

Никита Культин

Delphi

в задачах и примерах

3-е издание

Санкт-Петербург
«БХВ-Петербург»

2012

УДК 681.3.068+800.92Delphi
ББК 32.973.26-018.1
К90

Культин Н. Б.

К90 Delphi в задачах и примерах. — 3-е изд., перераб.
и доп. — СПб.: БХВ-Петербург, 2012. — 288 с.: ил.

ISBN 978-5-9775-0811-7

Книга представляет собой сборник примеров программ и задач для самостоятельного решения в среде программирования Delphi. Примеры различной степени сложности — от простейших до программ работы с графикой, звуком и базами данных — демонстрируют возможности среды разработки Delphi, назначение основных компонентов. Справочник содержит описание наиболее часто используемых компонентов и функций. В третьем издании обновлены старые и добавлены новые примеры. Проекты из книги размещены на сайте издательства.

Для начинающих программистов

УДК 681.3.068+800.92Delphi
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Екатерина Капалыгина</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Игоря Цырульниковца</i>
Оформление обложки	<i>Марины Дамбиевой</i>
Зав. производством	<i>Николай Тверских</i>

Подписано в печать 29.02.12.

Формат 60×90¹/₁₆. Печать офсетная. Усл. печ. л. 18.

Тираж 2000 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Отпечатано с готовых диапозитивов

в ГУП "Типография "Наука"

199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0811-7

© Культин Н. Б., 2012

© Оформление, издательство "БХВ-Петербург", 2012

Оглавление

Предисловие	5
ЧАСТЬ 1. Примеры и задачи	7
Базовые компоненты.....	8
Графика	63
Мультимедиа	116
Файлы.....	137
Игры и полезные программы	145
Базы данных	213
Печать счета	233
ЧАСТЬ 2. Справочник	239
Форма.....	240
Базовые компоненты.....	242
<i>Label</i>	242
<i>Edit</i>	243
<i>Button</i>	244
<i>Memo</i>	245
<i>RadioButton</i>	246
<i>CheckBox</i>	247
<i>ListBox</i>	248
<i>ComboBox</i>	249
<i>StringGrid</i>	251
<i>Image</i>	252
<i>Timer</i>	253
<i>SpeedButton</i>	254
<i>UpDown</i>	256
<i>OpenDialog</i>	257

<i>SaveDialog</i>	258
<i>Animate</i>	259
<i>MediaPlayer</i>	260
Компоненты доступа/манипулирования данными	261
<i>ADOConnection</i>	261
<i>ADOTable</i>	262
<i>ADODataset</i>	263
<i>ADOQuery</i>	264
<i>DataSource</i>	265
<i>DBEdit, DBMemo, DBText</i>	265
<i>DBGrid</i>	266
<i>DBNavigator</i>	268
Графика	269
<i>PaintBox</i>	269
<i>Canvas</i>	270
<i>Pen</i>	272
<i>Brush</i>	273
Цвет	274
Функции	275
Функции ввода и вывода	275
Математические функции	276
Функции преобразования	276
Функции манипулирования датами и временем	277
События	279
Исключения	280
Приложение. Описание электронного архива	283
Предметный указатель	285



Предисловие

В последнее время резко возрос интерес к программированию. Это связано с развитием и внедрением в повседневную жизнь информационно-коммуникационных технологий. Если человек имеет дело с компьютером, то рано или поздно у него возникает желание, а иногда и необходимость программировать.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую "быструю разработку". В основе идеологии систем быстрой разработки (RAD-систем, Rapid Application Development — среда быстрой разработки приложений) лежат технологии визуального проектирования и событийного объектно-ориентированного программирования, суть которых заключается в том, что среда разработки берет на себя большую часть рутинной работы, оставляя программисту работу по конструированию диалоговых окон и созданию функций обработки событий. Производительность программиста при использовании RAD-систем — фантастическая!

Среди RAD-систем особо выделяется среда Delphi, которая позволяет создавать различные программы: от простейших однооконных приложений до программ управления распределенными базами данных.

В качестве языка программирования в Delphi используется язык Delphi (Delphi language), являющийся прямым потомком языка

Pascal хорошо известного многим, в том числе начинающим программистам.

Чтобы научиться программировать, надо программировать — писать программы, решать практические, реальные задачи. Для этого необходимо изучить язык программирования и среду разработки. Освоить язык программирования Delphi не очень сложно. Труднее изучить среду программирования, точнее научиться использовать компоненты. И здесь хорошим подспорьем могут быть программы, демонстрирующие назначение компонентов и особенности работы с ними.

В книге, которую вы держите в руках, собраны разнообразные примеры, которые не только демонстрируют назначение компонентов и возможности среды разработки Delphi, но и знакомят с принципами работы с графикой, звуком, базами данных. Следует обратить внимание, что большинство примеров не являются учебными в чистом смысле, это вполне работоспособные программы.

Книга состоит из двух частей.

- *Часть 1* содержит примеры и задачи для самостоятельного решения. Примеры представлены в виде краткого описания, сформулированного в форме задания для самостоятельного решения, диалоговых окон и хорошо документированных текстов программ.
- *Часть 2* — это краткий справочник по компонентам и функциям Delphi. В нем можно найти описание базовых компонентов, компонентов доступа к базам данных, описание часто используемых функций.

Научиться программировать можно только программируя, решая реальные задачи. При этом достигнутые в программировании успехи в значительной степени зависят от опыта. Поэтому, чтобы получить максимальную пользу от книги, вы должны работать с ней активно. Изучайте листинги, старайтесь понять, как работают программы. Не бойтесь экспериментировать — вносите изменения в программы. Если что-то не понятно, обратитесь к справочнику (*часть 2*) или к справочной системе Delphi.



ЧАСТЬ

1



Примеры и задачи

БАЗОВЫЕ КОМПОНЕНТЫ

В этом разделе приведены простые примеры и задачи, основное назначение которых — научить работать с базовыми компонентами.

Общие замечания

- Процесс создания программы в Delphi состоит из двух шагов: сначала нужно создать форму программы (диалоговое окно), затем — написать процедуры обработки *событий*. Форма *приложения* (так принято называть прикладные программы, работающие в Windows) создается путем добавления на форму *компонентов* и последующей их настройки.
- На форме практически любого приложения есть компоненты, которые обеспечивают интерфейс (взаимодействие) между программой и пользователем. Такие компоненты называют *базовыми*. К базовым компонентам можно отнести:
 - Label — поле вывода текста;
 - Edit — поле ввода/редактирования текста;
 - Button — командную кнопку;
 - CheckBox — независимую кнопку выбора;
 - RadioButton — зависимую кнопку выбора;

- `ListBox` — список выбора;
 - `ComboBox` — комбинированный список выбора.
- Вид компонента, его размер и поведение определяются значениями *свойств* (характеристик) компонента (описание свойств базовых компонентов можно найти в справочнике, во второй части книги).
- Основную работу в программе выполняют процедуры обработки *событий* (описание основных событий можно найти в справочнике, во второй части книги).
- Исходную информацию программа может получить из полей ввода/редактирования (компонент `Edit`), списка выбора (компонент `ListBox`) или комбинированного списка (компонент `ComboBox`). Для ввода значений логического типа можно использовать компоненты `CheckBox` и `RadioButton`.
- Результат программа может вывести в поле вывода текста (компонент `Label`) или в окно сообщения (функция `MessageDlg`).
- Для преобразования текста, например находящегося в поле ввода/редактирования, в целое число нужно использовать функцию `StrToInt`, а в дробное — функцию `StrToFloat`. Для преобразования целого, например значения переменной, в строку нужно использовать функцию `IntToStr`, а для преобразования дробного — функцию `FloatToStr` или `FloatToStrF`.

1. Написать программу **Мили-километры**, которая пересчитывает расстояние из миль в километры (1 миля равна 1 км 609,34 м). Рекомендуемый вид формы приведен на рис. 1.1.

```
// щелчок на кнопке Пересчет
```

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

```
var
```

```
    mile: real; // расстояние в милях
```

```
    km: real; // расстояние в километрах
```

```
begin
```

```
    // ввести исходные данные
```

```
    mile := StrToFloat(Edit1.Text);
```

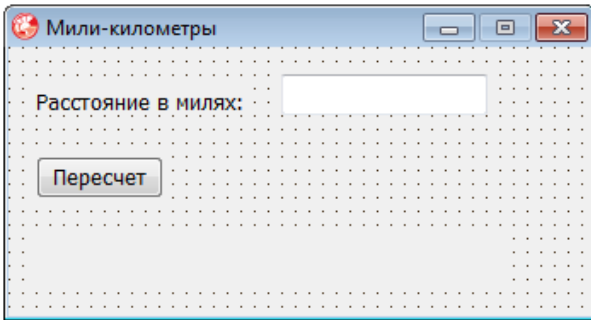


Рис. 1.1. Форма программы Мили-километры

```
// пересчитать
km := mile * 1.609344; // 1 МИЛЯ – 1,609344 км

// вывести результат
Label2.Caption := FloatToStr(mile) + ' МИЛЬ – это ' +
                  FloatToStr(km) + ' км.';

end;
```

2. Усовершенствуйте программу Мили-километры так, чтобы пользователь мог ввести в поле Расстояние в милях только число.

```
// нажатие клавиши в поле компонента Edit1
procedure TForm1.Edit1KeyPress(Sender: TObject;
                               var Key: Char);
begin
    // Процедура проверяет, является ли символ, соответствующий
    // нажатой клавише (Key), допустимым. Если символ неверный,
    // то он заменяется "нуль символом", который в поле
    // редактирования не отображается. В результате
    // у пользователя создается впечатление, что клавиатура
    // не реагирует на нажатие "неверных" клавиш.
    // В данном случае правильными являются
    // цифровые клавиши, запятая и <Backspace>.

    case Key of
        '0'..'9', #8: ; // цифра или <Backspace>
        ',': // запятая
```

```
        if Pos(',', Edit1.Text) <> 0 // запятая уже
                                   // введена
            then Key := #0;
        else Key := #0; // остальные символы не отображать
    end;
end;

// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
    mile: real; // расстояние в милях
    km: real;   // расстояние в километрах
begin
    // Если в поле Edit1 нет данных, то при выполнении
    // функции StrToFloat возникает исключение (ошибка).
    // Проверим, введены ли исходные данные.

    if Length(Edit1.Text) = 0 then
    begin
        ShowMessage('Надо ввести исходные данные');
        exit;
    end;

    // пользователь ввел расстояние в милях
    mile := StrToFloat(Edit1.Text);

    km := mile * 1.609344; // 1 миля - 1,609344 км

    Label2.Caption := FloatToStr(mile) + ' миль - это ' +
        FloatToStrF(km, ffFixed, 6, 2) + ' км.';
end;
```

3. Написать программу **Конвертор**, которая пересчитывает цену из долларов в рубли. Рекомендуемый вид формы приведен на рис. 1.2. Программа должна быть спроектирована таким образом, чтобы пользователь мог ввести в поля редактирования только правильные данные (дробные числа). При нажатии клавиши <Enter> в поле **Курс** курсор должен переходить в поле **Цена**, а

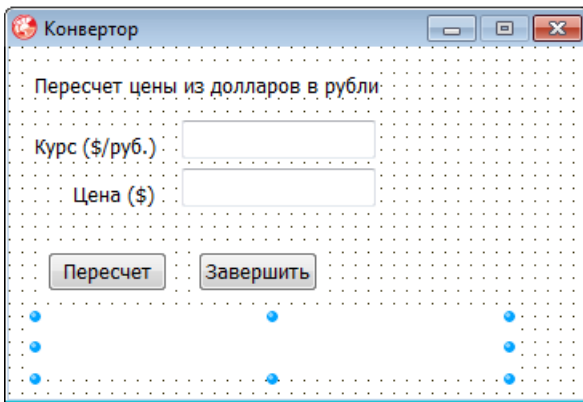


Рис. 1.2. Форма программы **Конвертор**

при нажатии этой же клавиши в поле **Цена** — на кнопку **Пересчет**.

```
// нажатие клавиши в поле Курс
procedure TForm1.Edit1KeyPress(Sender: TObject;
                                var Key: Char);
begin
    case Key of
        '0'..'9', #8: ; // цифры и <Backspace>

        '.', ',', ':
            // Обработку десятичного разделителя
            // сделаем "интеллектуальной". Заменяем точку
            // и запятую на символ
            // FormatSettings.DecimalSeparator – символ,
            // который при текущей настройке операционной
            // системы является десятичным разделителем.
            begin
                Key := FormatSettings.DecimalSeparator;
                // Проверим, введен ли уже в поле
                // Edit десятичный разделитель
                if pos(FormatSettings.DecimalSeparator,
                    Edit1.Text) <> 0
                    then Key := #0;
            end;
    end;
```

```
#13: Edit2.SetFocus; // Нажата клавиша <Enter>
           // Переместить курсор в поле Edit2
// остальные символы запрещены
else Key := #0;
end;
end;

// нажатие клавиши в поле Цена
procedure TForm1.Edit2KeyPress(Sender: TObject;
                               var Key: Char);
begin
  case Key of
    '0'..'9', #8: ; // цифры и <Backspace>
    '.', ',', ':
      begin
        Key := FormatSettings.DecimalSeparator;
        // проверим, введен ли уже в поле Edit
        // десятичный разделитель
        if
pos(FormatSettings.DecimalSeparator, Edit1.Text) <> 0
          then Key := #0;
        end;
        #13: Button1.SetFocus; // Сделать активной
                               // кнопку Пересчет

      else Key := Char(0); // остальные символы запрещены
    end;
  end;

// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
  usd: real; // цена в долларах
  k: real; // курс
  rub: real; // цена в рублях

begin
  k := StrToFloat(Edit1.Text);
  usd := StrToFloat(Edit2.Text);
```

```
// пересчитать цену из долларов в рубли
rub := usd * k;

// вывести результат расчета в поле Label4
Label4.Caption := FloatToStr(usd) + '$ = ' +
                  FloatToStrF(rub, ffCurrency, 6,2);
end;
```

4. Усовершенствуйте программу Конвертор так, чтобы событие `KeyPress` **обоих** полей редактирования обрабатывала одна процедура, а также чтобы кнопка **Пересчет** становилась доступной только после ввода данных в оба поля редактирования.

```
{
  Процедура EditKeyPress обрабатывает нажатие клавиш
  в полях Курс и Цена. Сначала надо обычным образом
  создать процедуру обработки события KeyPress для поля
  Edit1, назвав ее EditKeyPress. Затем надо выбрать компонент
  Edit2 и указать процедуру EditKeyPress в качестве процедуры
  обработки события Keypress.
  Чтобы узнать, на каком компоненте произошло
  событие, надо проверить значение свойства Sender.
}
procedure TForm1.EditKeyPress(Sender: TObject;
                               var Key: Char);
begin
  case Key of
    '0'..'9', #8: ; // цифры и <Backspace>

    '.',',',' ':
      // Обработку десятичного разделителя
      // сделаем "интеллектуальной". Заменяем точку
      // и запятую на символ
      // FormatSettings.DecimalSeparator – символ,
      // который при текущей настройке операционной
      // системы должен использоваться при записи
      // дробных чисел.
      begin
        Key := FormatSettings.DecimalSeparator;
```

```
        // проверим, введен ли уже в поле Edit
        // десятичный разделитель
        if
pos(FormatSettings.DecimalSeparator, Edit1.Text) <> 0
            then Key := #0;
        end;
#13: // клавиша <Enter>
        // параметр Sender содержит имя компонента,
        // на котором произошло событие
        if Sender = Edit1 then
            Edit2.SetFocus // Переместить курсор
                            // в поле Edit2
        else Button1.SetFocus; // Установить фокус
                            // на Button1

        else Key := #0; // остальные символы запрещены
    end;
end;

// EditChange - текст, находящийся в поле редактирования,
// изменился. Процедура EditChange обрабатывает изменение
// текста в полях Курс и Цена
procedure TForm1.EditChange(Sender: TObject);
begin
    // проверим, есть ли данные в полях редактирования
    if (Length(Edit1.Text) = 0) or (Length(Edit2.Text) = 0)
        then Button1.Enabled := False // кнопка Пересчет
                                        // недоступна
        else Button1.Enabled := True; // кнопка Пересчет
                                        // доступна
    end;

// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
    usd: real; // цена в долларах
    k: real; // курс
    rub: real; // цена в рублях
```

begin

```
k := StrToFloat(Edit1.Text);
usd := StrToFloat(Edit2.Text);
```

```
// пересчитать цену из долларов в рубли
rub := usd * k;
```

```
// вывести результат расчета в поле Label4
Label4.Caption := FloatToStr(usd) + '$ = ' +
                  FloatToStrF(rub, ffCurrency, 6, 2);
```

end;

5. Написать программу, которая пересчитывает вес из фунтов в килограммы (1 фунт = 0,4536 кг). Рекомендуемый вид формы приведен на рис. 1.3. Программа должна быть спроектирована таким образом, чтобы пользователь мог ввести в поле **Вес в фунтах** только положительное число (целое или дробное).

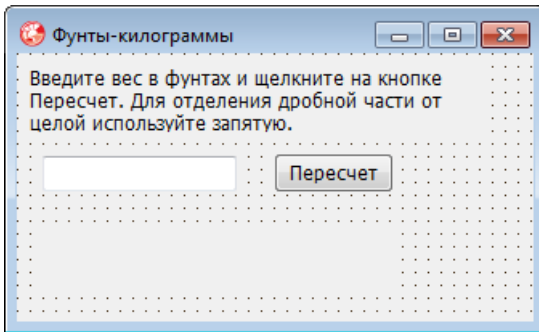


Рис. 1.3. Форма программы **Фунты-килограммы**

6. Написать программу, вычисляющую скорость (км/час), с которой бегун пробежал дистанцию. Рекомендуемый вид формы приведен на рис. 1.4. Программа должна быть спроектирована таким образом, чтобы в поля **Дистанция** и **Минут** можно было ввести только целое число, а в поле **Секунд** — дробное.


```

begin
  case Key of
    '0'..'9':
      #8      :
      #13     : Edit3.SetFocus; // <Enter> – курсор
                // в поле Секунд

    // остальные символы – запрещены
    else Key :=Chr(0); // символ не отображать
  end;
end;

// нажатие клавиши в поле Секунд
procedure TForm1.Edit3KeyPress(Sender: TObject;
                               var Key: Char);

begin
  case Key of
    '0'..'9': ;
    ',','.': // десятичный разделитель
      begin
        Key := FormatSettings.DecimalSeparator;
        if
          Pos(FormatSettings.DecimalSeparator,Edit3.Text) <> 0
        then Key := Char(0);
      end;
    #8: ; // <Backspace>
    #13: Button1.SetFocus; // установить фокус
          // на кнопку Вычислить

    // остальные символы – запрещены
    else Key :=Chr(0); // символ не отображать
  end;
end;

// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
var
  dist : integer; // дистанция, метров
  min  : integer; // время, минуты

```



```
FloatToStrF(sek, ffFixed,4,2) + ' сек' +
#13 + 'Скорость: ' +
FloatToStrF(v,ffFixed,4,2) + ' км/час';
```

```
end;
```

```
// щелчок на кнопке Завершить
```

```
procedure TForm1.Button2Click(Sender: TObject);
```

```
begin
```

```
Form1.Close; // закрыть главную форму – завершить
              // работу программы
```

```
end;
```

7. Написать программу, которая вычисляет силу тока в электрической цепи. Рекомендуемый вид формы приведен на рис. 1.5. Программа должна быть спроектирована таким образом, чтобы кнопка **Вычислить** была доступна только в том случае, если пользователь ввел величину напряжения и сопротивления.

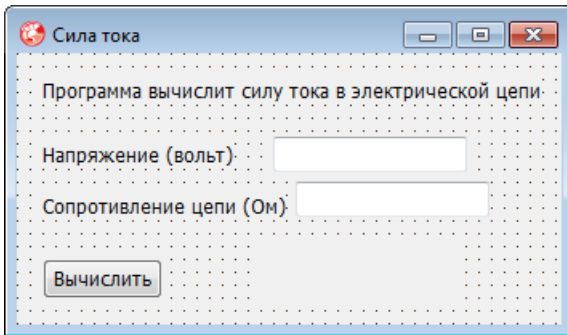
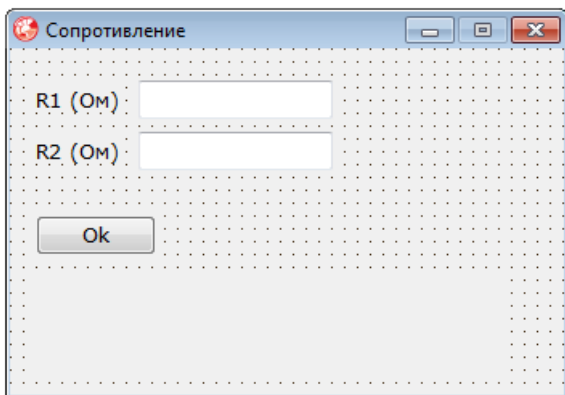
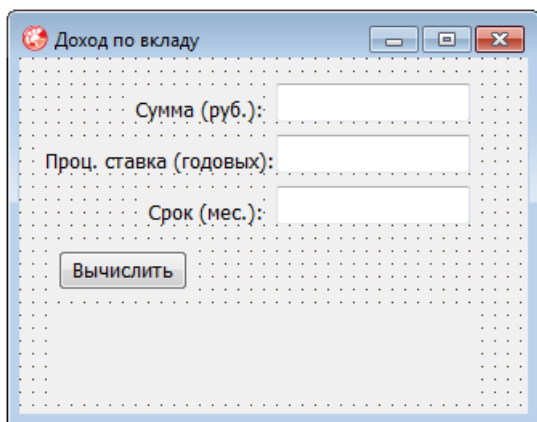


Рис. 1.5. Форма программы **Сила тока**

8. Написать программу, которая вычисляет сопротивление электрической цепи, состоящей из двух параллельно соединенных резисторов. Рекомендуемый вид формы приведен на рис. 1.6.

9. Написать программу, которая вычисляет доход по вкладу методом простых процентов ($\text{Доход} = \text{Сумма} * (\text{Процент} / 12) * \text{Срок}$). Рекомендуемый вид формы программы приведен на рис. 1.7. В результате щелчка на кнопке **Вычислить** в окне про-

Рис. 1.6. Форма программы **Сопrotивление**Рис. 1.7. Форма программы **Доход по вкладу**

граммы должна отображаться величина дохода и сумма в конце срока вклада. Программа должна быть спроектирована таким образом, чтобы в поля **Сумма** и **Проц. ставка** можно было ввести дробные числа, а в поле **Срок** — только целое.

10. Написать программу, которая вычисляет сопротивление электрической цепи, состоящей из двух сопротивлений. Сопротивления могут быть соединены последовательно или параллельно. Рекомендуемый вид формы приведен на рис. 1.8. Если величина

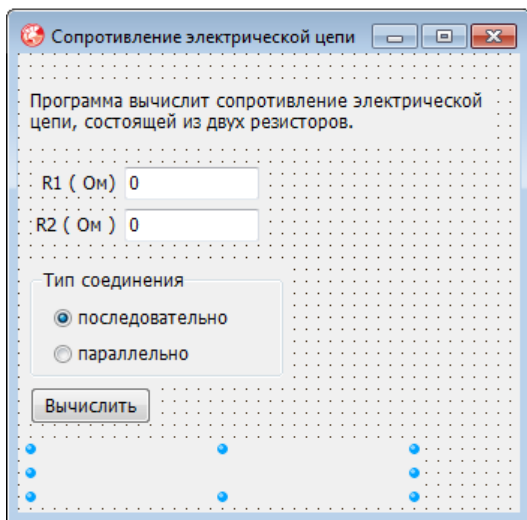


Рис. 1.8. Форма программы Сопrotивление электрической цепи

сопротивления цепи превышает 1 000 Ом, то результат должен быть выведен в килоомах.

```
// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
var
    r1,r2: real; // величины сопротивлений
    r: real;     // сопротивление цепи
begin
    // получить исходные данные
    r1 := StrToFloat(Edit1.Text);
    r2 := StrToFloat(Edit2.Text);

    if (r1 = 0) and (r2 = 0) then
    begin
        ShowMessage('Надо задать величину хотя бы одного
                    сопротивления');
        exit;
    end;

    // переключатели RadioButton1 и RadioButton2
    // зависимые, поэтому о типе соединения можно
    // судить по состоянию одного из них
```

```
if RadioButton1.Checked
  then // выбран переключатель Последовательно
    r:= r1+r2
  else // выбран переключатель Параллельно
    r:= (r1*r2)/(r1+r2);

Label4.Caption := 'Сопротивление цепи: ';
if r < 1000 then
  Label4.Caption := Label4.Caption +
    FloatToStrF(r, ffFixed, 3, 2) + ' Ом'
else
  begin
    r:=r/1000;
    Label4.Caption := Form1.Label4.Caption +
      FloatToStrF(r, ffFixed, 3, 2) + ' кОм';
  end
end;

// щелчок на переключателе Последовательно
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
  // пользователь изменил тип соединения
  Label4.Caption := '';
end;

// щелчок на переключателе Параллельно
procedure TForm1.RadioButton2Click(Sender: TObject);
begin
  // пользователь изменил тип соединения
  Label4.Caption := '';
end;
```

11. Напишите программу, которая вычисляет доход по вкладу. Программа должна обеспечивать расчет простых и сложных процентов. Простые проценты начисляются в конце срока вклада, сложные — ежемесячно и прибавляются к текущей (накопленной) сумме вклада и в следующем месяце проценты начисляются на новую сумму. Рекомендуемый вид формы программы приведен на рис. 1.9.

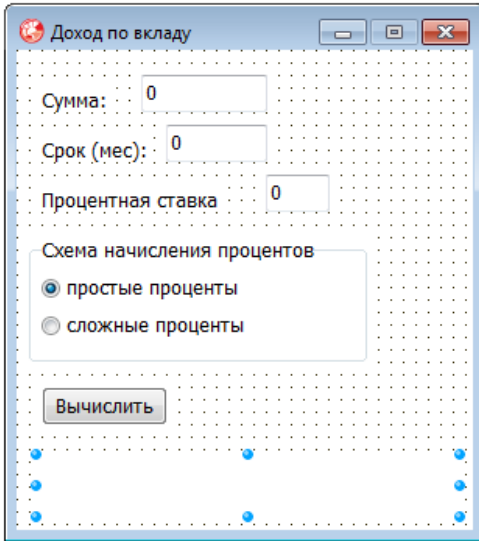


Рис. 1.9. Форма программы Доход

```
// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
var
    sum : real;           // сумма вклада
    pr : real;           // процентная ставка
    period: integer;     // срок вклада
    profit: real;        // доход по вкладу

    sum2: real;          // сумма, при вычислении методом
                        // сложных процентов

    i: integer;

begin
    // получить исходные данные
    sum := StrToFloat(Edit1.Text);
    pr := StrToFloat(Edit2.Text);
    period := StrToInt(Edit3.Text);

    if RadioButton1.Checked then
        // выбран переключатель Простые проценты
        profit := sum * (pr/100/12) * period
```



```
else
    // т. к. в группе два переключателя, то если
    // не выбран RadioButton1, то выбран
    // RadioButton2 – Сложные проценты
begin
    sum2:= sum;
    for i:=1 to period do
        sum2 := sum2 + sum2 * (pr/100/12);
    // здесь sum2 – сумма в конце срока вклада
    profit := sum2 – sum;
end;

sum := sum + profit;
Label4.Caption := 'Доход: ' +
    FloatToStrF(profit, ffCurrency, 6, 2) +
    #13 +
    'Сумма в конце срока вклада: ' +
    FloatToStrF(sum, ffCurrency, 6, 2);

end;
```

12. Написать программу, которая вычисляет стоимость жалюзи. Рекомендуемый вид формы приведен на рис. 1.10.

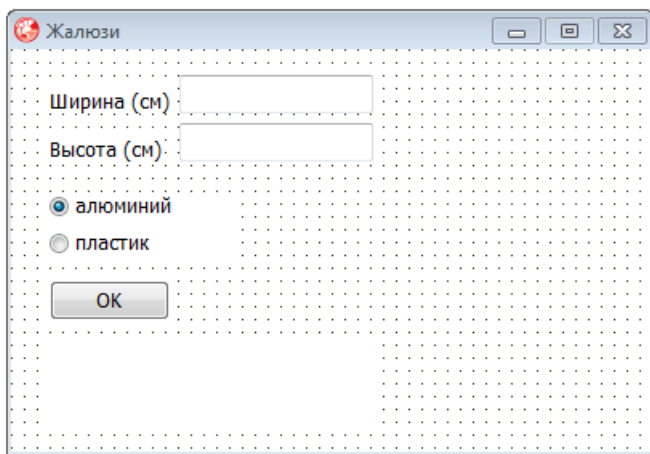


Рис. 1.10. Форма программы Жалюзи

13. Написать программу, которая позволяет пересчитать цену из долларов в рубли или из рублей в доллары. Рекомендуемый вид формы приведен на рис. 1.11. Во время работы программы, в результате выбора вида конвертации, соответствующим образом должен меняться заголовок окна и текст, поясняющий назначение полей ввода.

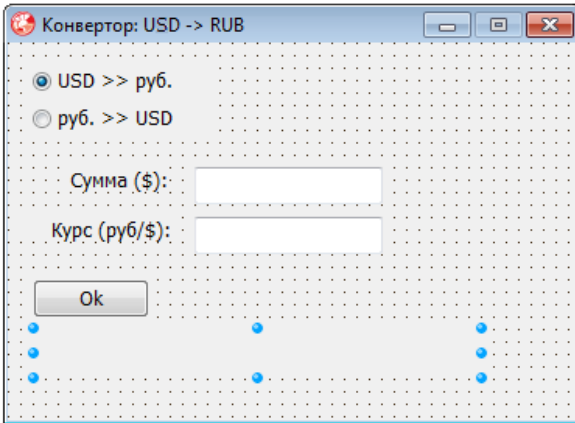


Рис. 1.11. Форма программы **Конвертор**

```
// щелчок на переключателе USD >> руб.
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
    // изменить заголовок окна
    Form1.Caption := 'Конвертор: USD -> RUB';

    // изменить текст перед полем Edit1
    Label1.Caption := 'Сумма ($) : ';

    // установить курсор в поле Сумма
    Edit1.SetFocus;

    Label3.Caption := '';
end;

// щелчок на переключателе руб. >> USD
procedure TForm1.RadioButton2Click(Sender: TObject);
```

begin

Form1.Caption := 'Конвертор: RUB -> USD';

Label1.Caption := 'Сумма (руб.): ';

Edit1.SetFocus;

Label3.Caption := '';

end;

// Щелчок на кнопке ОК

procedure TForm1.Button1Click(Sender: TObject);

var

usd: real; *// цена в долларах*

rub: real; *// цена в рублях*

k: real; *// курс*

begin

k := StrToFloat(Edit2.Text);

if RadioButton1.Checked **then**

begin

// пересчет из долларов в рубли

usd := StrToFloat(Edit1.Text);

rub := usd * k;

Label3.Caption := FloatToStrF(usd, ffFixed, 6, 2) +
'\$ = ' +
FloatToStrF(rub, ffCurrency, 6, 2);

end

else begin

// пересчет из рублей в доллары

rub := StrToFloat(Edit1.Text);

usd := rub / k;

Label3.Caption := FloatToStrF(rub, ffCurrency, 6, 2) +
' = ' +
FloatToStrF(usd, ffFixed, 6, 2) + '\$';

end;

end;

// процедура обрабатывает событие EditChange

// компонентов Edit1 и Edit2

```

procedure TForm1.EditChange(Sender: TObject);
begin
    // если в каком-либо из полей Edit нет данных,
    // сделать кнопку Button1 недоступной
    if (Length(Edit1.Text) = 0) or (Length(Edit2.Text) = 0)
        then Button1.Enabled := False
        else Button1.Enabled := True;

    Label3.Caption := '';
end;

// нажатие клавиши в поле Сумма
procedure TForm1.Edit1KeyPress(Sender: TObject;
                                var Key: Char);
begin
    case Key of
        '0'..'9', #8: ; // цифры и <Backspace>

        '.', ',', ':
            // Обработку десятичного разделителя
            // сделаем "интеллектуальной". Заменяем точку
            // и запятую на символ
            // FormatSettings.DecimalSeparator – символ,
            // который при текущей настройке операционной
            // системы должен использоваться при записи
            // дробных чисел.
            begin
                Key := FormatSettings.DecimalSeparator;
                // проверим, введен ли уже в поле Edit
                // десятичный разделитель
                if
                    pos(FormatSettings.DecimalSeparator, Edit1.Text) <> 0
                then Key := #0;
            end;
        #13: // клавиша <Enter>
            Edit2.SetFocus // Переместить курсор
                          // в поле Курс

        else Key := #0; // остальные символы запрещены
    end;
end;

```

```
// нажатие клавиши в поле Курс
procedure TForm1.Edit2KeyPress(Sender: TObject;
                                var Key: Char);
begin
    case Key of
        '0'..'9', #8: ; // цифры и <Backspace>

        '.', ',', ':
            begin
                Key := FormatSettings.DecimalSeparator;
            if
                pos(FormatSettings.DecimalSeparator, Edit1.Text) <> 0
            then Key := #0;
            end;
        #13: // клавиша <Enter>
            Edit2.SetFocus // Переместить фокус
                // на кнопку Ok

        else Key := #0; // остальные символы запрещены
    end;
end;
```

14. Написать программу, которая вычисляет доход по вкладу сроком на 1, 2, 3, 6 месяцев или на один год (предполагается, что процентная ставка зависит от срока вклада). Рекомендуемый вид формы приведен на рис. 1.12.

```
// щелчок на кнопке ОК
procedure TForm1.Button1Click(Sender: TObject);
var
    sum: real;           // сумма вклада
    period: integer;    // срок вклада (месяцев)
    percent: real;     // процент (годовых)
    profit: real;      // доход
    sum2: real;        // сумма в конце срока вклада

begin
    sum := StrToFloat(Edit1.Text);
    case RadioGroup1.ItemIndex of
```

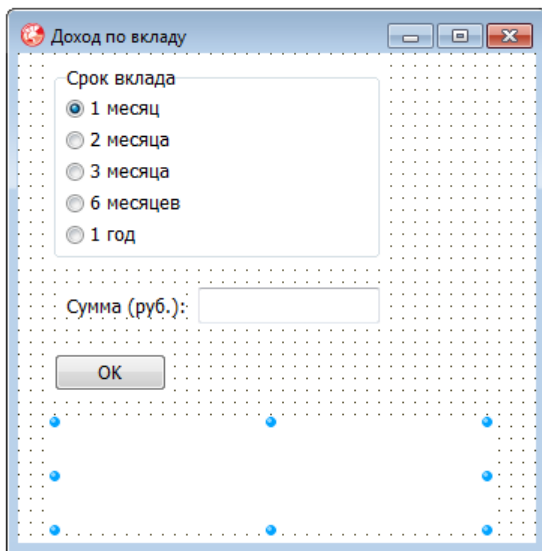


Рис. 1.12. Форма программы Доход по вкладу

```
0: begin
    period := 1;
    percent := 8;
end;
1: begin
    period := 2;
    percent := 8.5;
end;
2: begin
    period := 3;
    percent := 9;
end;
3: begin
    period := 6;
    percent := 10 ;
end;
4: begin
    period := 12;
    percent := 11;
end;
end;
```