МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

«МАТИ» - Российский государственный технологический университет
им. К.Э.Циолковского

Кафедра высшей математики

Создание приложения «Калькулятор» в среде программирования Delphi

Методические указания к лабораторной работе по курсу “Информатика”

Составитель: **В.М. Захарченко**

Москва - 2004

Начальные сведения о среде визуального программирования Delphi

Среда визуального программирования Дельфи (Delphi), производства фирмы Borland, в настоящее время держит первое место по популярности, как для профессиональных разработчиков сложных программных комплексов, так и для начинающих программистов, только осваивающих азы программирования. Этой популярности она обязана своим широким возможностям в сочетании с простотой и доступностью в изучении.

Дельфи в основе своей имеет язык Паскаль и его объектно-ориентированные версии. Среда Дельфи постоянно развивается и сейчас (2004 год) появилась уже восьмая версия системы. Развитие системы идет в основном за счет расширения возможностей работы с базами данных, построения сетевых приложений и пр. Поэтому для начинающего программиста в принципе не имеет большой разницы: работать с третьей или восьмой версией. Для выполнения работы может быть использована любая версия.

В результате выполнения лабораторной работы студент должен прочувствовать доступность современных программных средств и возможность самостоятельной разработки нормальных Windows приложений, будь то игрушка для себя, или компьютерное моделирование технической идеи или математической задачи.

Оконный интерфейс Delphi.

Интегрированная среда разработки Delphi представляет собой многооконную систему. После загрузки интерфейс Delphi включает пять основных окон для 6-й версии или 4 для 5-й и ниже.

- главное окно (**Delphi 6 – Project1);**
- окно обозревателя дерева объектов **(Object Tree View);** (может отсутствовать)
- окно инспектора объектов **(Object Inspector);**
- окно конструктора формы (**Form1);**
- окно редактора кода **Unit1.pas;**
- окно Проводника кода **(Exploring Unit1.pas)**

Вспомогательные окна возникают при вызове соответствующих средств, например **Image Editor** – редактор изображений.

Delphy – может одновременно работать только с одним приложением (проектом приложения).

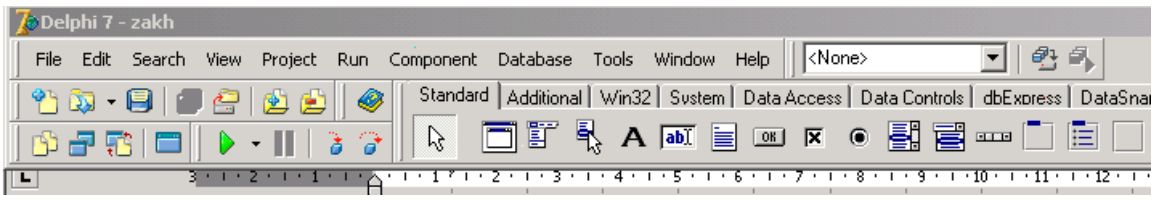
Главное окно Delphi включает:

- главное меню;
- панели инструментов;
- палитру компонентов;

Главное меню содержит набор команд для доступа к функциям Delphi.
Панели инструментов находятся под главным меню слева и содержит 15 кнопок для вызова часто используемых команд меню **File | Open, Run**): Возможен вызов комбинацией клавиш. Команды делятся на 6 групп

- Standart (стандартная)
- View (Просмотра)
- Debug (Отладка)
- Custom (Пользователя)
- Desktop (Рабочий стол)
- Internet (Интернет)

Панель инструментов настраивается щелчком правой кнопки мыши.



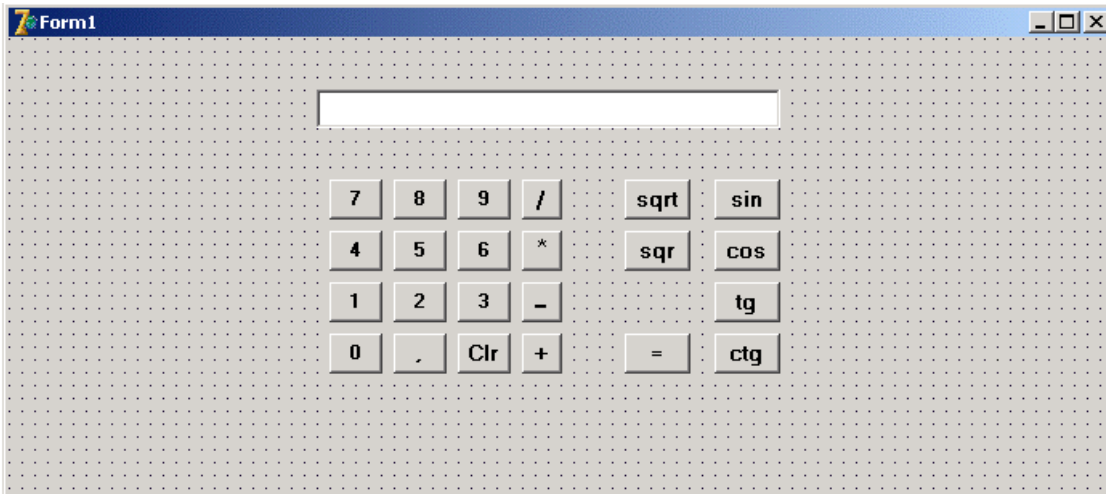
Палитра компонентов находится под главным меню справа и содержит компоненты для форм (интерфейса программы).

Компоненты - строительные блоки программы. Все компоненты разбиты на группы:

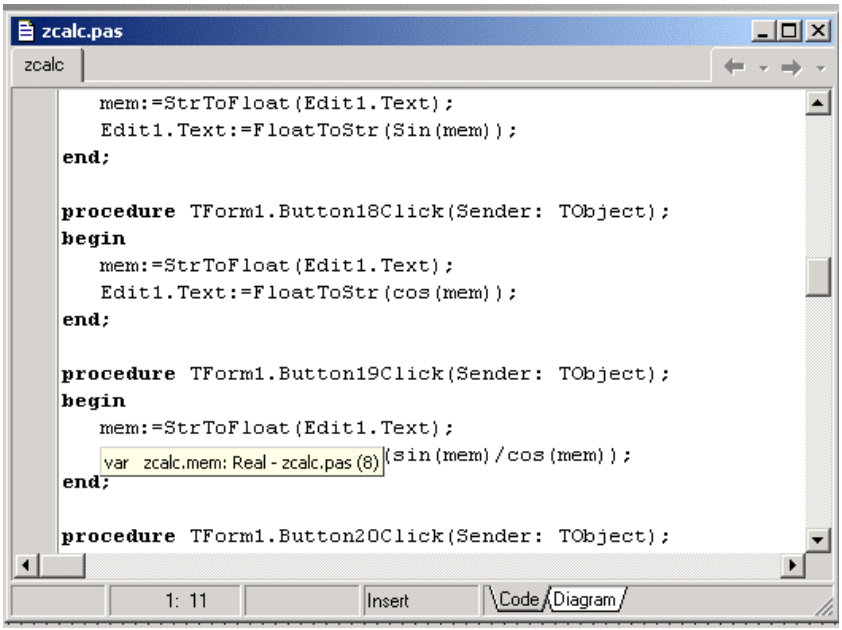
- **Standart** (Стандартная);
- **Additional** (дополнительная);
- **Win32** (32-разрядный интерфейс Windows);

- **System** (доступ к системным функциям);
- **Data Access** (Работа с базами данных);
- **Data Controls** (Управление данными);
- **BDE** (Доступ к данным с BDE (Borland Data Engine));
- **Qreport** (Составление отчетов);
- **Dialogs** (Создание стандартных диалоговых окон);

Окно **Конструктора форм** (первичный заголовок Form1) служит для переноса в это окно отдельных компонентов из палитры компонентов. Например, щелчок на компоненте Button, затем в окне формы. Затем двигаем кнопку, изменяем размеры, форму, надпись и пр.



Окно **Редактора кодов** (первичный заголовок Unit1.pas) служит для редактирования текста программных модулей и других текстовых файлов.



Окно **Проводника кода (Exploring Unit1.pas)** пристыковано слева к окну редактора кода. Служит для просмотра переменных, процедур, функций.
Окно **Инспектора объектов (Object Inspector)** находится в левой части экрана и служит для отображения и установки свойств объектов текущей формы.

Окно имеет две страницы:

Страница **Properties** отображает информацию о выбранном компонете окна формы и позволяет изменять эти свойства. Пример- Button1.

Страница **Events** определяет процедуру, которую компонент должен выполнить при возникновении указанного события. При работе программы, при возникновении этого события автоматически выполняется указанная процедура. Например: на форму перенесли кнопку с названием **Button1.**
В окне инспектора на странице свойств отображаются название, цвет, форма, надпись на кнопке и пр. На странице Events для этой кнопки обозначено событие OnClick т.е. один щелчок. Ему сопоставлена процедура программы (procedure Button1Click), которая вычисляет значение синуса при щелчке мыши на этой кнопке.

Характеристики проекта.

Состав проекта

В терминологии Delphi проект – это набор файлов, из которых компилятор создает исполняемые программы .exe.

.DPR. -Главный файл проекта. Его имя совпадает с именем всего проекта. Содержит общее описание проекта и основную программу (та что кончается **end.**) В ней запуск формы и всего с ней связанного. Она очень короткая.

.DOF – файл параметров проекта. Имя файла совпадает с названием проекта.

.RES - описание ресурсов. Имя файла совпадает с названием проекта. Может содержать пиктограммы, растровые изображения, курсоры.

.CFG – файл конфигурации проекта. Имя файла совпадает с названием проекта.

Файл каждой формы и связанного с ним программного файла носят собственное имя. Таких пар может быть несколько.

.DFM – описание формы. Содержит автоматически создаваемую программу вывода и исполнения формы.

.PAS - файл программного модуля, содержит текст программы, связанной с формой.

.PAS - файл программного модуля, не связанного с конкретной формой. В таком модуле размещают процедуры и функции, общие для нескольких модулей с формами.

.DCU – файл с откомпилированным кодом модуля: Имеет то же имя что и dfm и pas.

Компиляция и запуск проекта.

Большинство файлов проекта создается автоматически. Программист создает только файл формы (визуальное проектирование) и файл программного модуля (текст программы).

Запуск процесса компиляции происходит по команде **Project | Compile <Project1>** Имя проекта присваивает сам программист. В окне компиляции выводятся ошибки, предупреждения и подсказки. При компиляции сначала компилируются файлы всех модулей, затем компиляция файла проекта и создание исполняемого приложения с именем проекта и расширением **exe.**

Запустить приложение можно из Дельфи или из Windows как любую другую программу. Из Дельфи запуск проекта осуществляется командой **Run | Run** (Выполнение) или клавишей **<F9>.**

Продолжить разработку проекта можно только после завершения работы приложения.

При зависании приложения его надо завершить с помощью команды **Run | Program Reset**. (Выполнение | Остановить программу)

Создание проекта и интерфейса приложения «Калькулятор»

Начало работы . Создание приложения.

Перед началом работы создадим каталог для сохранения своего проекта. Имя должно включать номер группы и фамилию.

Запускаем Дельфи. На экране 4 окна. Автоматически уже созданы все основные файлы проекта, даже если мы ничего не вносили.

Сохраним проект под своим именем. Для этого меню **File -** команда **Save Project As** (Сохранить проект). В верхней части окна установить каталог, который вы создали для своего проекта.

Если проект сохраняется впервые, то сначала появляется окно **Save Unit1 as** (Сохранить программный модуль как). Надо присвоить программному модулю имя, например: Calculator1. Затем сохранить.

После этого выводится окно на переименование всего проекта **Save Project As** (Сохранить проект). Вводим имя проекта, например: **PetrovProject**. Далее сохранить.

После сохранения, создается директория с именем проекта **PetrovProject**, в ней содержатся основные файлы проекта: **PetrovPrj.dpr, PetrovPrj.dof,** и файлов формы и модуля: **Calculator1.dfm, Calculator1.pas.**

Работа с формой. Создание интерфейса приложения.

Создание эскиза интерфейса За основу можно взять стандартный калькулятор Windows).

Составляем список элементов интерфейса приложения:

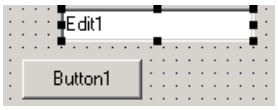
- типы элементов (кнопки, окна ввода-вывода, надписи)
- количество элементов каждого типа;
- размер и форма элементов;
- оформление, надписи, цвет;
- взаимное расположение элементов;

Оформление окна ввода-вывода

Вставка окна редактора.

Для ввода и вывода данных используем компонент Edit – готовый однострочный редактор.
находим среди палитры компонентов окошечко ab (Edit)

щелкаем на нем мышкой
ставим курсор мышки на то место формы, где должна быть окно для ввода текста
щелкаем мышкой - вставляем окно редактора
передвигаем окно редактора
изменяем размер окна редактора



Определение свойств окна ввода-вывода в инспекторе объектов

Выделяем в окне формы окно редактора Edit1 щелчком мыши. При этом в окне Инспекторе объектов появляется имя этого компонента, например Edit1.

Переходим в инспектор объектов для определения свойств кнопки. Должна быть выделена закладка Properties (Свойства).

находим строку **Text** | **Edit1**.
вместо Edit1 оставляем пустое место.
находим строку +**Font** | **(Tfont)** (Шрифт) и дважды щелкаем на TFont
в открывшемся окне устанавливаем параметры шрифта: тип шрифта, размер, цвет.

В окно редактора должны вводиться только цифры с клавишей калькулятора. Ввод других символов с клавиатуры компьютера приведет к сбою программы. Поэтому ввод с клавиатуры компьютера должен быть заблокирован. Для этого в окне инспектора объектов находим строку **ReadOnly | False** и меняем значение **False** на **True.**

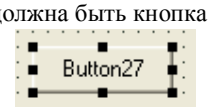
Оформление клавиатуры калькулятора

Для определения событий типа нажатия клавиши, используются объекты Button (кнопка)

Вставка кнопок калькулятора.

находим в палитре компонентов в линейке **Standart** кнопку OK (Button)

щелкаем на ней мышкой
затем ставим курсор мышки на то место формы, где должна быть кнопка
щелкаем мышкой - вставляем кнопку
передвигаем кнопку
изменяем размер кнопки



Определение свойств кнопок в инспекторе объектов

Выделяем в окне формы кнопку, например Button1 щелчком мыши. При этом в окне Инспекторе объектов появляется имя этого компонента Button1 .

Переходим (щелкаем) в инспектор объектов Закладка Properties

находим строку Caption и редактируем строку с надписью Button1

вместо Button1 пишем «1»

находим строку Font (Шрифт) и дважды щелкаем на TFont

в открывшемся окне устанавливаем параметры шрифта.

Переходим в инспектор объектов (Закладка **Events**).

В строке Action, действию **OnClick** должно соответствовать значение **Button1Click,** т.е. кнопке **Button1** соответствует одиночный щелчок мыши.

Оформление надписей в окне приложения.

Для вывода на экран произвольных надписей (статических, или динамических, меняющихся в ходе работы программы используется компонент Label группа **Standart** (вывод нередактируемых надписей.

Вставка вспомогательных надписей .

Находим среди компонентов значок надписи : (Label)

щелкаем на нем мышкой

затем перемещаем курсор мышки на то место формы, где должна быть надпись

щелкаем мышкой - вставляем надпись с первичным текстом

передвигаем надпись на нужное место

Определение свойств надписи в инспекторе объектов

Выделяем в окне формы надпись, например Label1 щелчком мыши. При этом в окне Инспекторе объектов появляется имя этого компонента Label1 | TLabel.

Переходим (щелкаем) в инспектор объектов Закладка Properties

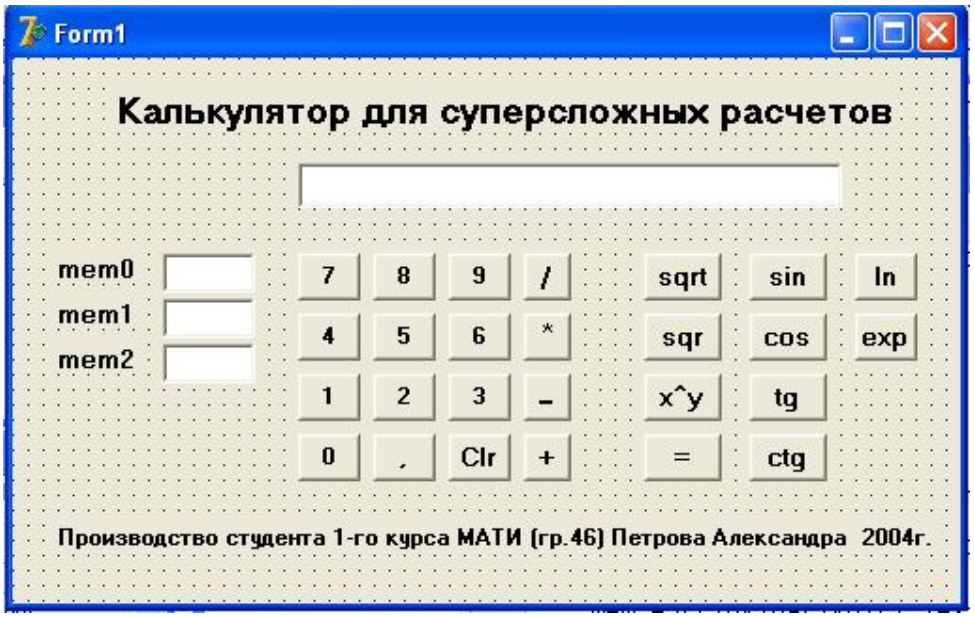
находим строку Caption | **Label1** и печатаем вместо Label1 нужный текст, например: «Калькулятор Для научных расчетов»

находим строку Font (Шрифт) и дважды щелкаем на TFont

в открывшемся окне устанавливаем параметры шрифта: тип, размер, жирность.

Группи ровка элементов интефейса.

Перемещаем элементы с помощью мыши, в соответствии с планом их взаимного расположения. Элементы можно перемещать по-отдельности или сразу группами, выделив мышкой прямоугольник с группой элементов.



Примерный вид окна формы с интерфейсом приложения «Калькулятор».

Написание кода программы

Основная программа генерируется автоматически и выводится в окне редактора кода с первичным именем Unit1.pas.

Начальный текст программы:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs;

type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
end.
```

Для работы программы калькулятора требуется несколько переменных типа **real** для ввода исходных данных, хранения результатов промежуточных вычислений, вывода результатов расчетов, и одна переменная типа char для хранения символа типа текущей математической операции.

Объявления переменных вставляем в программу вручную. После раздела **uses** и имен подключаемых модулей пишем

```
var x,y,z,x,y: real;
op:char;
```

Вся остальная программа состоит из набора процедур и функций, выполняющих основные действия программы.

Начальная заготовка процедуры (функции) и ее объявление в списке процедур, осуществляется автоматически. Для этого следует перейти в окно формы и дважды щелкнуть мышью на кнопке, с которой связано выполнение процедуры.

Просле этого в разделе **type п**оявляются строки

```
Button1: TButton;
procedure Button1Click(Sender: TObject);
```

а фокус редактора перемещается в тело созданной процедуры, которая будет выполеяться при нажатии клавиши Button1.

```
procedure TForm1.Button1Click(Sender: TObject);
begin

end;
```

Код программы - обработчика события, размещаемую между **begin** и **end;** программист должен написать самостоятельно.

Стандартные функции и процедуры, используемые в программе

Передать текст из окна редактора в строковую переменную (st:string)

```
st:=Edit1.Text;
```

Ввести в окно редактора текст из символьной переменной:

```
Edit1.Text:=st;
```

Ввод строки символов в окно редактора:

```
Edit1.Text:=’Введите число’;
```

Добавить к тексту в окне редактора символ или группу символов

```
Edit1.Text:=Edit1.Text+'1';
```

Преобразовать цифры в окне редактора в число и передать его в числовую переменную:
X:=StrToFloat(Edit1.Text);

Преобразовать значение числовой переменной в цифры и вывести в окне редактора:
Edit1.Text:=FloatToStr(X);

В качестве аргумента функции FloatToStr( ) может быть целое математическое выражение:

```
Edit1.Text:=FloatToStr(X+Y);
Edit1.Text:=FloatToStr(X-Y);
Edit1.Text:=FloatToStr(X*Y);
Edit1.Text:=FloatToStr(X/Y);
Edit1.Text:=FloatToStr(ln(X));
Edit1.Text:=FloatToStr(sin(X));
Edit1.Text:=FloatToStr(exp(X));
```

Для вывода строки на экран может использоваться компонент Label
Label.Caption:=’Результат расчета’;
Label.Caption:= FloatToStr(X);
Label2.Caption:=Edit1.Text
Два текста можно сложить в одну строку:
Label.Caption:= FloatToStr(X)+Edit1.Text;

Пример простой процедуры, осуществляющей пересчет фунтов в килограммы (может быть использована в калькуляторе):

```
Procedure TForm1.Button1.Click(Sender: Tobject);
begin
x:=StrToFloat(Edit1.Text);                                  {число фунтов}
y:=f*0.4059;                                                {перевод в килограммы}
Label2.Caption:=Edit1.Text + ‘фунтав  это ‘+FloatToStrF(y) + ‘килограммов’;
End;                                                        {10 фунтов это 4.059 килограммов}
```

Текст программы calc.pas

```
 unit zcalc;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;
  var  z,x,y: real;                                  {переменные для числовых данных}
     op:char;                                        {флажок типа операции}
type                                  {Этот раздел формируется автоматически}
  TForm1 = class(TForm)               {В нем содержится перечень всех объектов, введенных }
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Button5: TButton;
    Button6: TButton;
    Button7: TButton;
    Button8: TButton;
    Button9: TButton;
    Button10: TButton;
    Edit1: TEdit;
    Button11: TButton;
    Button12: TButton;
    Button13: TButton;
    Button14: TButton;
    Button15: TButton;
    Button16: TButton;
    Button17: TButton;
    Button18: TButton;
```

```
    Button19: TButton;
    Button20: TButton;
    Button21: TButton;
    Button22: TButton;
    Button23: TButton;
    Edit2: TEdit;
    Edit3: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Edit4: TEdit;
    Label3: TLabel;
    Button24: TButton;
    Button25: TButton;
    Button26: TButton;
    procedure Button16Click(Sender: TObject);
    procedure Button15Click(Sender: TObject);
    procedure Button17Click(Sender: TObject);
    procedure Button18Click(Sender: TObject);
    procedure Button19Click(Sender: TObject);
    procedure Button20Click(Sender: TObject);
    procedure Button21Click(Sender: TObject);
    procedure Button10Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button6Click(Sender: TObject);
    procedure Button7Click(Sender: TObject);
    procedure Button8Click(Sender: TObject);
    procedure Button9Click(Sender: TObject);
    procedure Button22Click(Sender: TObject);
    procedure Button12Click(Sender: TObject);
    procedure Button23Click(Sender: TObject);
    procedure Button13Click(Sender: TObject);
    procedure Button11Click(Sender: TObject);
    procedure Button14Click(Sender: TObject);
    procedure Button24Click(Sender: TObject);
    procedure Button25Click(Sender: TObject);
    procedure Button26Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

{ Группа процедур по вводу цифр в окно редактора }

procedure TForm1.Button10Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'0';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'1';                                    {}
```

```
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'2';                                    {}
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'3';                                    {}
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'4';                                    {}
end;

procedure TForm1.Button5Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'5';                                    {}
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'6';                                    {}
end;

procedure TForm1.Button7Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'7';                                    {}
end;

procedure TForm1.Button8Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'8';                                    {}
end;

procedure TForm1.Button9Click(Sender: TObject);
begin
Edit1.Text:=Edit1.Text+'9';                                    {}
end;

procedure TForm1.Button22Click(Sender: TObject);
begin                                          {разделитель может быть точкой или}
Edit1.Text:=Edit1.Text+',';                    {запятой, в зависимости от установок}
end;                                           {в Windows}
```

{ Процедура очистки окон основного и вспомогательых редакторов }

{Окна Edit2, Edit3, Edit4 введены для контроля текущих значений переменных в процессе отладки программы}

```
procedure TForm1.Button21Click(Sender: TObject);          {Clr}
begin
      Edit1.Text:='';                                             {}
      Edit2.Text:='';                                             {}
      Edit3.Text:='';                                             {}
      Edit4.Text:='';                                             {}
      z:=0;                                                       {}
      x:=0;                                                       {}
      y:=0;                                                       {}
end;
```

{ Группа процедур вычисления функции от одного аргумента }

```
procedure TForm1.Button16Click(Sender: TObject);        {Sqrt}
begin
  z:=StrToFloat(Edit1.Text);                            {Счит-е цифр и  преобразование в число}
  Edit1.Text:=FloatToStr(Sqrt(z));                      {выполнение операции и вывод рез-та}
end;

procedure TForm1.Button15Click(Sender: TObject);        {Sqr}
begin
  z:=StrToFloat(Edit1.Text);
  Edit1.Text:=FloatToStr(Sqr(z));
end;

procedure TForm1.Button17Click(Sender: TObject);        {Sin}
begin
  z:=StrToFloat(Edit1.Text);
  Edit1.Text:=FloatToStr(Sin(z));
end;

procedure TForm1.Button18Click(Sender: TObject);        {Cos}
begin
  z:=StrToFloat(Edit1.Text);
  Edit1.Text:=FloatToStr(cos(z));
end;

procedure TForm1.Button19Click(Sender: TObject);        {tg}
begin
  z:=StrToFloat(Edit1.Text);
  Edit1.Text:=FloatToStr(sin(z)/cos(z));
end;

procedure TForm1.Button20Click(Sender: TObject);        {Ctg}
begin
  z:=StrToFloat(Edit1.Text);
  Edit1.Text:=FloatToStr(cos(z)/sin(z));
end;

procedure TForm1.Button24Click(Sender: TObject);        {ln}
begin
  z:=StrToFloat(Edit1.Text);
Edit1.Text:=FloatToStr(ln(z));                          {Вывод результата}
end;

procedure TForm1.Button26Click(Sender: TObject);        {exp}
begin
  z:=StrToFloat(Edit1.Text);                            {Считываем цифры в переменную}
  Edit1.Text:=FloatToStr(exp(z));                       {Вычисляем exp и выводим в Edit1}
end;                                                    {Конец процедуры}
```

{ Группа процедур ввода первого аргумента в вычислениях с двумя аргументами }

```
procedure TForm1.Button11Click(Sender: TObject);        {*}{Операция умножения}
begin
   z:=StrToFloat(Edit1.Text);                           {Считываем число}
   x:=z;                                                {Запоминаем первый аргумент}
   Edit1.Text:='';                                      {Очистка окна редактора}
   op:='*';                                             {Ставим флажок операции}
Edit4.Text:=FloatToStr(z);                              {Контрольный вывод}
Edit2.Text:=FloatToStr(x);
Edit3.Text:=FloatToStr(y);
end;
```

```
procedure TForm1.Button13Click(Sender: TObject);          {-}{Операция вычитания}
begin
   z:=StrToFloat(Edit1.Text);                             {Считываем число}
   x:=z;                                                  {Запоминаем первый аргумент}
   Edit1.Text:='';                                        {Очищаем окно редактора}
   op:='-';                                               {Ставим флажок операции}
Edit4.Text:=FloatToStr(z);                                { Контрольный вывод  z}
Edit2.Text:=FloatToStr(x);                                { Контрольный вывод  x}
Edit3.Text:=FloatToStr(y);                                { Контрольный вывод y}
end;

procedure TForm1.Button12Click(Sender: TObject);          {+}
begin
   z:=StrToFloat(Edit1.Text);
   x:=z;
   Edit1.Text:='';
   op:='+';
Edit4.Text:=FloatToStr(z);
Edit2.Text:=FloatToStr(x);
Edit3.Text:=FloatToStr(y);
end;

procedure TForm1.Button14Click(Sender: TObject);          {/ - деление}
begin
   z:=StrToFloat(Edit1.Text);
   x:=z;
   Edit1.Text:='';
   op:='/';
Edit4.Text:=FloatToStr(z);                                {Контрольный вывод}
Edit2.Text:=FloatToStr(x);
Edit3.Text:=FloatToStr(y);
end;

procedure TForm1.Button25Click(Sender: TObject);          {x^y – возведение в степень}
begin
   z:=StrToFloat(Edit1.Text);
   x:=z;
   Edit1.Text:='';
   op:='^';
Edit4.Text:=FloatToStr(z);                                {Контрольный вывод}
Edit2.Text:=FloatToStr(x);
Edit3.Text:=FloatToStr(y);
end;
```

{ Процедура ввода второго аргумента и выполнения нужной математической функции при выполнении расчетов с двумя аргументами }

```
procedure TForm1.Button23Click(Sender: TObject);          {=}{обработка клавиши «=»}
begin
 if op='+' then                                           {проверяем флажок операции, если да,}
   begin                                                  {то начало составного оператора}
     y:=StrToFloat(Edit1.Text);                           {из строки в окне  в число и в y}
     z:=x+y;                                              {складываем числа }
     Edit1.Text:=FloatToStr(z);                           {число в строку цифр и в окно редактора}
   end;                                                   {конец оператора}
 if op='-' then                                           {проверка следующего условия}
   begin                                                  { начало составного оператора }
     y:=StrToFloat(Edit1.Text);                           {если условие не выполнено,}
     z:=x-y;                                              {то оператор игнорируется}
     Edit1.Text:=FloatToStr(z);
   end;
 if op='*' then                                           {флажок операции умножения}
```

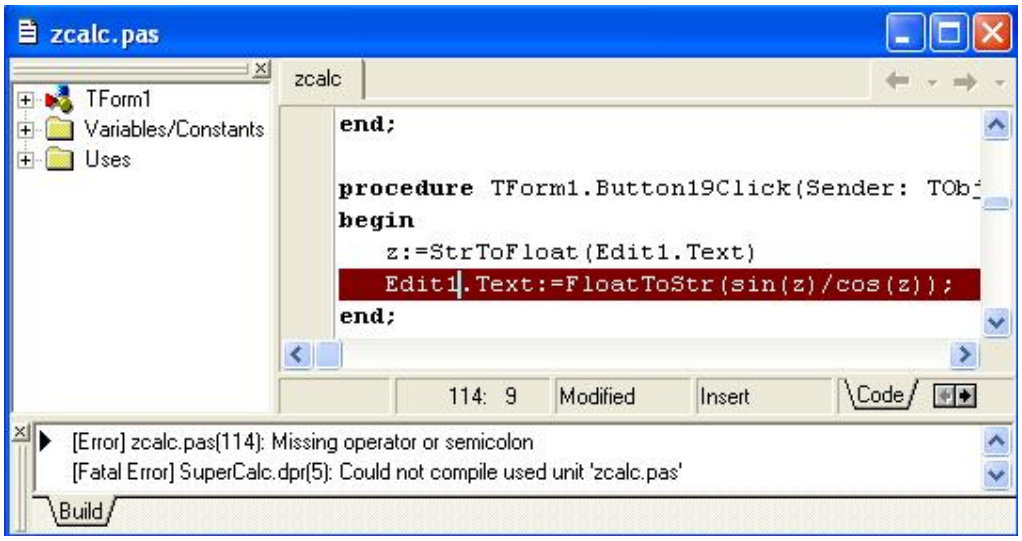
```
    begin
      y:=StrToFloat(Edit1.Text);
      z:=x*y;
      Edit1.Text:=FloatToStr(z);
    end;
  if op='/' then                                  { флажок операции деления}
    begin
      y:=StrToFloat(Edit1.Text);
      z:=x/y;
      Edit1.Text:=FloatToStr(z);
    end;
  if op='^' then                                  {операция возведения в степень}
    begin                                         {}
      y:=StrToFloat(Edit1.Text);                  {в Паскале нет операции возведения}
      z:=y*ln(x);                 { = Y*lnX }     {в степень. Поэтому используем опер.}
      z:=exp(z);                  { EXP(Y*lnX)=X^Y }  {LN  и  EXP}
      Edit1.Text:=FloatToStr(z);                  {Вывод результата в огно Edit1}
    end;
 Edit4.Text:=FloatToStr(z);                       { Контрольный вывод содержания z}
 Edit2.Text:=FloatToStr(x);                       { Контрольный вывод содержания x}
 Edit3.Text:=FloatToStr(y);                       { Контрольный вывод содержания y}
 end;
 end.                                             {Конец всей программы}
```

После того как написана программа или ее часть, выполняются функции компиляции и запуска программы. Окно программы при запуске выглядит примерно так:

Компиляция и запуск программы

Запуск процесса компиляции происходит по команде **Project | Compile <Project1>**

Если в программе найдена ошибка, то в окне программ, в нижней части, выводится список ошибок с номерами строк. Первая строка с ошибкой выделяется и в неё перемещается курсор.



Запуск программы осуществляется командой **Run | Run** (Выполнение) или клавишей **<F9>.**

Продолжить разработку проекта можно только после завершения работы приложения.

При зависании приложения его надо завершить с помощью команды **Run | Program Reset**. (Выполнение | Остановить программу)

Запустить приложение **SuperCalc.exe** можно из Дельфи или из Windows, как любую другую программу.

Примерный вид запущенного приложения: