Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический университет имени П. О. Сухого»

Кафедра «Информационные технологии»

ПРОГРАММИРОВАНИЕ В СРЕДЕ DELPHI

ЛАБОРАТОРНЫЙ ПРАКТИКУМ по курсу «Информатика» для студентов всех специальностей дневной формы обучения

Электронный аналог печатного издания

Гомель 2007

УДК 004.43(075.8)
ББК 32.973-018.1я73
П78

Рекомендовано к изданию научно-методическим советом факультета автоматизированных и информационных систем ГГТУ им. П. О. Сухого
(*протокол № 10 от 26.06.2006 г.*)

Автор-составитель: *В. И. Токочаков*

Рецензент: канд. техн. наук, доц. каф. «Электроснабжение» ГГТУ им. П. О. Сухого *Т. В. Алферова*

Программирование в среде Delphi : лаб. практикум по курсу «Информатика» для
П78 студентов всех специальностей днев. формы обучения / авт.-сост. В. И. Токочаков. – Гомель : ГГТУ им. П. О. Сухого, 2007. – 38 с.– Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: http://gstu.local/lib. – Загл. с титул. экрана.

ISBN 978-985-420-652-3.

Данный практикум является продолжением цикла лабораторных работ. Содержит материал по выполнению лабораторных работ в системе программирования Delphi. Приведены новые визуальные компоненты и их краткое описание.

Для студентов всех специальностей дневной формы обучения.

УДК 004.43(075.8)
ББК 32.973-018.1я73

ISBN 978-985-420-652-3

1. РАЗРАБОТКА ПРИЛОЖЕНИЙ В СРЕДЕ DELPHI

1.1. Разработка приложения Delphi, реализующего циклический алгоритм. Построение графика с использованием компонента PaintBox

Разработать приложение, позволяющее вычислять значение функции $y(x) = \sin(x)$ на интервале от $x_н$ до $x_к$ с шагом Δx. В главной форме должны находиться поля ввода значений $x_н$, $x_к$, Δx, таблица вывода значений аргумента и функции, рисунок с изображением графика функции, кнопка «Решение», кнопка «График», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать исходные данные, вычислить значение функции, поместить значения аргумента и функции в таблицу.

При нажатии кнопки «График» программа должна считать значения аргумента и функции из таблицы, вычислить минимальное и максимальное значения функции, нарисовать график функции, вывести значения границ диапазонов изменения аргумента и функции.

При нажатии кнопки «О программе» программа должна вывести окно сообщения о разработчике приложения.

При нажатии кнопки «Выход» приложение должно закрываться.

Вставляем в пустую форму компонент Edit1 для ввода значения $x_н$, компонент Edit2 для ввода значения $x_к$, компонент Edit3 для ввода значения Δx, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для вывода значений аргумента и функции, компонент PaintBox1 для выводы графика функции.

Познакомимся с новыми компонентами StringGrid и PaintBox. *Основными свойствами StringGrid* являются:

– Cells – матрица элементов таблицы строкового типа, первый индекс номер столбца, второй – номер строки, начиная с нуля;

– ColCount – количество столбцов целого типа;

– RowCount – количество строк целого типа;

– DefaultColWidth – ширина столбцов целого типа;

– DefaultRowHeight – высота строк целого типа;

– FixedColor – цвет фиксированной зоны;

– FixedCols – количество столбцов фиксированной зоны целого типа;

– FixedRows – количество строк фиксированной зоны целого типа;

– ColWidths – массив, содержащий ширину столбцов;

– Options – параметры таблицы, следует отметить параметр goEditing, предназначенный для разрешения редактирования таблицы.

Компонент PaintBox предназначен для рисования произвольных изображений. *Основными свойствами PaintBox* являются:

– Canvas – поле для рисования;

– Font – параметры настройки шрифтов;

– Pen – параметры настройки пера;

– Brush – параметры настройки кисти или основы.

Свойство Font имеет следующие свойства:

– Color – цвет шрифта;

– Height – высота шрифта в пикселах экрана;

– Style – стиль шрифта, принимает значения: fsBold – жирный, fsItalic – курсив, fsUnderline – подчеркнутый.

Свойство Pen имеет следующие свойства:

– Color – цвет вычерчиваемых пером линий;

– Width – толщина линий в пикселах;

– Style – стиль линий, принимает значения: psSolid – сплошная, psDash – штриховая длинная, psDot – штриховая короткая, psDashDot – штрихпунктирная.

Свойство Brush имеет следующие свойства:

– Color – цвет кисти или фона;

– Style – стиль кисти, принимает значения: bsSolid – сплошная черная, bsClear – сплошная белая, bsBDiagonal и bsFDiagonal – наклонная штриховка, bsCross и bsDiagCross – двойная штриховка, bsHorizontal – горизонтальная штриховка, bsVertical – вертикальная штриховка.

Свойство Canvas имеет следующие свойства:

– Font – параметры настройки шрифтов;

– Pen – параметры настройки пера;

– Brush – параметры настройки кисти или основы;

– PenPos – позиция пера в пикселах относительно левого верхнего угла канвы;

– Pixels – матрица пикселов канвы, имеет тип TColor.

Свойство Canvas имеет следующие методы:

– MoveTo(X,Y) – перемещает перо в положение (X,Y) без вычерчивания линий;

– LineTo(X,Y) – чертит линию от текущего положения пера до точки (X,Y);

– TextOut(X,Y,Text) – выводит строку Text так, чтобы левый верхний угол прямоугольника, охватывающего текст, располагался в точке (X,Y);

– другие методы вычерчивания дуги, окружности, хорды, эллипса, прямоугольника, сектора, многоугольника и т. д. [1], [2].

Форма приложения с исходными данными и результатами расчета показана на рис. 1.1.

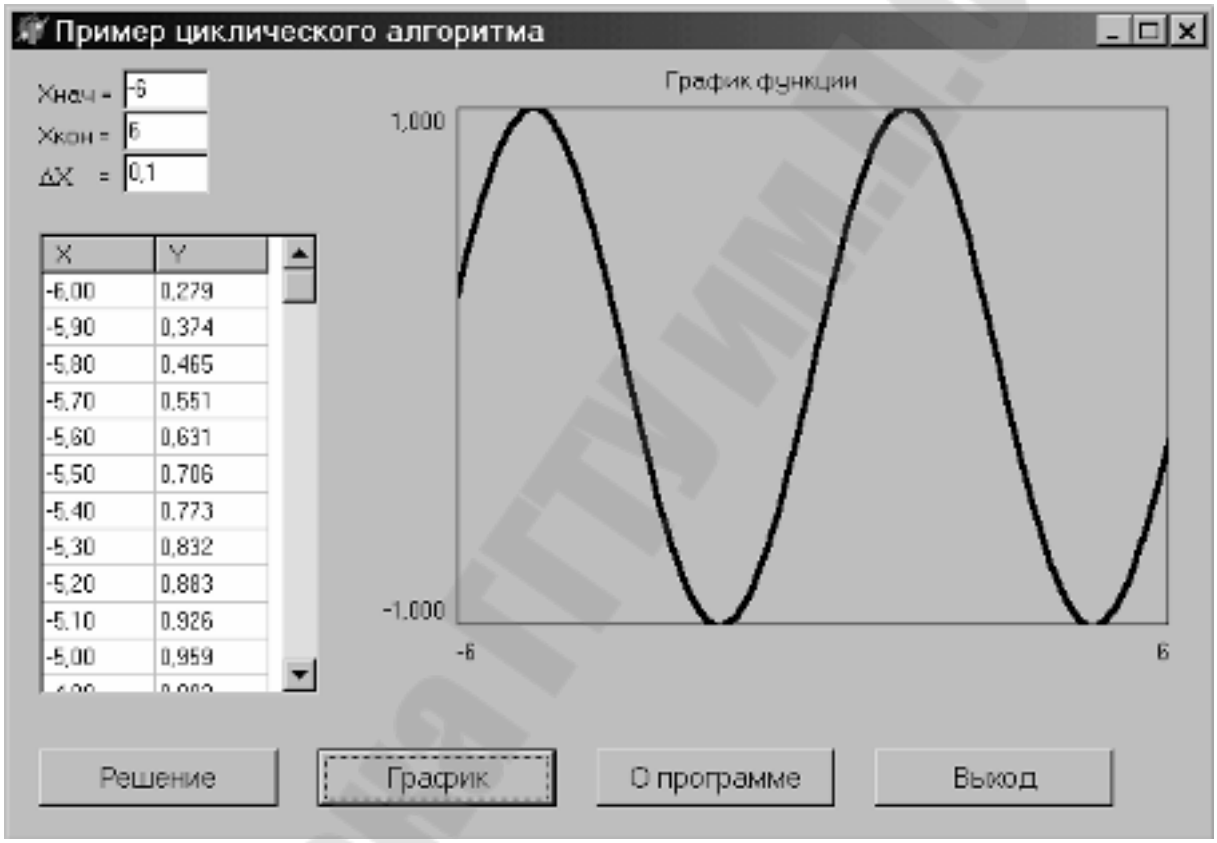

Рис. 1.1. Программа циклического алгоритма и рисование графика функции с использованием PaintBox

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, Grids, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
```

```
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    StringGrid1: TStringGrid;
    PaintBox1: TPaintBox;
    Label4: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
var  X, Xn, Xk, dX, Y : real;
// процедура формирования заголовка таблицы
procedure TForm1.FormCreate(Sender: TObject);
begin
   StringGrid1.Cells[0,0]:='    X';
   StringGrid1.Cells[1,0]:='    Y';
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
```

```
begin
  Application.MessageBox('Программу выполнил студент '+
    ' группы  ЛО-11 Макаров В.В.', 'О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
 var  i : integer;
begin
  Xn:= StrToFloat(Edit1.Text); // Чтение Хнач
  Xk:= StrToFloat(Edit2.Text); // Чтение Хкон
  dX:= StrToFloat(Edit3.Text); // Чтение шага по X
  X:=Xn;
  i:=1;
  while X<=Xk do  // цикл вывода значений X и Y в таблицу
    begin
     Y:=sin(X);
     StringGrid1.RowCount:=i+1;
     StringGrid1.Cells[0,i]:=FloatToStrF(X,ffFixed,6,2);
     StringGrid1.Cells[1,i]:=FloatToStrF(Y,ffFixed,6,3);
     X:=X+dX;
     i:=i+1;
    end;
end;
// процедура обработки кнопки "График"
procedure TForm1.Button2Click(Sender: TObject);
 var
  i, n : integer;
// минимальное и максимальное значения X
  maxX, minX : real;
// минимальное и максимальное значения Y
  maxY, minY : real;
begin
  PaintBox1.Refresh;  // обновление изображения PaintBox1
  PaintBox1.Canvas.Pen.Width:=2; // толщина пера 2 пиксела
  // рисование границ PaintBox1
  PaintBox1.Canvas.MoveTo(0,0);
  PaintBox1.Canvas.LineTo(0,PaintBox1.Height);
PaintBox1.Canvas.LineTo(PaintBox1.Width,PaintBox1.Height);
  PaintBox1.Canvas.LineTo(PaintBox1.Width,0);
  PaintBox1.Canvas.LineTo(0,0);
  minX:=StrToFloat(Edit1.Text);
  maxX:=StrToFloat(Edit2.Text);
  // отображение минимального значения X
  Label8.Caption:=Edit1.Text;
  // отображение максимального значения X
```

```
        Label9.Caption:=Edit2.Text;
        n:=StringGrid1.RowCount-1;
        maxY:=StrToFloat(StringGrid1.Cells[1,1]);
        for i:=2 to n do
         if maxY<StrToFloat(StringGrid1.Cells[1,i]) then
           maxY:=StrToFloat(StringGrid1.Cells[1,i]);
        minY:=StrToFloat(StringGrid1.Cells[1,1]);
        for i:=2 to n do
         if minY>StrToFloat(StringGrid1.Cells[1,i]) then
           minY:=StrToFloat(StringGrid1.Cells[1,i]);
        // отображение максимального значения Y
        Label6.Caption:=FloatToStrF(maxY,ffFixed,6,3);
        // отображение минимального значения Y
        Label7.Caption:=FloatToStrF(minY,ffFixed,6,3);
        PaintBox1.Canvas.Pen.Width:=3; // толщина пера 3 пиксела
        //перемещение пера в начальную точку графика
      PaintBox1.Canvas.MoveTo(Trunc((StrToFloat(StringGrid1.
      Cells[0,1])-minX)/(maxX-minX)*PaintBox1.Width),
         Trunc((maxY-StrToFloat(StringGrid1.Cells[1,1]))/(maxY-
minY)*PaintBox1.Height));
        //рисование графика
        for i:=1 to n do
          begin
           PaintBox1.Canvas.LineTo(Trunc((StrToFloat(StringGrid1.
             Cells [0,i])-minX)/(maxX-minX)*PaintBox1.Width),
             Trunc((maxY-StrToFloat(StringGrid1.Cells[1,i]))/
             (maxY-minY) *PaintBox1.Height));
          end;
        end;
      end.
```

1.2. Разработка приложения Delphi, реализующего циклический алгоритм. Построение графика с использованием компонента Chart

Разработать приложение, позволяющее вычислять значение функции $y(x) = -x\cos(x/2)$ на интервале от $x_н$ до $x_к$ с шагом Δx. В главной форме должны находиться поля ввода значений $x_н$, $x_к$, Δx, таблица вывода значений аргумента и функции, график функции, кнопка «Решение», кнопка «График», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать исходные данные, вычислить значение функции, поместить значения аргумента и функции в таблицу.

При нажатии кнопки «График» программа должна считать значения аргумента и функции из таблицы, вывести график функции.

Вставляем в пустую форму компонент Edit1 для ввода значения $x_н$, компонент Edit2 для ввода значения $x_к$, компонент Edit3 для ввода значения Δx, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для вывода значений аргумента и функции, компонент Chart1 для выводы графика функции.

Познакомимся с новым компонентом Chart. Основными свойствами и методами Chart являются:

– SeriesList – массив серий, которые выполняют построение каждого графика, номера серий начинаются с нуля;

– AddX, AddY, AddXY – методы добавления новых точек в график функции.

Компонент Chart автоматически масштабирует окно вывода графика. При нажатии на левую кнопку мыши в графической области изменяется масштаб графика, при нажатии на правую кнопку – смещается координата центра графической области.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.2.

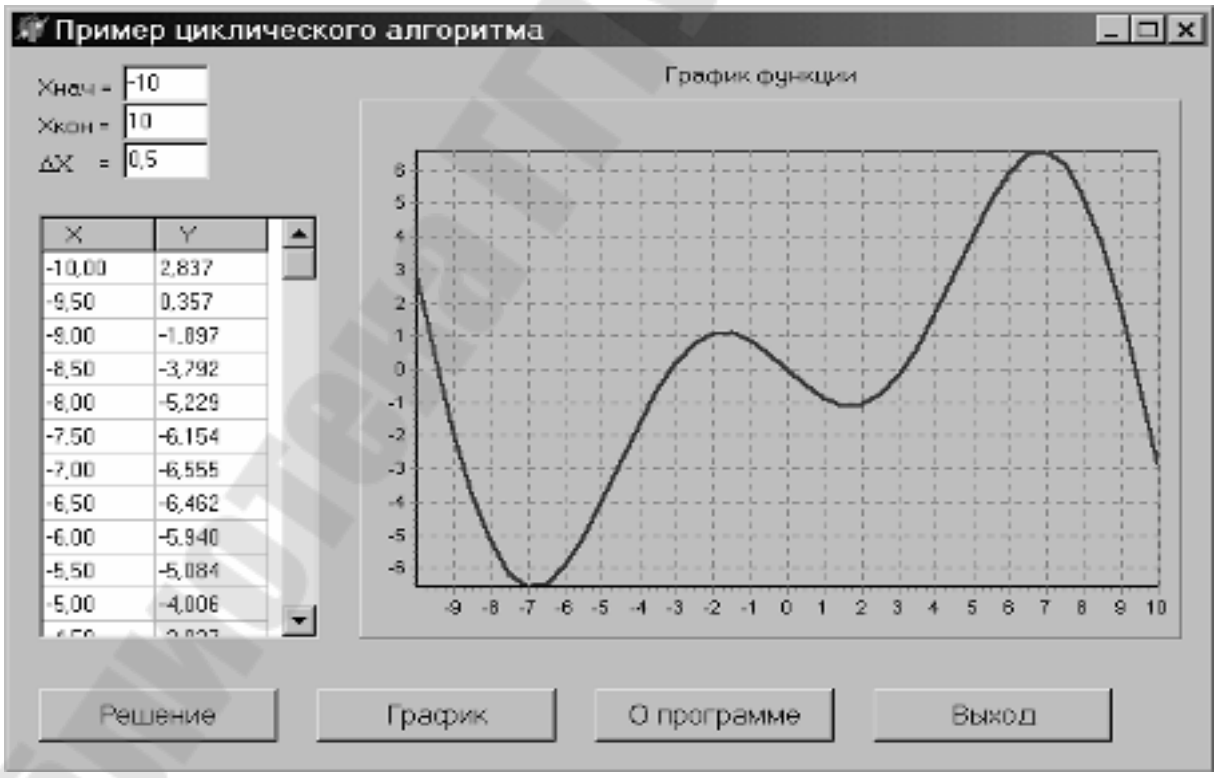

Рис. 1.2. Программа циклического алгоритма и вывода графика функции с использованием Chart

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,  Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs, TeEngine, Chart,
  Series;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    StringGrid1: TStringGrid;
    Label4: TLabel;
    Chart1: TChart;
    Series1: TLineSeries;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
var  X, Xn, Xk, dX, Y : real;
// процедура формирования заголовка таблицы
procedure TForm1.FormCreate(Sender: TObject);
begin
  StringGrid1.Cells[0,0]:='    X';
  StringGrid1.Cells[1,0]:='    Y';
end;
```

```
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
begin
    Application.MessageBox('Программу выполнил студент'+
      'группы Л-11 Иванов Л.И.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
 var   i : integer;
begin
  Xn:= StrToFloat(Edit1.Text); // Чтение Хнач
  Xk:= StrToFloat(Edit2.Text); // Чтение Хкон
  dX:= StrToFloat(Edit3.Text); // Чтение шага по X
  X:=Xn;
  i:=1;
  while X<=Xk do  // цикл вывода значений X и Y в таблицу
    begin
      Y:=-X*cos(X/2);
      StringGrid1.RowCount:=i+1;
      StringGrid1.Cells[0,i]:=FloatToStrF(X,ffFixed,6,2);
      StringGrid1.Cells[1,i]:=FloatToStrF(Y,ffFixed,6,3);
      X:=X+dX;
      i:=i+1;
    end;
end;
// процедура обработки кнопки "График"
procedure TForm1.Button2Click(Sender: TObject);
 var   i, n : integer;
begin
  n:=StringGrid1.RowCount-1;
  //очистка графической области
  Chart1.SeriesList[0].Clear;
  //рисование графика
for i:=1 to n do
 begin
  Chart1.SeriesList[0].AddXY(StrToFloat(StringGrid1.Cells[0,i]),
    StrToFloat(StringGrid1.Cells[1,i]),'',clRed);
   end;
end;
end.
```

1.3. Простейшие приемы работы с одномерными массивами

Разработать приложение, позволяющее подсчитать количество чисел, не принадлежащих промежутку (X,Y] и стоящих на четных местах в одномерном массиве вещественных чисел. В главной форме должны находиться таблица с одним столбцом или строкой, поле ввода размера массива, поля ввода величин X и Y, поля вывода результатов, кнопка «Решение», кнопка «О программе», кнопка «Выход».

Дополнительно введем кнопку «Установить размер массива», при нажатии на нее устанавливается новое значение свойства ColCount компонента StringGrid.

При нажатии кнопки «Решение» программа должна считать значения элементов массива, величин X и Y, вычислить значения результата и поместить их в поле вывода. Разработчик программы должен предусмотреть проверку значения аргумента на корректность ввода вещественного числа. При возникновении ошибки выполнения используем метод Application.MessageBox, а также конструкцию языка программирования try... except.

Вставляем в пустую форму компонент Edit1 для ввода размера массива, компонент Edit2 для ввода значения X, компонент Edit3 для ввода значения Y, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для ввода одномерного массива, компонент Edit4 для вывода результата.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.3.

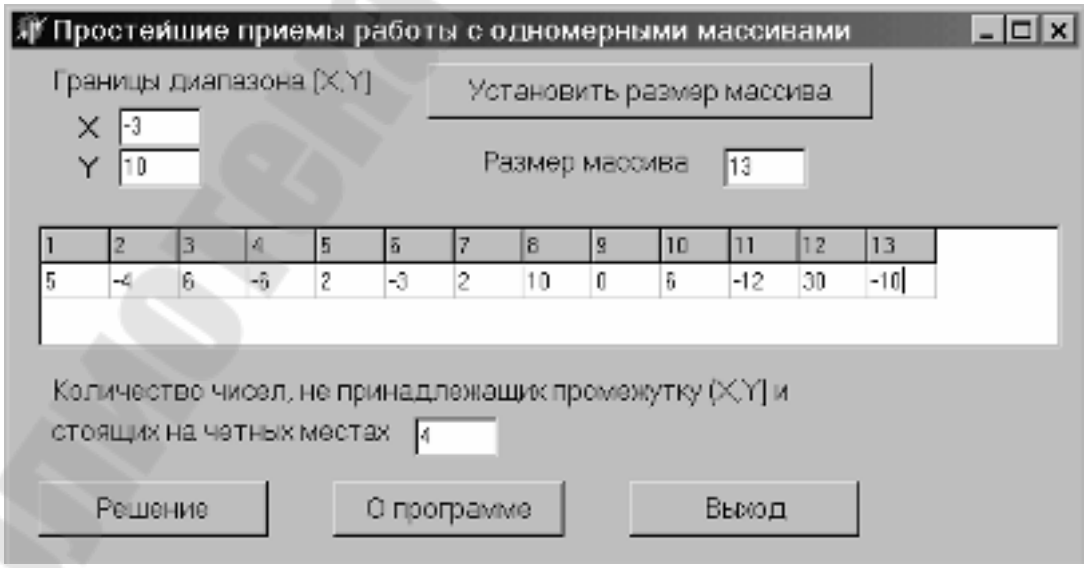

Рис. 1.3. Программа обработки одномерного массива

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs,
  TeEngine;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    StringGrid1: TStringGrid;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Edit4: TEdit;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
var
  X, Y : real;   // границы диапазона
  A : array [1..30] of real;   // одномерный массив
  N  : integer;  // Размер массива
  Kol  : integer;   // Результат
// процедура формирования заголовка таблицы
procedure TForm1.FormCreate(Sender: TObject);
```

```
var i : integer;
begin
  for i:=1 to 30 do
  StringGrid1.Cells[i-1,0]:= IntToStr(i);
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
begin
    Application.MessageBox('Программу выполнила студентка'+
       ' группы Д-11 Сычева Л.Н.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
 var   i : integer;
begin
  try
    X:=StrToFloat(Edit2.Text);
    Y:=StrToFloat(Edit3.Text);
    for i:=1 to N do
       A[i]:=StrToFloat(StringGrid1.Cells[i-1,1]);
  except
     Application.MessageBox('Ошибка в данных!',
        'Критическая ошибка');
     Exit;  // выход из процедуры
  end;
  Kol:=0;
  i:=2;
  while i<=N do  // цикл подсчета количества
     begin
       if (A[i]<=X)or(A[i]>Y) then
        Kol:=Kol+1;
        i:=i+2;
     end;
   Edit4.Text:=IntToStr(Kol);
end;
// процедура установки размера массива
procedure TForm1.Button2Click(Sender: TObject);
begin
   n:=StrToInt(Edit1.Text);
   StringGrid1.ColCount:=n;
end;
end.
```

1.4. Выделение минимального и максимального элементов массива

Разработать приложение, позволяющее поменять местами максимальный и минимальный элементы одномерного массива. В главной форме должны находиться две таблицы с одним столбцом или строкой, поле ввода размера массива, кнопка «Установить размер массива», кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать значения элементов массива из одной таблицы, найти минимальное и максимальное число, произвести обмен над элементами и поместить результирующий массив в другую таблицу. Разработчик программы должен предусмотреть проверку значения аргумента на корректность ввода исходных данных. При возникновении ошибки выполнения использовать функцию метод Application.MessageBox, а также конструкцию языка программирования try... except. Поиск минимального или максимального элемента оформить в виде подпрограммы.

Вставляем в пустую форму компонент Edit1 для ввода размера массива, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для ввода одномерного массива, таблицу StringGrid2 для вывода результирующего массива.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.4.

Рис. 1.4. Поиск минимального или максимального числа в массиве

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,  Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs, TeEngine;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Edit1: TEdit;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    StringGrid2: TStringGrid;
    Label3: TLabel;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
Type
  massiv = array [1..30] of real;
var
  A : massiv;   // одномерный массив
  N  : integer;  // Размер массива
// функция поиска номера максимального числа в массиве
Function NomerMax(B : massiv; M : integer): integer;
var i : integer; Max : real;
begin
  Max:=B[1];
  for i:=2 to M do
   if Max<B[i] then
```

```
    begin
      Max:=B[i];
      NomerMax:=i;
    end;
end;
// функция поиска номера минимального числа в массиве
Function NomerMin(B : massiv; M : integer): integer;
var i : integer; Min : real;
begin
  Min:=B[1];
  for i:=2 to M do
   if Min>B[i] then
    begin
      Min:=B[i];
      NomerMin:=i;
    end;
end;
// процедура формирования заголовков таблиц
procedure TForm1.FormCreate(Sender: TObject);
var i : integer;
begin
  for i:=1 to 30 do
    begin
      StringGrid1.Cells[i-1,0]:= IntToStr(i);
      StringGrid2.Cells[i-1,0]:= IntToStr(i);
    end;
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
begin
  Application.MessageBox('Программу выполнила студентка'+
     'группы  Д-12 Соколова Ю.А.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
var i : integer;
   temp: real; // временная переменная
begin
  try
    for i:=1 to N do
```

```
      A[i]:=StrToFloat(StringGrid1.Cells[i-1,1]);
    except
      Application.MessageBox('Ошибка в данных!',
        'Критическая ошибка');
      Exit;  // выход из процедуры
    end;
    // обмен чисел
    temp:=A[NomerMax(A,N)];
    A[NomerMax(A,N)]:=A[NomerMin(A,N)];
    A[NomerMin(A,N)]:=temp;
    // вывод результирующего массива
    for i:=1 to N do
      StringGrid2.Cells[i-1,1]:=FloatToStr(A[i]);
  end;
  // процедура установки размера массива
  procedure TForm1.Button2Click(Sender: TObject);
  begin
    n:=StrToInt(Edit1.Text);
    StringGrid1.ColCount:=n;
    StringGrid2.ColCount:=n;
  end;
  end.
```

1.5. Простейшие приемы работы с двумерными массивами

Разработать приложение, позволяющее вычислить сумму квадратов отрицательных элементов, стоящих в столбцах и строках с четными номерами. В главной форме должны находиться таблица размером $N \times M$, поля ввода размера массива, поле вывода результата, кнопка «Установить размер массива», кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать значения элементов двумерного массива и некоторых величин, вычислить значение результата и поместить его в поле вывода. Разработчик программы должен предусмотреть проверку значения аргумента на корректность ввода исходных данных. При возникновении ошибки выполнения использовать функцию метод Application.MessageBox, а также конструкцию языка программирования try... except.

Вставляем в пустую форму компонент Edit1 для ввода количества строк массива, компонент Edit2 для ввода количества столбцов массива, четыре кнопки Button1–Button4 для запуска соответствую-

щих процедур, таблицу StringGrid1 для ввода двумерного массива, компонент Edit3 для вывода результата.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.5.

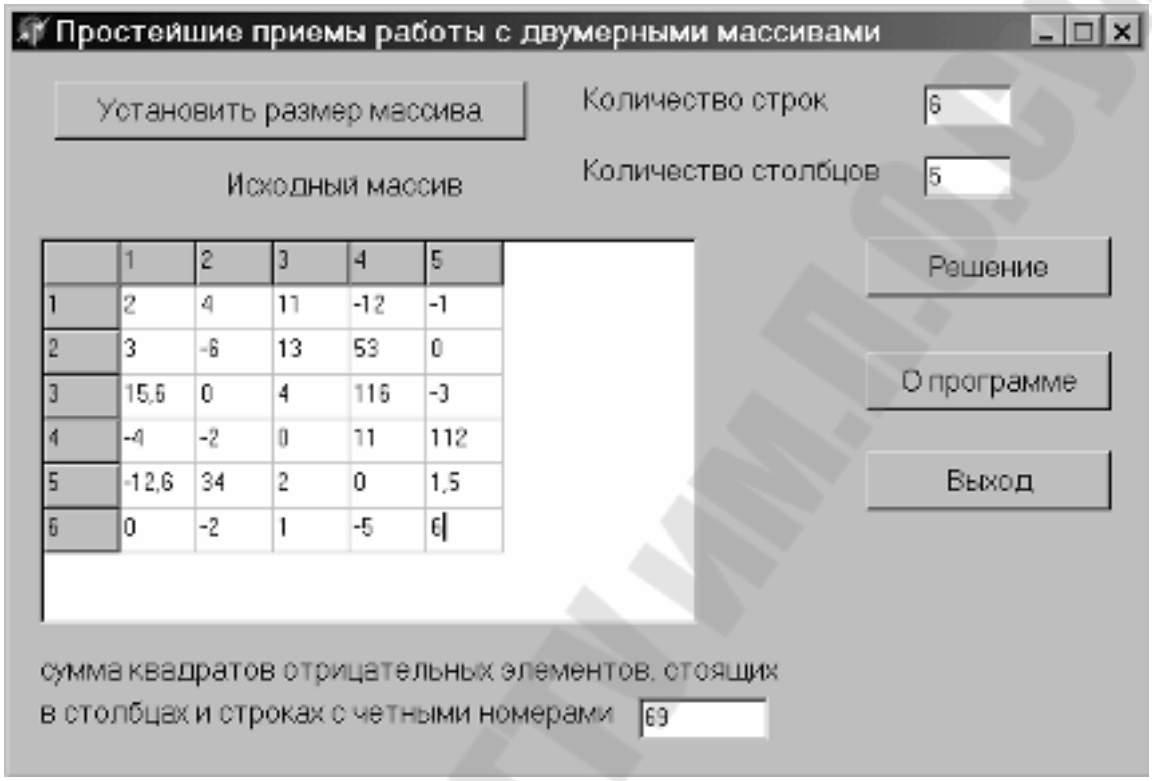

Рис. 1.5. Простейшие приемы работы с двумерными массивами

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs, TeEngine,
Chart,
  Series;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Edit1: TEdit;
```

```
    StringGrid1: TStringGrid;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Edit2: TEdit;
    Label5: TLabel;
    Edit3: TEdit;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
Type
  massiv = array [1..30,1..30] of real;
var
  A : massiv;   // двумерный массив
  N  : integer;  // Количество строк массива
  M  : integer;  // Количество столбцов массива
// процедура формирования заголовков таблиц
procedure TForm1.FormCreate(Sender: TObject);
var i, j : integer;
begin
  for i:=1 to 8 do
   for j:=1 to 8 do
    begin
      StringGrid1.Cells[i,0]:= IntToStr(i);
      StringGrid1.Cells[0,i]:= IntToStr(i);
    end;
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
  Close
end;
// процедура обработки кнопки "О программе"
```

```
procedure TForm1.Button3Click(Sender: TObject);
begin
  Application.MessageBox('Программу выполнила студентка '+
   'группы Т-11 Юркова Н.А.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
var i,j : integer;
 summa : real;
begin
  try
    for i:=1 to N do
     for j:=1 to M do
      A[i,j]:=StrToFloat(StringGrid1.Cells[j,i]);
  except
    Application.MessageBox('Ошибка в данных!',
        'Критическая ошибка');
    Exit;  // выход из процедуры
  end;
  summa:=0;
  i:=2;
  while i<=N do
    begin
      j:=2;
      while j<=M do
       begin
         if A[i,j]<0 then
           summa:=summa+sqr(A[i,j]);
         j:=j+2;
       end;
      i:=i+2;
    end;
  Edit3.Text:=FloatToStr(summa);
end;
// процедура установки размера массива
procedure TForm1.Button2Click(Sender: TObject);
begin
  n:=StrToInt(Edit1.Text);
  StringGrid1.RowCount:=n+1;
  m:=StrToInt(Edit2.Text);
  StringGrid1.ColCount:=m+1;
end;
end.
```

1.6. Простейшие приемы работы с квадратными матрицами

Разработать приложение, позволяющее вычислить количество элементов, не больших заданного числа *D* и стоящих выше главной диагонали квадратной матрицы. В главной форме должны находиться таблица размером $N \times N$, поля ввода размера массива, поле ввода числа *D*, поле вывода результата, кнопка «Установить размер массива», кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать значения элементов двумерного массива и величины *D*, вычислить значение результата и поместить его в поле вывода. Разработчик программы должен предусмотреть проверку значения аргумента на корректность ввода исходных данных. При возникновении ошибки выполнения использовать функцию метод Application.MessageBox, а также конструкцию языка программирования try... except.

Вставляем в пустую форму компонент Edit1 для ввода размера массива, компонент Edit2 для ввода числа *D*, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для ввода двумерного массива, компонент Edit3 для вывода результата.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.6.

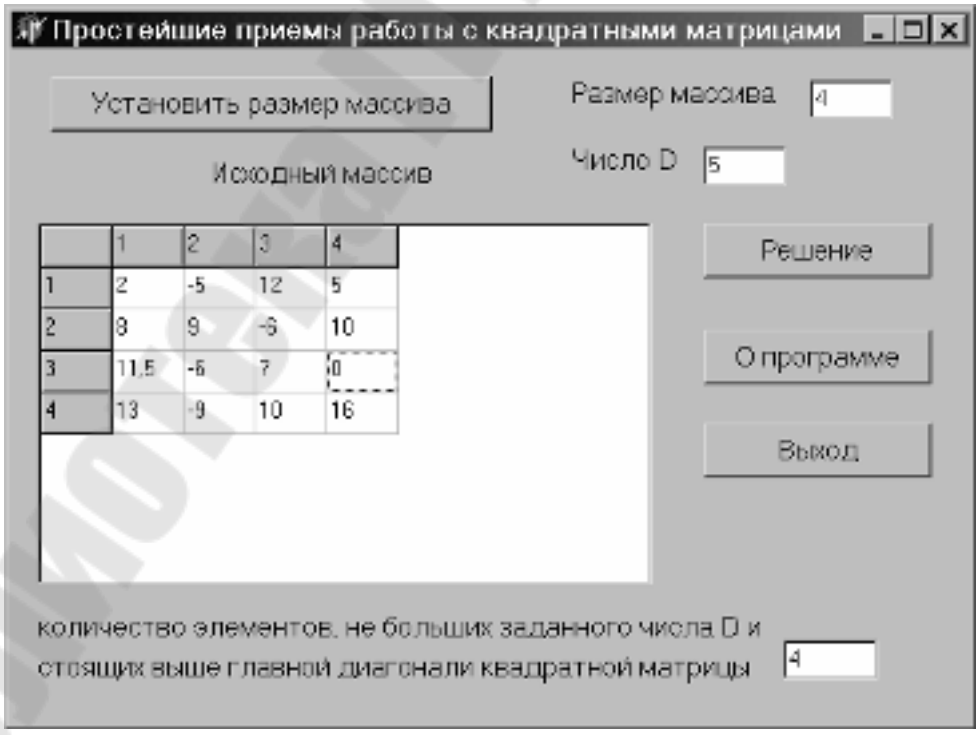

Рис. 1.6. Простейшие приемы работы с двумерными массивами

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs,
  TeEngine, Chart, Series;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Edit1: TEdit;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Edit2: TEdit;
    Label5: TLabel;
    Edit3: TEdit;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
Type
  massiv = array [1..30,1..30] of real;
var
  A : massiv;   // двумерный массив
  N  : integer;  // Размер массива
  D  : real;  // Число D
// процедура формирования заголовков таблиц
procedure TForm1.FormCreate(Sender: TObject);
var i,j : integer;
begin
```

```
  for i:=1 to 8 do
   for j:=1 to 8 do
    begin
      StringGrid1.Cells[i,0]:= IntToStr(i);
      StringGrid1.Cells[0,i]:= IntToStr(i);
    end;
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
begin
  Application.MessageBox('Программу выполнил студент '+
   'группы С-11 Сазонов Н.Н.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
var i, j : integer;
 Kol : integer;   // количество
begin
  try
    D:=StrToFloat(Edit2.Text);
    for i:=1 to N do
      for j:=1 to N do
       A[i,j]:=StrToFloat(StringGrid1.Cells[j,i]);
  except
     Application.MessageBox('Ошибка в данных!',
      'Критическая ошибка');
     Exit;  // выход из процедуры
  end;
  Kol:=0;
  for i:=1 to N do
     for j:=1 to N do
       if (A[i,j]<=D)and(i<j)then
         Kol:=Kol+1;
  Edit3.Text:=IntToStr(Kol);
end;
// процедура установки размера массива
procedure TForm1.Button2Click(Sender: TObject);
begin
  n:=StrToInt(Edit1.Text);
  StringGrid1.RowCount:=n+1;
  StringGrid1.ColCount:=n+1;
end;
end.
```

1.7. Простейшие приемы работы со строками

Разработать приложение, позволяющее подсчитывать количество пар символов «!?» и удалять из массива строк все символы «/». В главной форме должны находиться исходная таблица строк, таблица измененных строк, кнопка «Установить размер массива», кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать значения одномерного массива строк, выполнить некоторые действия над массивом строк, поместить результирующий массив в таблицу-результат. Действия над строкой оформить в виде функции или процедуры.

Вставляем в пустую форму компонент Edit1 для ввода размера массива строк, четыре кнопки Button1–Button4 для запуска соответствующих процедур, таблицу StringGrid1 для ввода массива строк, таблицу StringGrid2 для вывода результирующей таблицы, компонент Edit2 для вывода количества пар символов.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.7.

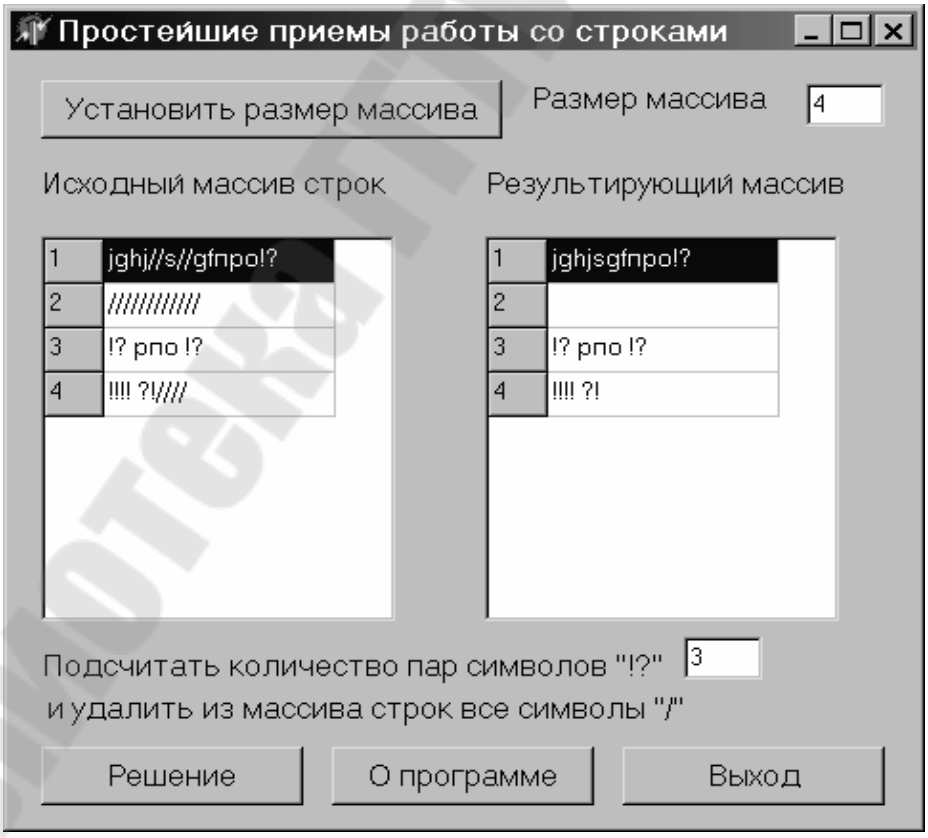

Рис. 1.7. Простейшие приемы работы со строками

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs, TeEngine,
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label1: TLabel;
    Edit1: TEdit;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    Label3: TLabel;
    StringGrid2: TStringGrid;
    Label4: TLabel;
    Label5: TLabel;
    Edit2: TEdit;
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
Type
  massiv = array [1..30] of string;
var
  A : massiv;   //  массив строк
  N  : integer;  // Размер массива
// функция подсчета количества символов "!?"
Function KolS(stroka:string): integer;
var i : integer;
begin
```

```
  Result:=0;
  for i:=1 to Length(stroka)-1 do
    if (stroka[i]='!')and(stroka[i+1]='?') then
      Result:=Result+1;
end;
// процедура удаления из строки символов "/"
Procedure Udal(var stroka:string);
var i : integer;
begin
  i:=1;
  while i<=Length(stroka) do
    if  stroka[i] = '/' then
      begin
       delete(stroka,i,1);
      end
     else
       i:=i+1;
end;
// процедура формирования заголовков таблиц
procedure TForm1.FormCreate(Sender: TObject);
var  i : integer;
begin
  // устанавливаем ширину столбцов таблиц
  StringGrid1.ColWidths[0]:=30;
  StringGrid2.ColWidths[0]:=30;
  StringGrid1.ColWidths[1]:=120;
  StringGrid2.ColWidths[1]:=120;
  for i:=1 to 10 do
    begin
      StringGrid1.Cells[0,i-1]:= IntToStr(i);
      StringGrid2.Cells[0,i-1]:= IntToStr(i);
    end;
end;
// процедура закрытия приложения
procedure TForm1.Button4Click(Sender: TObject);
begin
 Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button3Click(Sender: TObject);
begin
   Application.MessageBox('Программу выполнил студент '+
    'группы С-12 Суриков Н.Е.','О программе')
end;
// процедура обработки кнопки "Решение"
```

```
procedure TForm1.Button1Click(Sender: TObject);
var i : integer;
 Kol : integer;   // количество
begin
  for i:=1 to N do
    A[i]:=StringGrid1.Cells[1,i-1];
  Kol:=0;
  for i:=1 to N do
    begin
      Kol:=Kol+KolS(A[i]);
      Udal(A[i]);
      StringGrid2.Cells[1,i-1]:=A[i];
    end;
  Edit2.Text:=IntToStr(Kol);
end;
// процедура установки размера массива строк
procedure TForm1.Button2Click(Sender: TObject);
begin
  n:=StrToInt(Edit1.Text);
  StringGrid1.RowCount:=n;
  StringGrid2.RowCount:=n;
end;
end.
```

1.8. Простейшие приемы работы с многострочным текстом

Разработать приложение, позволяющее выделять слова, имеющих размер пять букв, в многострочном тексте. В главной форме должны находиться многострочный редактор Memo, кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна обработать набранный текст, найти необходимые слова и поместить результат в аналогичный многострочный редактор.

Вставляем в пустую форму компонент Memo1 для ввода текста, три кнопки Button1–Button3 для запуска соответствующих процедур, компонент Memo2 для вывода результирующего текста.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.8.

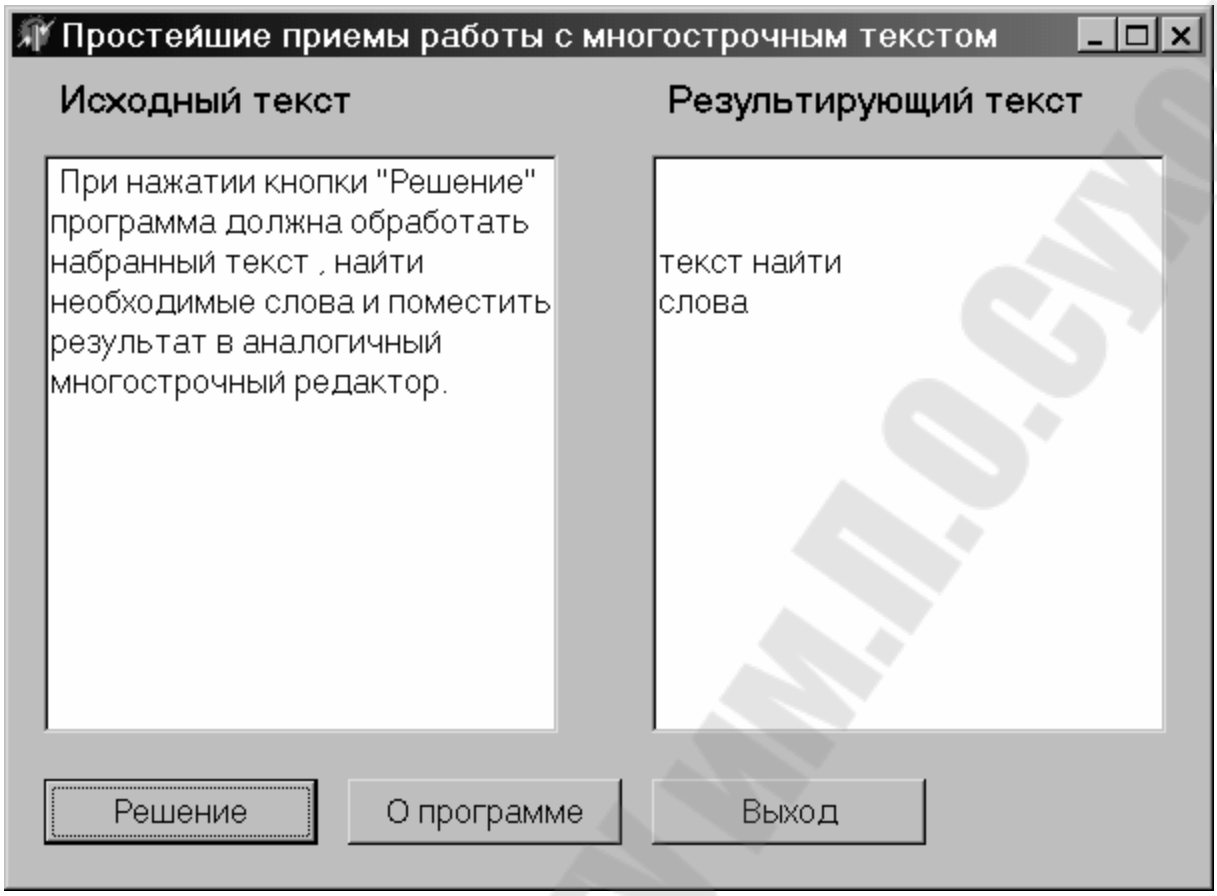

Рис. 1.8. Простейшие приемы работы с многострочным текстом

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs, TeEngine;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Label2: TLabel;
    Label3: TLabel;
    Memo1: TMemo;
    Memo2: TMemo;
    procedure Button3Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private
```

```
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
var N  : integer;
// процедура закрытия приложения
procedure TForm1.Button3Click(Sender: TObject);
begin
   Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button2Click(Sender: TObject);
begin
   Application.MessageBox('Программу выполнил студент '+
      'группы Д-11 Ульянов Н.С.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
var i,j,k,kk : integer;
 Stroka1, Stroka2  : string;
begin
  N:=Memo1.Lines.Count; // число строк в исходном тексте
  for i:=1 to N do
   begin
    Stroka1:=Memo1.Lines[i-1]+' ';  // считываем текущую строку
    k:=0;
    Stroka2:='';  // очищаем строку переноса слов
    kk:=Length(Stroka1);
    for j:=1 to kk do
     if Stroka1[j]<>' ' then k:=k+1
       else
         begin
           if (k=5) and (Stroka1[j]=' ') then
              Stroka2:=Stroka2+Stroka1[j-5]+Stroka1[j-4]+
                Stroka1[j-3]+Stroka1[j-2]+Stroka1[j-1]+' ';
           k:=0;
         end;
    Memo2.Lines.Add(Stroka2); // копируем слова из 5 букв
   end;
  end;
end.
```

1.9. Простейшие приемы работы с классами геометрических фигур

Разработать приложение, позволяющее определять площадь и периметр произвольного треугольника через его стороны. В главной форме должны находиться поля ввода исходных величин, поля вывода некоторых свойств фигуры, кнопка «Решение», кнопка «О программе», кнопка «Выход». Студент должен создать класс геометрической фигуры, включающий в себя поля, методы и свойства. При описании свойств фигуры необходимо иметь в виду, что некоторые свойства доступны только для чтения, другие – для записи, третьи – для чтения и записи.

При нажатии кнопки «Решение» программа должна считать значения исходных полей или свойств, выполнить некоторые действия над ними, поместить результаты в поля вывода.

Вставляем в пустую форму компонент Edit1–Edit2 для ввода сторон треугольника, три кнопки Button1–Button3 для запуска соответствующих процедур, компоненты Label1 и Label7 для вывода периметра и площади треугольника.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.9.

Рис. 1.9. Простейшие приемы работы с классами геометрических фигур

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

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

```
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Label1: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
type
  TTriang = class(TObject)
   a,b,c: Real; {стороны треугольника}
   alpha,beta,gamma: Real; {углы треугольника}
   function GetPerimetr : real;
   function GetPlosh_abc : real;
   function GetPlosh_abu : real;
   property Perimetr  :real read GetPerimetr;
   property Plosh_abc : real read GetPlosh_abc;
end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
var t: TTriang;
procedure TForm1.Button1Click(Sender: TObject);
begin
   t:=TTriang.Create; // создание объекта класса TTriang
   t.a:=StrToFloat(Edit1.text);
   t.b:=StrToFloat(Edit2.text);
   t.c:=StrToFloat(Edit3.text);
   Label1.Caption:=floattostrf(t.Perimetr,ffFixed,6,3);
   Label7.Caption:=floattostrf(t.Plosh_abc,ffFixed,6,3);
```

```
   t.Free;
end;
// описание метода определения периметра треугольника
Function TTriang.GetPerimetr : real;
  begin
    if (a+b>c)and(a+c>b)and(b+c>a)then Result:=a+b+c
       else Result:=0;
  end;
// описание метода определения площади треугольника
Function TTriang.GetPlosh_abc : real;
var p : real;
  begin
    if (a+b>c)and(a+c>b)and(b+c>a)then
       begin
         p:=(a+b+c)/2;
         Result:=sqrt(p*(p-a)*(p-b)*(p-c))
       end
        else Result:=0;
  end;
// процедура закрытия приложения
procedure TForm1.Button3Click(Sender: TObject);
begin
 Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button2Click(Sender: TObject);
begin
 Application.MessageBox('Программу выполнил студент '+
   'группы Д-11 Ульянов Н.С.','О программе')
end;
end.
```

1.10. Решения систем алгебраических уравнений

Разработать приложение, позволяющее производить решение системы линейных алгебраических уравнений третьего порядка методом Крамера. В главной форме должны находиться таблица основных коэффициентов системы уравнений, таблица-столбец свободных членов системы, таблица для вывода результата, кнопка «Решение», кнопка «О программе», кнопка «Выход».

При нажатии кнопки «Решение» программа должна считать значения исходных величин, решить систему уравнений, поместить результаты в таблицу вывода результата.

Вставляем в пустую форму компонент StringGrid1 для ввода матрицы коэффициентов A, компонент StringGrid2 для ввода вектора B, компонент StringGrid3 для вывода результата, три кнопки Button1–Button3 для запуска соответствующих процедур.

Форма приложения с исходными данными и результатами расчета показана на рис. 1.10.

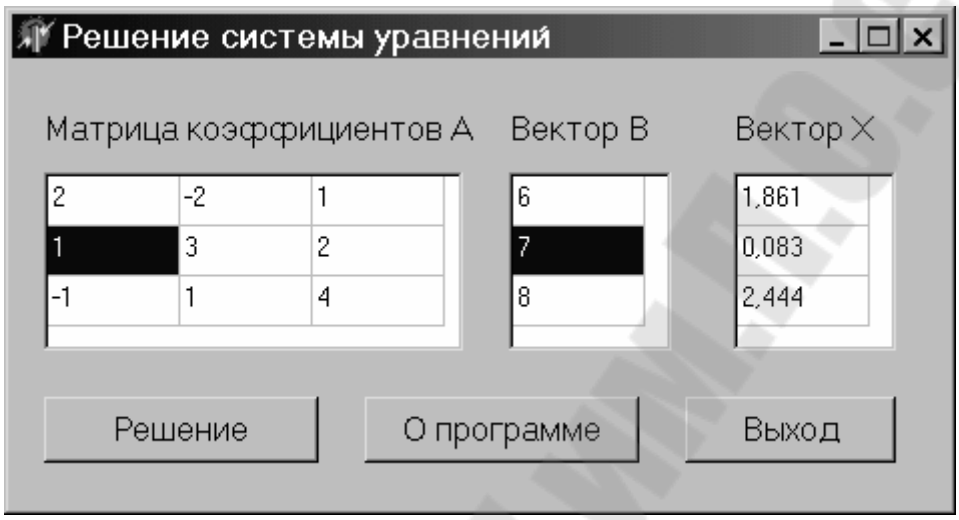

Рис. 1.10. Создание классов для решения систем алгебраических уравнений

Формируем процедуры обработки нажатия кнопок и компилируем программу. Текст модуля приводим ниже:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms, Dialogs, Grids, StdCtrls, ExtCtrls, TeeProcs,
  TeEngine, Chart, Series;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    StringGrid2: TStringGrid;
    Label1: TLabel;
    StringGrid3: TStringGrid;
    Label3: TLabel;
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
```

```
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
Type
 mas1 = array[1..3] of Real;
 mas3 = array[1..3,1..3] of Real;
var Form1:TForm1;
implementation
{$R *.dfm}
procedure Metod(A:mas3;B:mas1;var X:mas1);
var
d,d1,d2,d3 : real;
begin
  d :=A[1,1]*A[2,2]*A[3,3]+A[1,2]*A[2,3]*A[3,1]+A[2,1]*A[3,2]*A[1,3]-
      A[1,3]*A[2,2]*A[3,1]-A[1,2]*A[2,1]*A[3,3]-A[2,3]*A[3,2]*A[1,1];
  d1:=B[1]*A[2,2]*A[3,3]+A[1,2]*A[2,3]*B[3]+B[2]*A[3,2]*A[1,3]-
      A[1,3]*A[2,2]*B[3]-A[1,2]*B[2]*A[3,3]-A[2,3]*A[3,2]*B[1];
  d2:=A[1,1]*B[2]*A[3,3]+B[1]*A[2,3]*A[3,1]+A[2,1]*B[3]*A[1,3]-
      A[1,3]*B[2]*A[3,1]-B[1]*A[2,1]*A[3,3]-A[2,3]*B[3]*A[1,1];
  d3:=A[1,1]*A[2,2]*B[3]+A[1,2]*B[2]*A[3,1]+A[2,1]*A[3,2]*B[1]-
      B[1]*A[2,2]*A[3,1]-A[1,2]*A[2,1]*B[3]-B[2]*A[3,2]*A[1,1];
  x[1]:=d1/d;
  x[2]:=d2/d;
  x[3]:=d3/d;
end;
// процедура закрытия приложения
procedure TForm1.Button3Click(Sender: TObject);
begin
 Close
end;
// процедура обработки кнопки "О программе"
procedure TForm1.Button2Click(Sender: TObject);
begin
  Application.MessageBox('Программу выполнила студентка '+
    'группы ТЭ-11 Юсова Н.Е.','О программе')
end;
// процедура обработки кнопки "Решение"
procedure TForm1.Button1Click(Sender: TObject);
var i,j : integer;
A : mas3;
B, X : mas1;
begin
```

```
  for i:=1 to 3 do
    for j:=1 to 3 do
      A[i,j]:=StrToFloat(StringGrid1.Cells[j-1,i-1]);
  for i:=1 to 3 do
      B[i]:=StrToFloat(StringGrid2.Cells[0,i-1]);
  Metod(A,B,X);
  for i:=1 to 3 do
      StringGrid3.Cells[0,i-1]:=FloatToStrF(X[i],ffFixed,6,3);
end;
end.
```

ЛИТЕРАТУРА

1. Фаронов, В. В. Delphi. Программирование на языке высокого уровня : учеб. для вузов / В. В. Фаронов. – Санкт-Петербург : Питер, 2005. – 640 с.

2. Фаронов, В. В. Delphi 5. Учебный курс / В. В. Фаронов. – Москва : Нолинж, 2001. – 608 с.

3. Программирование в среде Delphi : практ. пособие по курсу «Информатика» для студентов всех специальностей днев. отд-ния / авт.-сост.: Е. В. Коробейникова, В. И. Токочаков. – Гомель : ГГТУ им. П. О. Сухого, 2004. – 24 с.

4. Программирование в среде Delphi : практикум к лаб. работам по курсу «Информатика» для студентов всех специальностей днев. формы обучения / авт.-сост.: В. И. Токочаков, Е. В. Коробейникова, К. С. Курочка. – Гомель: ГГТУ им. П. О. Сухого, 2004. – 25 с.

СОДЕРЖАНИЕ

Учебное электронное издание комбинированного распространения

Учебное издание

ПРОГРАММИРОВАНИЕ В СРЕДЕ DELPHI

Лабораторный практикум по курсу «Информатика для студентов всех специальностей дневной формы обучения

Автор-составитель: **Токочаков** Владимир Иванович

Редактор *Н. Г. Мансурова*
Компьютерная верстка *Н. В. Широглазова*

Подписано в печать 20.11.07.
Формат 60x84/16. Бумага офсетная. Гарнитура Таймс.
Ризография. Усл. печ. л. 2,32. Уч.-изд. л. 2,1.
Изд. № 123.
E-mail: ic@gstu.gomel.by
http://www.gstu.gomel.by

Издатель и полиграфическое исполнение:
Издательский центр учреждения образования
«Гомельский государственный технический
университет имени П. О. Сухого».
ЛИ № 02330/0131916 от 30.04.2004 г.
246746, г. Гомель, пр. Октября, 48.