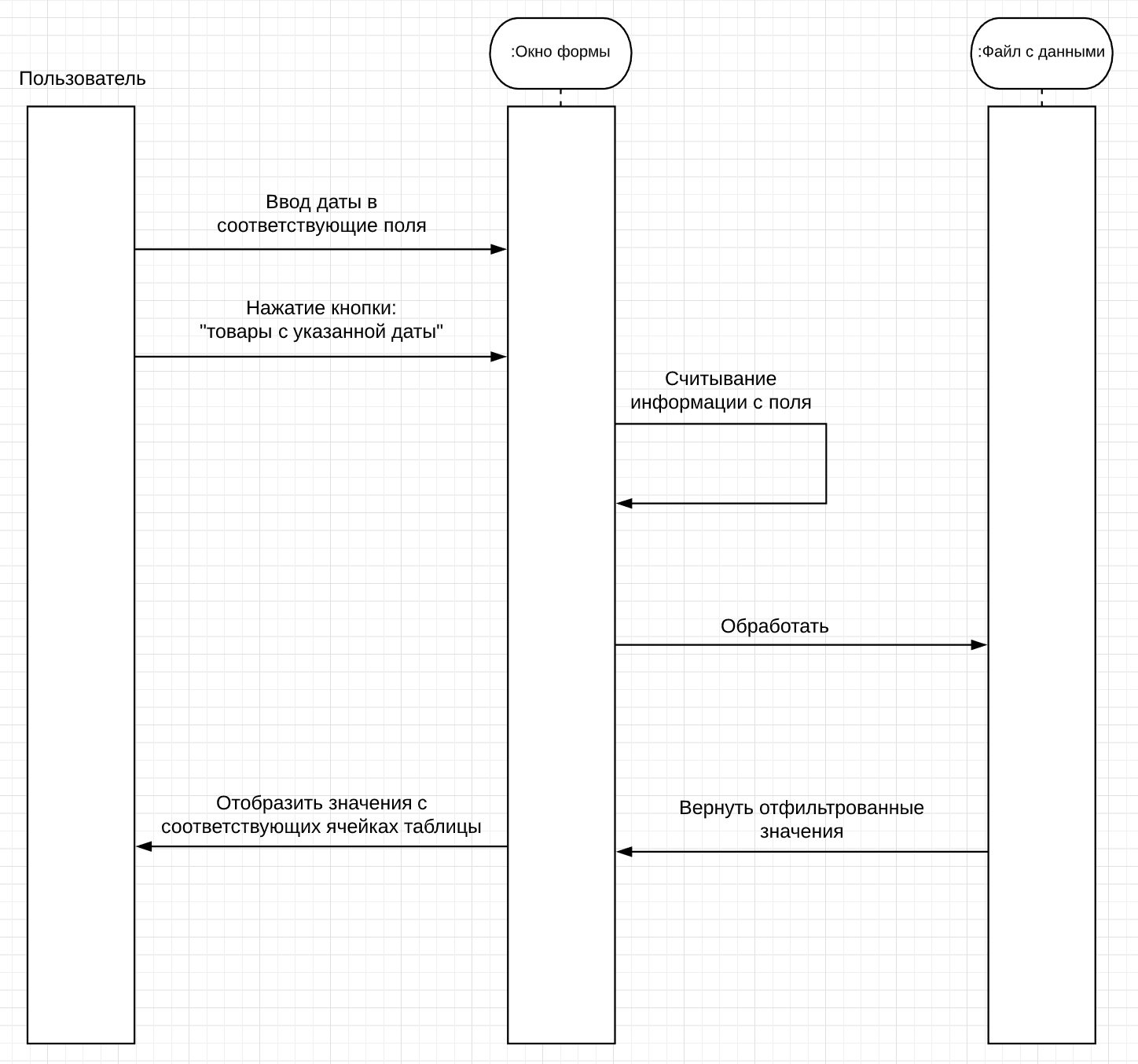
end;

end.

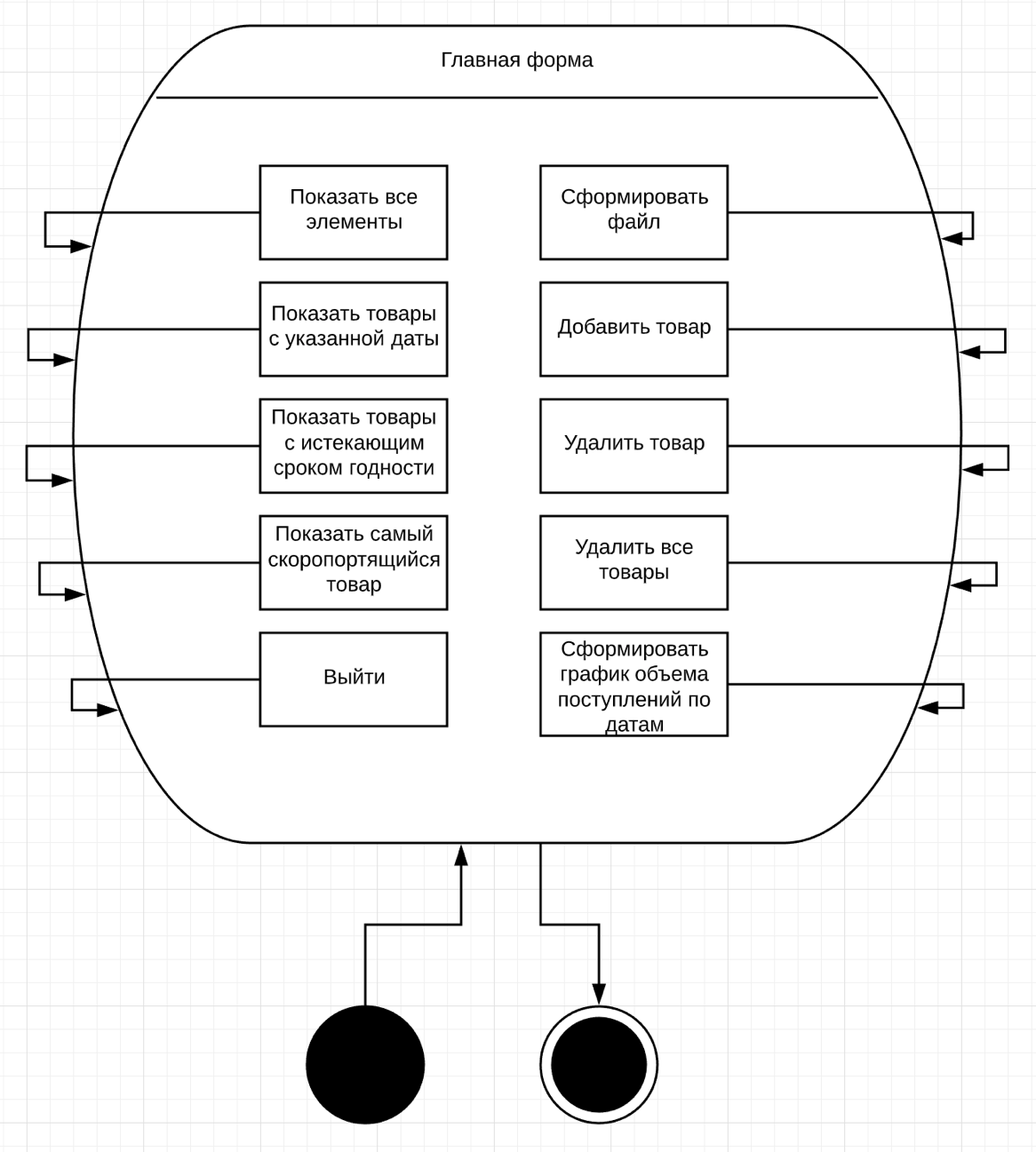
**Интерфейс:**

**Диаграмма классов:**

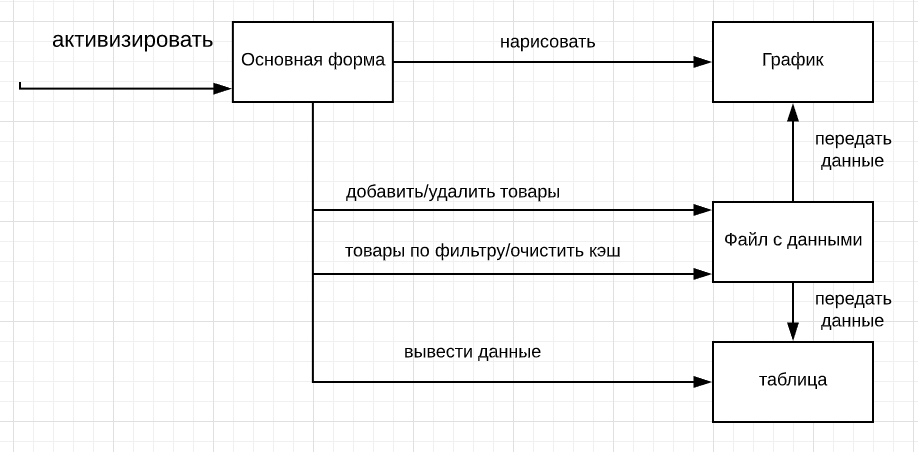


**Диаграмма последовательности определения наименования товаров, срок годности которых истекает сегодня: **

**Диаграмма состояний интерфейса:**

****

**Объектная декомпозиция:**

****

**Вывод:**

Среда разработки Delphi позволяет нам реализовывать событийное программирование при работе с формами, мы можем подключать различные классы, в т.ч. реализующие построение таблиц и вручную рисовать адаптивные графики для наглядного отображения нужных данных из файла с помощью канваса.