«МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ»
федерального государственного бюджетного образовательного учреждения высшего профессионального образования
«Российский государственный торгово-экономический университет»
(МПТ РГТЭУ)

Специальность: 230115 «Программирование в компьютерных системах»

ОТЧЁТ

Учебная практика «Разработка программных модулей»

Тема: «Графический редактор построения блок-схем по готовым файлам»

Листов 28

Выполнил студент:

Мосеин Павел Сергеевич

Группа П-329

_____

«____»_____2014 г.

Руководитель практики:

Соколова Лариса Алексеевна

_____

«___»_____2014 г.

2014 год

Содержание

Введение

Процесс обучения подразумевает проведение лекций с демонстрацией работы и контролем пройденного материала, студент, обучающийся по дисциплинам “Информатика”, ”Основы алгоритмизации”, “Прикладное программирование”, “Системное программирование” в силу психологического состояния или болезни может пропустить информацию выдаваемую преподавателем по конкретной теме. Данное приложение и позволит облегчить труд преподавателю по контролю и ведению тем. Данную программу можно использовать с целью наглядного разбора фрагментов исходного кода, преобразовывать готовые программы в готовые к использованию блок-схемы.

Студенту данная программа помогает в углублённом изучении тем, закрепления пройденного материала, самоконтроля по изученному материалу. Из-за этого обучающие-контролирующее программное обеспечение долгое время будет востребовано.

1. Общая часть

1.1. Цель разработки

Программа разрабатывается для разных групп студентов, как и для тех, кто только начинают изучать программирование, так и для тех, кому необходимо автоматизировать процесс рисования блок-схем, с целью экономии времени, уменьшение затрат на создание документации и т.д.

Также программа полезна для преподавателей, она позволяет закрепить знания, полученные на уроке и наглядно демонстрировать разные алгоритмы и блок-схемы.

Основная функция программы – это создание и редактирование блок-схем из исходного кода

1.2. Средства разработки

1.2.1 Программные средства

Для разработки использовалась среда Embarcadero Delphi XE5 интегрированная среда разработки ПО для Microsoft Windows, в основе среды лежит язык Object Pascal.

Среда предназначена для быстрой (RAD) разработки прикладного ПО для операционных систем Windows. Благодаря уникальной совокупности простоты языка и генерации машинного кода, позволяет непосредственно, и, при желании, достаточно низкоуровневое взаимодействие с операционной системой. Созданные программы не зависимы от стороннего ПО.

1.2.2 Аппаратные средства

Для использования приложения требуется:

- Процессор i7-3630Q
- Оперативная память DDR3 1ГБ
- Жесткий диск 1TB HDD

2. Специальная часть

2.1. Постановка задачи

Разработать программу для использования её преподавателями для обучения и закрепления знаний студента и для студентов желающих лучше программировать, с помощью преобразователя блок-схем.

2.2. Внешняя спецификация

2.2.1 Описание задачи

Данная программа предназначена для преобразования исходного кода программы на одном из яков программирования, в рисунок с готовой блок-схемой

2.2.2 Входные данные

Входные данные	Тип данных	Описание
Исходный код	*.pas	Программа на Pascal
Язык программирования	String	Название языков программирования
Блок цикла	bmp	Элемент блок-схемы, используемый для рисования циклов
Блок условия	bmp	Элемент блок-схемы, используемый для рисования условия
Блок присвоения	bmp	Элемент блок-схемы, используемый для рисования присвоения
Блок начала	bmp	Элемент блок-схемы, используемый для рисования начала и конца блок-схемы

Соединительный блок	bmp	Элемент блок-схемы, используемый для рисования соединительных блоков
Блок ввода	bmp	Элемент блок-схемы используемый для рисования блоков ввода и вывода

2.2.3 Выходные данные

Выходные данные	Тип данных	Описание
Блок схема	bmp	Файл, содержащий блок схему.

2.2.4 Методы

При разработке приложения использовались:

Объектно-ориентированное программирование (ООП) — парадигма программирования, в которой основными концепциями являются понятия объектов и классов.

Среда быстрой разработки приложений (RAD) - концепция создания средств разработки программных продуктов, уделяющая особое внимание быстроте и удобству программирования, созданию технологического процесса, позволяющего программисту максимально быстро создавать компьютерные программы.

Событийное программирование - парадигма программирования, в которой выполнение программы определяется событиями — действиями пользователя (клавиатура, мышь), сообщениями других программ и потоков, событиями операционной системы.

Технология Canvas. Позволяет динамически рисовать фрагменты блок-схем, в зависимости от операторов языка. Позволяет максимально эффективно отрисовывать фигуры.

2.3. Проектирование

2.3.1 Внутреннее проектирование

Рисунок 1. Функциональная схема

2.4. Результаты работы программы

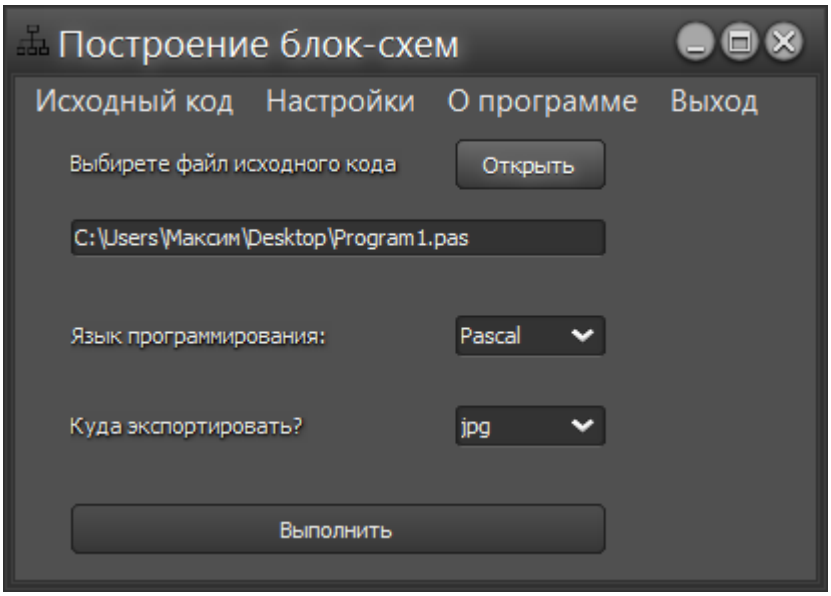

Рисунок 2. Главное окно

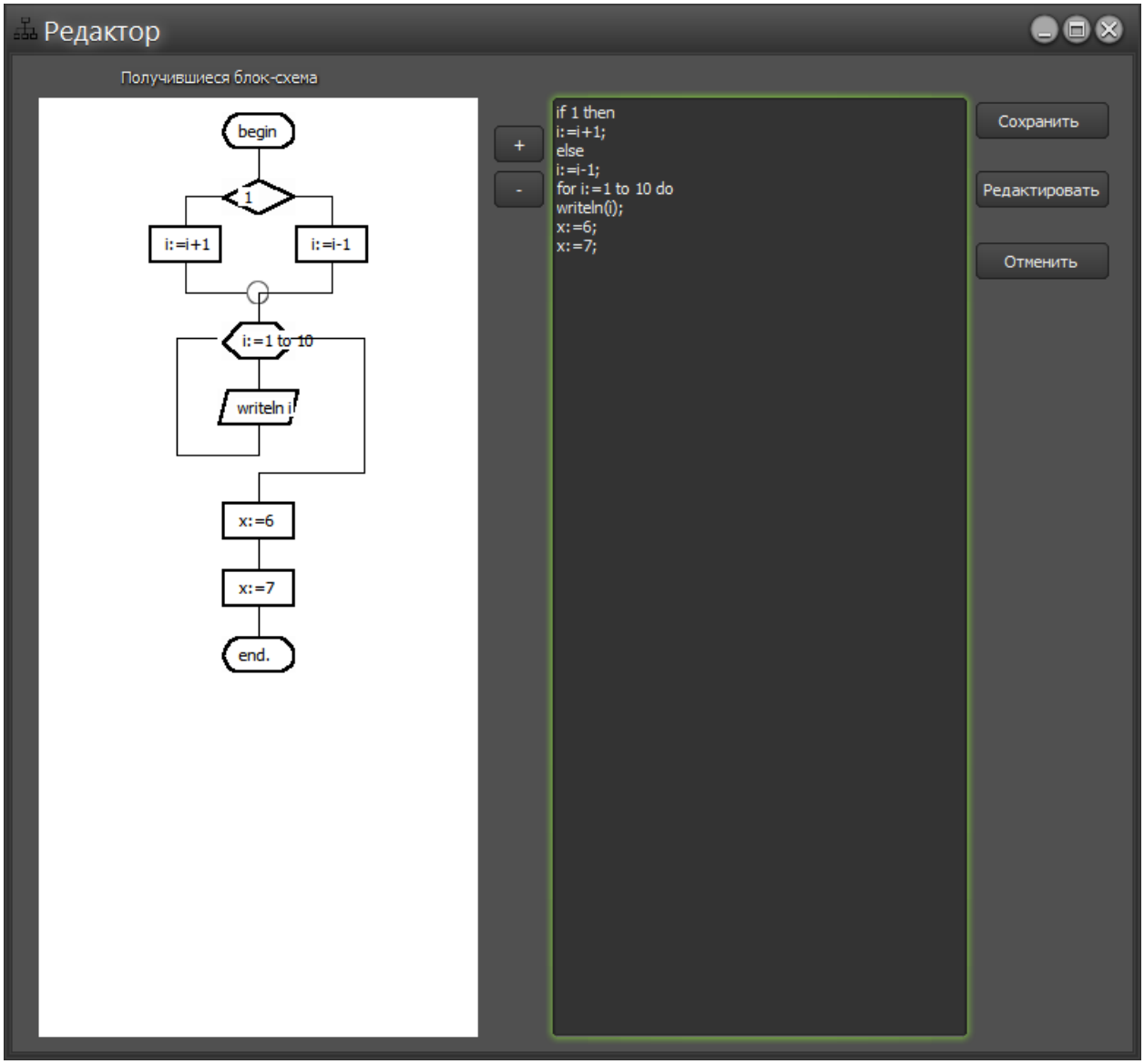

Рисунок 3. Окно просмотра блок-схемы

3. Технологическая часть

3.1. Инструментальные средства разработки

При разработке программы использовалась программа Embarcadero Delphi XE5. В роли языка выступает Object Pascal. Выбор среды объясняется тем, что она позволяет создавать более современные приложения, с помощью объектно-ориентированной модели программирования. В данной программе используются стандартные компоненты Delphi и дополнительна библиотека Alpha Control, с помощью который мы можем изменять и настраивать интерфейс программы.

С помощью текстового редактора или другой инструментальной среды создавались программы на языке Object Pascal, для тестирования их программой.

Для проверки результатов работы программы использовался графический редактор Paint.

3.2. Отладка программы

В данном продукте для отладки использовался интегрированный отладчик Delphi, с помощью которого мы можем контролировать ход выполнения программы. Устанавливать точки остановки, проход программы по шагам, инспектирование и модификация значений переменных.

При работе над программой у нас возникали ошибки:

Ошибки выполнения программы:

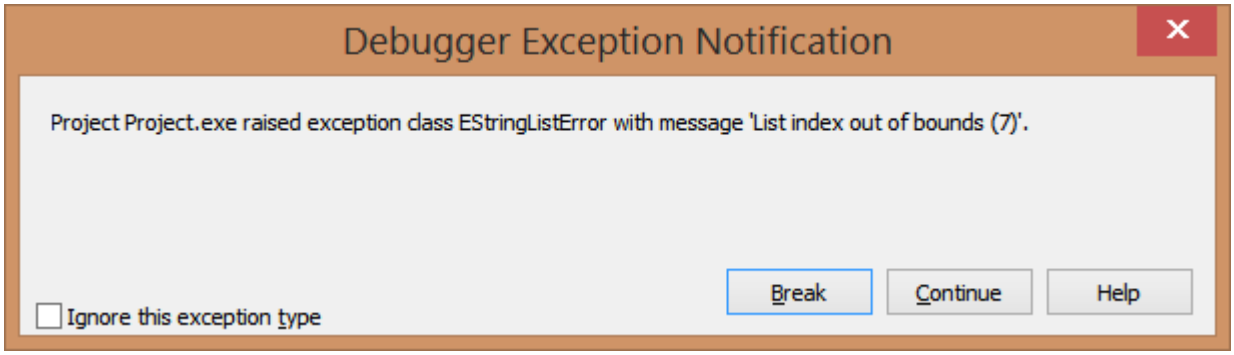

Рисунок 4. Семантическая ошибка

Ошибки компилирования программы:

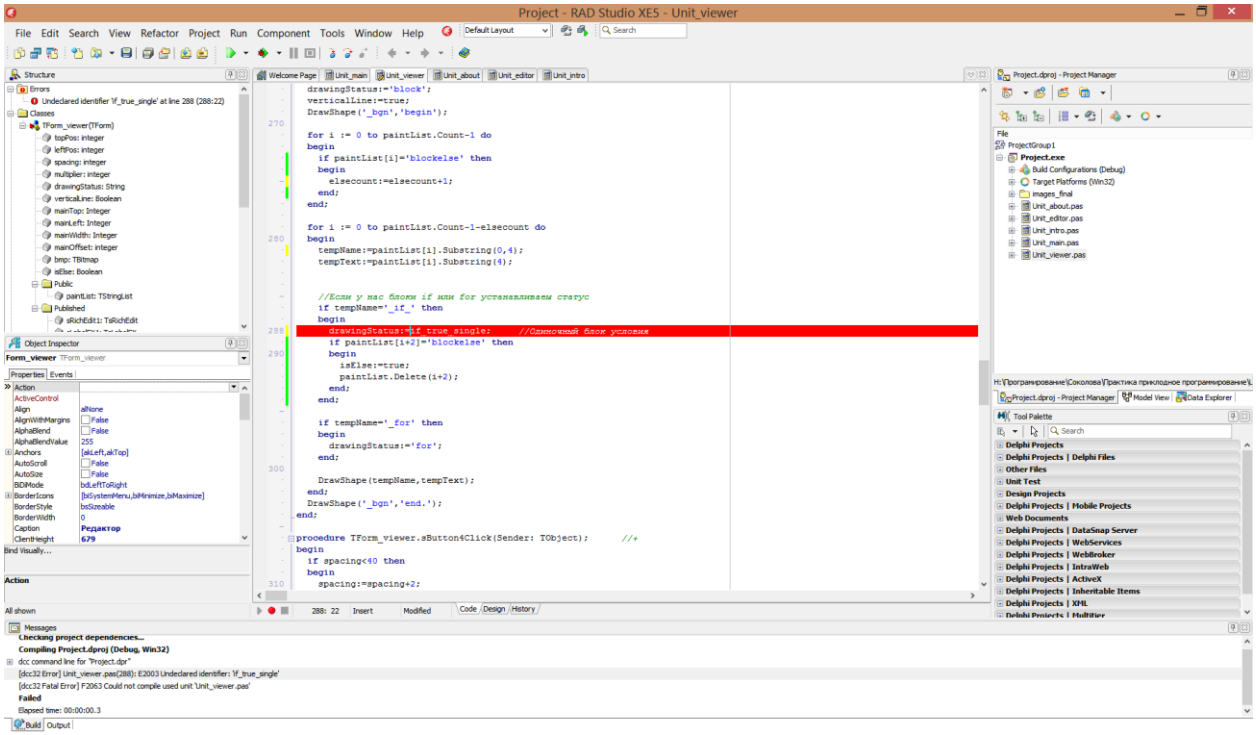

Рисунок 5. Синтаксическая ошибка

3.3 Характеристика программы

Данное приложение включает следующие характеристики.

№ п/п	Название файла	Размер файла	Описание
1	Файл Code2Flow.exe	20 Мбайт	Исполняемый файл
2	Файл code2flow.dproj	30 строк 730 Байт	Проектный модуль
3	Файл Unit_main.pas	112 строк 2 Кбайт	Модуль главной формы
4	Файл Unit_viewer	530 строка 4 Кбайт	Модуль просмотра блок-схемы
5	Файл Unit_editor.pas	40 строк 0.6 Кбайт	Модуль для редактирования блок-схемы
6	Файл Unit_intro.pas	37 строк 0.5 Кбайт	Модуль заставки
7	Файл Unit_about.pas	38 строк 0.6 Кбайт	Модуль «О программе»

3.4. Защитное программирование

Для защиты программы использовалась проверка входных данных.

Заключение

Я справился с основной задачей учебной практики по дисциплине «Прикладное программирование» - сдал индивидуальную работу в виде готового прикладного приложения. При проектировании приложения научился представлять общую структуру приложения, проектировать интерфейс приложения и продумывать действия пользователя.

Первая неделя работы заключалась в проектировании внешнего вида приложения, структуры приложения в виде функциональной схемы, определения входных и выходных данных и его реализации в среде программирования Delphi. В приложении удалось реализовать все функции и определить цель внедрения программы, которые были прописаны в постановке задачи.

На второй неделе практики стал заниматься оптимизацией и упрощением кода приложения. Затем после завершения работы с приложением начался процесс создания отчета по данному приложению в виде пояснительной записки. В ходе написания отчета протестировал программу на все возможные ошибки пользователя. После всех процессов реализации приложения сдал отчет по практике руководителю.

Список используемых материалов.

Методические материалы:

Лекции по дисциплине «Прикладное программирование». Соколова Л.А.

Учебник 100 компонентов общего назначение Delphi 5. А. Я. Архангельский.

Приложение 1. Текст программы.

Номер модуля	Наименование модуля	Описание
1	code2flow.dproj	Проектный модуль
2	Unit_main.pas	Модуль главной формы
3	Unit_veiwer.pas	Модуль просмотра сгенерированной блок-схемы
4	Unit_about.pas	Модуль «О программе»
5	Unit_intro.pas	Модуль заставки
6	Unit_editor.pas	Модуль редактирования блок-схемы

Исходный код программы

code2flow.dproj

```
program code2flow;

{$R *.dres}

uses
Vcl.Forms,
mmsystem,
Unit_main in 'Unit_main.pas' {Form_main},
Unit_viewer in 'Unit_viewer.pas' {Form_viewer},
Unit_intro in 'Unit_intro.pas' {Form_intro},
Unit_editor in 'Unit_editor.pas' {Form_editor},
Unit_about in 'Unit_about.pas' {Form_about};

{$R *.res}

begin
Application.Initialize;
Form_intro:=TForm_intro.create(application);
Form_intro.Show;
Form_intro.Update;
//PlaySound('Alarm01.wav',0,SND_ASYNC);
while Form_intro.Timer1.Enabled do
Application.ProcessMessages;
//Application.MainFormOnTaskbar := True;  //??????
Application.CreateForm(TForm_main, Form_main);
Application.CreateForm(TForm_viewer, Form_viewer);
Application.CreateForm(TForm_editor, Form_editor);
Application.CreateForm(TForm_about, Form_about);
//Application.CreateForm(TForm_intro, Form_intro);
Form_intro.Close;
Form_intro.Free;
Application.Run;
end.
```

Unit_main.pas

```
unit Unit_main;

interface

uses
Winapi.Windows, Winapi.Messages, System.SysUtils,
System.Variants, System.Classes, Vcl.Graphics,
Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.StdCtrls,
System.Actions,
Vcl.ActnList, sLabel, sButton, sSkinManager, sEdit,
sComboBox, sCalculator,
Vcl.ExtCtrls;

type
TForm_main = class(TForm)
MainMenu1: TMainMenu;
N1: TMenuItem;
N2: TMenuItem;
N3: TMenuItem;
J1: TMenuItem;
N6: TMenuItem;
C1: TMenuItem;
OpenDialog1: TOpenDialog;
ActionList1: TActionList;
OpenFile: TAction;
LangChoose: TAction;
N4: TMenuItem;
N5: TMenuItem;
N7: TMenuItem;
formClose: TAction;
sButton1: TsButton;
sComboBox1: TsComboBox;
sComboBox2: TsComboBox;
sEdit1: TsEdit;
sButton2: TsButton;
sSkinManager1: TsSkinManager;
sLabelFX1: TsLabelFX;
sLabelFX2: TsLabelFX;
sLabelFX3: TsLabelFX;
procedure OpenFileExecute(Sender: TObject);
procedure formCloseExecute(Sender: TObject);
procedure sButton1Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure J1Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;

var
Form_main: TForm_main;

implementation

{$R *.dfm}

uses Unit_viewer, Unit_editor, Unit_about;


procedure TForm_main.FormCreate(Sender: TObject);
var menu:tmenuitem;
i:integer;
begin
//TODO переделать в функцию
//* Генерим mainmenu для языков
menu:=TMenuItem.Create(self);
for I := 0 to sComboBox2.Items.Count-1 do
begin
menu.Caption:=sComboBox2.Items[i];
if i=0 then
menu.Checked:=true
else menu.Checked:=false;

N5.add(menu);
end;
//*Mainmenu для экспорта
menu:=TMenuItem.Create(self);
for I := 0 to sComboBox1.Items.Count-1 do
begin
menu.Caption:=sComboBox1.Items[i];
if i=0 then
menu.Checked:=true;
N7.Add(menu);
end;
end;

procedure TForm_main.J1Click(Sender: TObject);
begin
Form_about.Show;
end;

procedure TForm_main.OpenFileExecute(Sender: TObject);
//Открытие файла
begin
OpenDialog1.Execute;
sEdit1.Text:=OpenDialog1.FileName;
end;
```

```
procedure TForm_main.sButton1Click(Sender: TObject);
//Выполнить
begin
if (sEdit1.Text<>'C:\') then
begin
Form_viewer.Show;
hide;
end
else ShowMessage('Откройте файл');
end;

procedure TForm_main.formCloseExecute(Sender: TObject);
//Закрытие
begin
Form_main.Close;
end;

end.
```

Unit_viewer.pas

```
unit Unit_viewer;

interface

uses
Winapi.Windows, Winapi.Messages, System.SysUtils,
System.Variants, System.Classes, Vcl.Graphics,
Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls,
Vcl.ComCtrls, Vcl.ExtCtrls,
acImage, sRichEdit, sButton, sLabel, Vcl.Grids;

type
TForm_viewer = class(TForm)
sRichEdit1: TsRichEdit;
sLabelFX1: TsLabelFX;
sButton1: TsButton;
sButton2: TsButton;
sButton3: TsButton;
sButton4: TsButton;
sButton5: TsButton;
Image1: TImage;
RichEdit1: TRichEdit;
SaveDialog1: TSaveDialog;
procedure Button2Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action:
TCloseAction);
procedure Image1MouseDown(Sender: TObject; Button:
TMouseButton;
Shift: TShiftState; X, Y: Integer);
procedure risovanie;
procedure DrawShape(figureName:string;text:string);
procedure FormActivate(Sender: TObject);
procedure sButton4Click(Sender: TObject);
procedure sButton5Click(Sender: TObject);
procedure SplitCode;
function offset(num:integer):integer;
procedure sButton1Click(Sender: TObject);
procedure addBeginEnd;
procedure addnullblock(i:integer);
procedure sRichEdit1KeyUp(Sender: TObject; var Key: Word;
Shift: TShiftState);
procedure sButton3Click(Sender: TObject);

private
topPos:integer;
leftPos: integer;
spacing:integer;
multiplier:integer;
drawingStatus:String;
verticalLine:Boolean;
mainTop,mainLeft,mainWidth:Integer;
mainOffset:integer;
bmp: TBitmap;
isElse:Boolean;
{ Private declarations }
public
paintList:TStringList;
{ Public declarations }
end;

var
Form_viewer: TForm_viewer;

implementation

{$R *.dfm}

uses Unit_editor, Unit_main;

function tform_viewer.offset(num:integer):integer;  //Задел на
будующее
begin

Result:=round(Image1.Width/2)-num*50;
end;

procedure Tform_viewer.DrawShape(figureName:string;text:
string);
var

blockSize,blockSizeConn:TRect;

// tempTopPos:integer;

begin
with Image1 do
begin
//Рисуем блоки
{ bmp.LoadFromResourceName(HInstance,figureName);

leftPos:=Offset(mainOffset)-round(bmp.Width/2);
//Выравниваем

blockSize:=blockSize.Create(
leftPos-multiplier,
topPos-multiplier,
leftPos+bmp.Width+multiplier,
topPos+bmp.Height+multiplier
);
Canvas.StretchDraw(blockSize,bmp);

Canvas.TextOut(leftPos+round(bmp.Width/4),topPos+Round(bmp
.Height/4),text); //  TODO выравнять текст

//Рисуем вертикальный линии
topPos:=topPos+bmp.Height;

Canvas.MoveTo(leftPos+round(bmp.Width/2),TopPos+multiplier)
;
topPos:=topPos+spacing;
Canvas.LineTo(leftPos+round(bmp.Width/2),TopPos);
}
if (drawingStatus<>'none') then
begin
if figureName='nullblock' then                //TODO
begin

Canvas.MoveTo(leftPos+round(bmp.Width/2),TopPos+multiplier)
;
topPos:=topPos+spacing;
Canvas.LineTo(leftPos+round(bmp.Width/2),TopPos);
topPos:=topPos-bmp.Height;
end
else
begin
bmp.LoadFromResourceName(HInstance,figureName);
//Рисуем фигуру и текст в ней
if not drawingStatus.Contains('if') then
//Выравнивание по середине.
leftPos:=round(Image1.Width/2)-round(bmp.Width/2);
```

```
blockSize:=blockSize.Create(
leftPos-multiplier,
topPos-multiplier,
leftPos+bmp.Width+multiplier,
topPos+bmp.Height+multiplier
);

Canvas.StretchDraw(blockSize,bmp);

Canvas.TextOut(leftPos+round(bmp.Width/4),topPos+Round(bmp
.Height/4),text);
end;
end;


//обработчик блока if

if drawingStatus='if_true_complete' then       //Завершающий
блок if  TRUE
begin
topPos:=topPos+bmp.Height;
leftPos:=leftPos+round(bmp.Width/2);
Canvas.MoveTo(leftPos,topPos+multiplier);     // |_
topPos:=topPos+spacing;
Canvas.LineTo(leftPos,topPos);
leftPos:=leftPos+round(leftPos*2/3);
Canvas.LineTo(leftPos-spacing,topPos);
//рисуем соеденитель

bmp.LoadFromResourceName(HInstance,'conn');

leftPos:=round(Image1.Width/2)-round(bmp.Width/2);
topPos:=topPos-Round(bmp.Height/2);
blockSizeConn:=blockSizeConn.Create(
leftPos-round(multiplier/2),
topPos-round(multiplier/2),
leftPos+round(bmp.Width)+round(multiplier/2),
topPos+Round(bmp.Height)+round(multiplier/2)
);

Canvas.StretchDraw(blockSizeConn,bmp);

if isElse  then
drawingStatus:='if_else_single'
else
drawingStatus:='block';
verticalLine:=true;
end;

if drawingStatus='if_true_single' then        //Одиночный блок
if True
begin
// _
// |
mainLeft:=leftPos;
mainWidth:=bmp.Width;
topPos:=topPos+round(bmp.Height/2);
mainTop:=topPos;
Canvas.MoveTo(leftPos,topPos);
leftPos:=leftPos-round(bmp.Width/2);

Canvas.LineTo(leftPos,TopPos);
//Canvas.TextOut(leftPos-10,topPos-10,'TRUE');
topPos:=topPos+spacing;
Canvas.LineTo(leftPos,TopPos);
leftPos:=leftPos-round(bmp.Width/2);
drawingStatus:='if_true_complete';
verticalLine:=false;
end;


if drawingStatus='if_else_complete' then       //Завершающий
блок if  FALSE
begin
topPos:=topPos+bmp.Height;
leftPos:=leftPos+round(bmp.Width/2);
Canvas.MoveTo(leftPos,topPos+multiplier);     // |_
topPos:=topPos+spacing;
Canvas.LineTo(leftPos,topPos);
leftPos:=round(Image1.Width/2);               //-conn.width/2
Canvas.LineTo(leftPos,topPos);
leftPos:=leftPos-round(bmp.Width/2);          //как-то
выравниваем
topPos:=topPos-bmp.Height;
drawingStatus:='block';
verticalLine:=true;
end;


if drawingStatus='if_else_single' then
//Одиночный else
begin
// _
// |
//topPos:=mainTop+round(bmp.Height/2);
topPos:=mainTop;                              //координаты
половины блока if
leftPos:=round(Image1.Width/2)+Round(mainWidth/2);
Canvas.MoveTo(leftPos,topPos);
leftPos:=leftPos+round(mainWidth/2);

Canvas.LineTo(leftPos,TopPos);
//Canvas.TextOut(leftPos-10,topPos-10,'TRUE');
topPos:=topPos+spacing;
Canvas.LineTo(leftPos,TopPos);
leftPos:=leftPos-round(mainWidth/2);
drawingStatus:='if_else_complete';
verticalLine:=false;
end;


if drawingStatus='for_end' then
begin
topPos:=topPos+bmp.Height;

Canvas.MoveTo(leftPos+round(bmp.Width/2),TopPos+multiplier)
;
topPos:=topPos+spacing;

Canvas.LineTo(leftPos+round(bmp.Width/2),TopPos);
leftPos:=leftPos-round(bmp.Width/2);
leftPos:=leftpos-multiplier;
Canvas.LineTo(leftPos,TopPos);
topPos:=topPos-(bmp.Height+spacing)*2+10;
Canvas.LineTo(leftPos,topPos);
leftPos:=leftPos+round(bmp.Width/2);
leftPos:=leftpos+multiplier;
Canvas.LineTo(leftPos-multiplier,TopPos);
verticalLine:=false;
drawingStatus:='for_finish';
end;


if drawingStatus='for_finish' then
```

```
begin
bmp.LoadFromResourceName(HInstance,'_for');
leftPos:=leftPos+bmp.Width;
Canvas.MoveTo(leftPos+multiplier,TopPos);
leftPos:=leftPos+bmp.Width;
Canvas.LineTo(leftPos+multiplier,TopPos);
topPos:=topPos+(bmp.Height+spacing)*2;
Canvas.LineTo(leftPos+multiplier,TopPos+multiplier);
leftPos:=round(Image1.Width/2);
Canvas.LineTo(leftPos,topPos+multiplier);
toppos:=toppos-bmp.Height;
leftpos:=leftpos-round(bmp.Width/2);
verticalLine:=true;
drawingStatus:='block';
end;

if drawingStatus='for' then
begin
drawingStatus:='for_end';
end;

if (text<>'end.') and (drawingStatus<>'none') and
(verticalLine)then
begin
topPos:=topPos+bmp.Height;

Canvas.MoveTo(leftPos+round(bmp.Width/2),TopPos+multiplier)
;
topPos:=topPos+spacing;
Canvas.LineTo(leftPos+round(bmp.Width/2),TopPos);
end;
end;
end;


procedure Tform_viewer.addnullblock(i:integer);
begin
// paintList.Insert(i,);
end;

procedure TForm_viewer.risovanie;
var i:integer;
tempName,tempText:string;
elsecount:integer;
begin
RichEdit1.Clear;
RichEdit1.Lines.AddStrings(paintList);
drawingStatus:='block';
verticalLine:=true;
DrawShape('_bgn','begin');
// DrawShape('nullblock','');

i:=0;
while i<paintlist.Count do
begin
tempName:=paintList[i].Substring(0,4);
tempText:=paintList[i].Substring(4);


//Если у нас блоки if или for устанавливаем статус
if tempName='_if_' then
begin
drawingStatus:='if_true_single';    //Одиночный блок условия
{
while not paintList[i+1].Contains('_if_') do
begin

end;
}
if paintList[i+4].Contains('noneelse') then
begin
isElse:=true;
paintList.Delete(i+4);
end;

end;

//addnullblock(i);       ???

if tempName='_for' then
begin
drawingStatus:='for';
end;

if tempName<>'none' then
DrawShape(tempName,tempText);

i:=i+1;
end;
DrawShape('_bgn','end.');
end;

procedure TForm_viewer.sButton1Click(Sender: TObject);
//SaveDialogToBmp
begin
SaveDialog1.Filter := GraphicFilter(TBitmap);
SaveDialog1.Execute;
Image1.Picture.SaveToFile(SaveDialog1.FileName);
end;

procedure TForm_viewer.sButton3Click(Sender: TObject);
begin
Image1.Picture:=nil;
sRichEdit1.Text:='';
Form_main.Show;
Form_viewer.hide;
end;

procedure TForm_viewer.sButton4Click(Sender: TObject);      //+
begin
if spacing<40 then
begin
spacing:=spacing+2;
multiplier:=multiplier+1;
topPos:=10;
if topPos<=multiplier then
topPos:=multiplier;
Image1.Picture:=nil;
risovanie;
Image1.Repaint;
end;
end;

procedure TForm_viewer.sButton5Click(Sender: TObject);      //-
begin
if spacing>=20 then
begin
spacing:=spacing-2;
multiplier:=multiplier-1;
topPos:=10;
if topPos<=multiplier then
topPos:=multiplier;
Image1.Picture:=nil;
risovanie;
end;
end;
```

```
procedure TForm_viewer.Button2Click(Sender: TObject);
begin
Form_editor.Image1.Picture:=Image1.Picture;
Form_editor.sRichEdit1.Lines:=sRichEdit1.Lines;
Form_editor.ShowModal;
end;

procedure TForm_viewer.addBeginEnd;
var i:integer;
begin
for i:=0 to paintList.Count-1 do
begin
if (paintList[i].Contains('_if_')) and not
(paintList[i+1].Contains('noneBeginBlock')) then
begin
paintList.Insert(i+1,'noneBeginBlock_true');
paintList.Insert(i+3,'noneEndBlock_true');
end;

if (paintList[i].Contains('noneelseblock')) and not
(paintList[i+1].Contains('noneBeginBlock')) then
begin
paintList.Insert(i+1,'noneBeginBlock_else');
paintList.Insert(i+3,'noneEndBlock_else');
end;

if (paintList[i].Contains('_for')) and not
(paintList[i+1].Contains('noneBeginBlock')) then
begin
paintList.Insert(i+1,'noneBeginBlock');
paintList.Insert(i+3,'noneEndBlock');
end;




end;
end;


procedure tform_viewer.SplitCode;
var
strList:TStringList;
i:integer;
Splitted: TArray<String>;
tempStr:string;
tempPos:integer;
startOfComment,endOfComment:integer;
tempText:string;
begin
strList:=TStringList.Create;
strList.LoadFromFile(Form_main.OpenDialog1.FileName);
//заполняем richedit
for i:=0 to strList.IndexOf('begin')-1 do          //Вырезаем все до
begin
begin
strList.Delete(0);
end;

//Удаляем все коментарии TODO
startOfComment:=strList.Text.IndexOf('{');
endOfComment:=strList.Text.IndexOf('}');
tempText:=strList.Text;
Delete(tempText,startOfComment,endOfComment);
strList.Text:=tempText;

sRichEdit1.Lines.AddStrings(strList);//Добавляем в richEdit
//Caption:='';
for I := 0 to strList.Count-1 do
begin

if strList[i]='begin' then
paintList.Add('noneBeginBlock');

if (strList[i]='end;') or (strList[i]='end') or (strList[i]='end.') then
paintList.Add('noneEndBlock');

if strList[i].Contains(':=') and strList[i].Contains(';') then
//действие  добавляем
begin
tempStr:='_rct'+strList[i];
tempPos:=tempStr.IndexOf(';');
if (tempPos=length(tempStr)-1) then          //Удаляем
ковычку из строки
Delete(tempStr,tempPos+1,1);
paintList.add(tempStr);
end;

if (strList[i].Contains('write(')) or (strList[i].Contains('read('))
//Ввод вывод переменных
or (strList[i].Contains('writeln(')) or
(strList[i].Contains('readln(')) then
begin
tempStr:='inpt';//+strList[i];
Splitted:=strList[i].Split(['(']);
tempStr:=tempStr+Splitted[0]+' ';
Splitted:=Splitted[1].Split([')']);
tempStr:=tempStr+Splitted[0];
paintList.add(tempstr);
end;

if strList[i].Contains('if') and strList[i].Contains('then') then
begin
tempStr:='_if_';
Splitted:=strList[i].Split(['if'], none);
Splitted:=Splitted[1].Split(['then'],none);
tempStr:=tempStr+Splitted[0];
paintList.Add(tempStr);
end;

if strList[i]='else' then
paintList.Add('noneelse');


if strList[i].Contains('for') and strList[i].Contains('to') and
strList[i].Contains('do') then
begin
tempStr:='_for';
Splitted:=strList[i].Split(['for'], none);
Splitted:=Splitted[1].Split(['do'], none);
tempStr:=tempStr+Splitted[0];
paintList.Add(tempStr);
end;




end;

end;

procedure TForm_viewer.sRichEdit1KeyUp(Sender: TObject; var
Key: Word;
```

```
Shift: TShiftState);
begin
if key=VK_TAB then
begin

//RichEdit1.Paragraph.FirstIndent;
end;
end;

procedure TForm_viewer.FormActivate(Sender: TObject);
begin
bmp := TBitmap.Create;
topPos:=10;
spacing:=20;
mainOffset:=0;
paintList:=TStringList.Create;


splitCode;
addBeginEnd;
risovanie;
end;

procedure TForm_viewer.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Form_main.Close;
FreeAndNil(paintList);

end;

procedure TForm_viewer.Image1MouseDown(Sender: TObject;
Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
begin
Caption:='x='+IntToStr(x)+' y='+inttostr(y);
end;

end.
```

Unit_about.pas

```
unit Unit_about;

interface

uses
Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Imaging.GIFImg, Vcl.ExtCtrls,
Vcl.StdCtrls, sButton;

type
TForm_about = class(TForm)
sButton1: TsButton;
Image1: TImage;
procedure sButton1Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;

var
Form_about: TForm_about;

implementation

{$R *.dfm}

uses Unit_main;

procedure TForm_about.sButton1Click(Sender: TObject);
begin
Form_about.Close;
Form_main.Show;

end;

end.
```

Unit_intro.pas

```
unit Unit_intro;

interface

uses
Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.jpeg, acImage,mmsystem;

type
TForm_intro = class(TForm)
sImage1: TsImage;
Timer1: TTimer;
procedure Timer1Timer(Sender: TObject);
private
i:integer;
{ Private declarations }
public
{ Public declarations }
end;

var
Form_intro: TForm_intro;

implementation

{$R *.dfm}

uses Unit_main;

procedure TForm_intro.Timer1Timer(Sender: TObject);
var i:integer;
begin
Timer1.Enabled:=false;
end;

end.
```

Приложение 2. Руководство пользователя.

Системные требования:

ОС: Windows XP, Windows 7,Windows 8.

Процессор: Intel Core I3.

Оперативная память: 1ГБ.

Жесткий диск: 100 МБ свободного места.

В комплект поставки входит программа установки программы на жесткий диск компьютера, которую нужно запустить для установки данного приложения. При запуске инсталлятора нужно выбрать путь установки программы. После установки на рабочем столе появится ярлык на программу с названием приложения, а приложение будет храниться в выбранном каталоге. Запуск приложения происходит при нажатии на Code2Flow.exe. Также внутри папки находятся другие каталоги с файлами, необходимыми для работы данного приложения.

При запуске приложения запускается заставка (рис. 8), которая работает 5 секунд. За ней появляется главное окно (рис. 6) в котором можно выбрать файл с исходным кодом, который будет превращен в блок-схему.

После (рис 7) у вас будет возможность сохранить получившуюся блок-схему или загрузить другой файл с исходным кодом.

Для просмотра информации о программе нажмите на соответствующую кнопку (рис 9).

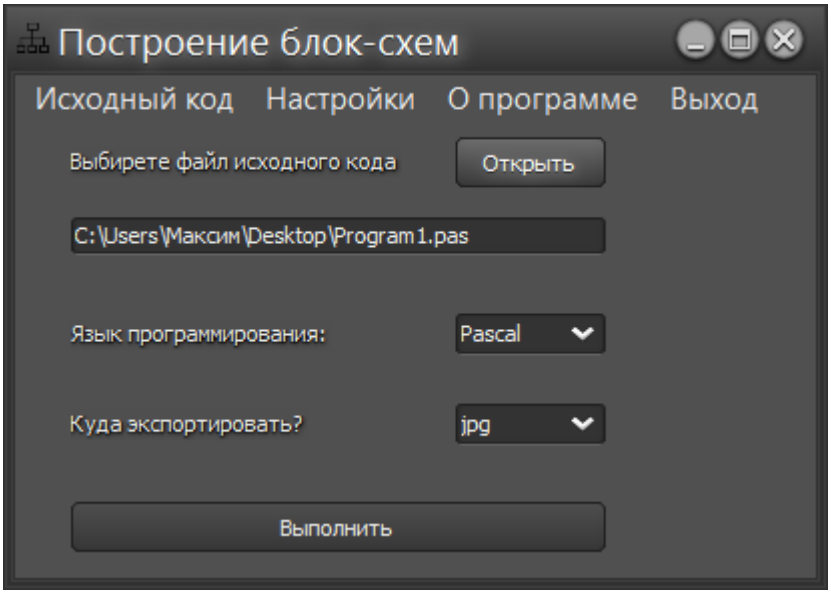

Рисунок 6. Главное окно программы.

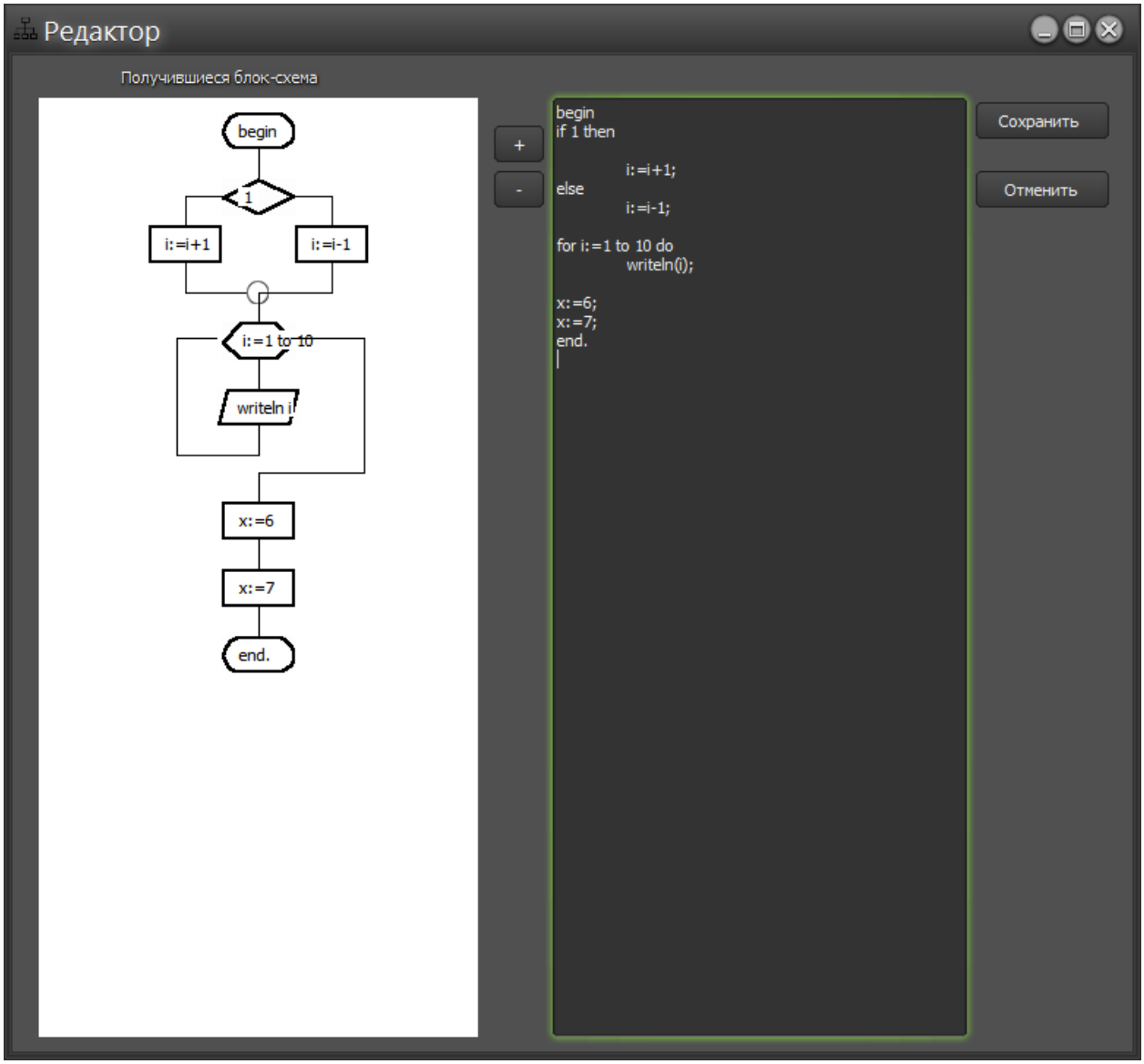

Рисунок 7. Окно просмотра.

Рисунок 8. Заставка.

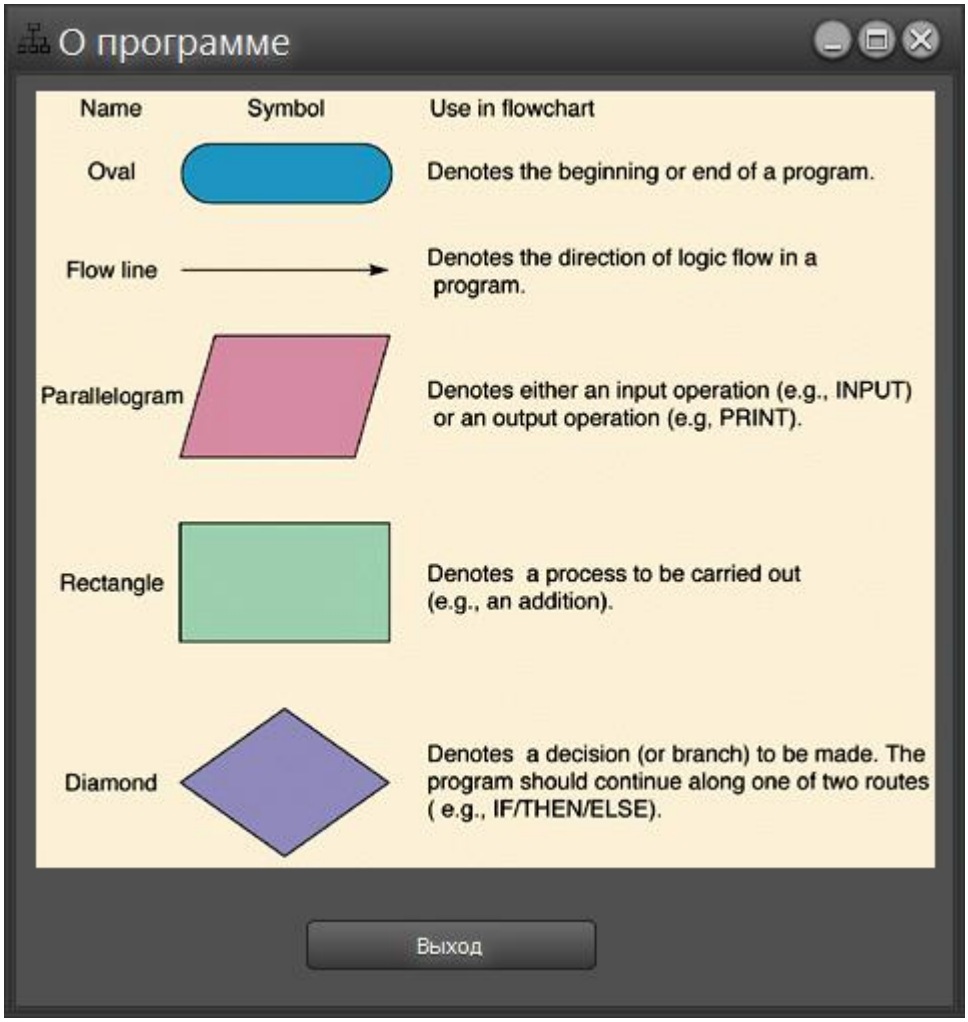

Рисунок 9. О программе.