Томский государственный университет

Факультет информатики

А.В. Скворцов, Т.Н. Поддубная

Построение графиков функций

(методические указания к

лабораторной работе № 6)

Томск – 1999

Указания РАССМОТРЕНЫ и УТВЕРЖДЕНЫ методической комиссией факультета информатики.

Протокол №____ от «____»_____1999 г.

Председатель методической комиссии факультета информатики,

профессор _____ Поддубный В.В.

Утверждаю. Зав. кафедрой прикладной информатики, доцент

_____ Сущенко С.П. «____» _____ 1999 г.

Методические указания посвящены программной среде Borland Delphi компании Inprise Corporation – одному из мощнейших современных средств разработки приложений для Windows.

Указания разработаны для студентов межфакультетской специализации факультета информатики Томского государственного университета. Содержат подробное описание лабораторной работы: цель, задачи, описание работы и используемых компонентов Delphi. В конце работы приведены тексты программ.

Резенцент – канд. техн. наук, доцент **Ю.Л. Костюк**.

Данная лабораторная работа предназначена для знакомства с основами создания графических изображений. На примере простой программы построения графика функции, заданной таблично, необходимо изучить такие графические элементы, как перо, кисть, шрифт, линии, прямоугольники и т.д.; познакомиться с компонентом `TPaintBox` для изображения произвольной графики и классом `TCanvas`, объединяющим все основные примитивы для рисования графики в Delphi.

Создаваемое приложение должно состоять из таблицы для задания списка точек графика функции и графического поля для рисования графика. При этом любое изменение данных в таблице должно вызывать обновление графика.

Цель работы

Изучение средств Delphi для рисования произвольных изображений на экране.

Задачи работы

1. Ознакомление с объектами для создания произвольных графических изображений.
2. Создание программы для построения графиков функций.

Описание работы

Наша программа будет состоять всего из одной формы, на которой необходимо разместить таблицу для ввода координат точек, панели с компонентом `TPaintBox` для отображения графика функции, панели для отображения ошибок ввода данных, а также кнопки выхода из программы.

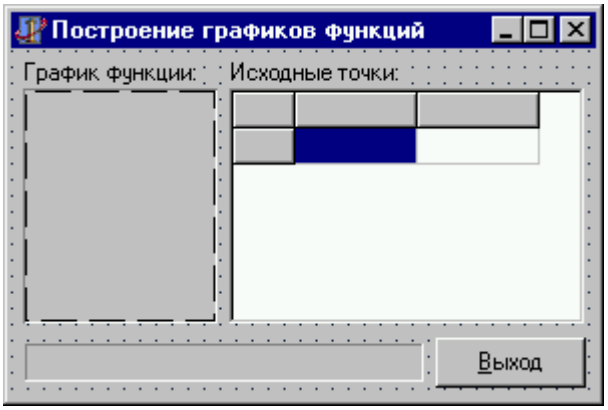

Рис. 19. Внешний вид главного окна приложения

Таблица для ввода координат точек должна автоматически изменять размер в соответствии с количеством уже введенных точек функции. Как это сделать, описано в предыдущих лабораторных работах.

После ввода данных необходимо сразу же сформировать два массива введенных координат *X* и *Y*, отсортировать их по оси *X*, после чего необходимо перерисовать окно с графиком.

Теперь рассмотрим, как Windows отображает все свои окна. Окнами в самом общем случае являются кнопки, строки ввода, списки и т.д. Они могут находиться друг в друге, например кнопки в диалоговом окне или окна внутри многодокументного интерфейса. Каждое окно при этом имеет специальную главную управляющую процедуру, которая обрабатывает поступающие от системы сообщения. Некоторые из этих сообщений Delphi преобразует в события, которые можно затем обрабатывать в программе. Windows сама никогда не хранит изображения окон, поэтому одним из сообщений является команда о необходимости отображения содержимого окна на экране.

Замечание

При необходимости перерисовки компонента в ответ на какое-то событие, например нажатие кнопки, сразу не рисуйте, а вызовите метод `Invalidate` *и создайте обработчик события* `OnPaint` *визуального компонента, например формы или* `TPaintBox`.

В нашей программе сразу же после ввода координат точек мы не должны вызывать процедуру рисования. При этом мы должны указать Windows, что как только появится свободное процессорное время, система должна послать сообщение о необходимости реального отображения окна. При таком подходе, если окно приложения будет временно закрыто другим окном и опять станет видимым, система автоматически пошлет сообщение о необходимости восстановления графического изображения в окне.

В Delphi для указания, что необходимо перерисовать любой элемент управления, нужно вызвать его метод `Invalidate`.

Для изображения произвольной графики в Delphi обычно используется компонент `TPaintBox`, находящийся на закладке System палитры компонентов. Этот компонент имеет событие `OnPaint`, для которого мы и должны создать обработчик для рисования графика.

Для рисования произвольных изображений в Delphi используется класс `TCanvas`. В табл. 11-14 приведено краткое описание свойств и методов объектов для рисования графики.

Для упрощения алгоритма рисования графика будем считать, что обе координаты всех точек попадают в интервал [–10, 10]. Процедура рисования графика при этом распадается на две части: рисование осей координат с засечками и подписями и собственно изображение графика.

Теперь рассмотрим вопрос автоматического изменения положения и размеров визуальных компонентов при изменении формы. Для этого у компонентов имеются такие свойства, как `Align` и `Anchors`. Собственно, свойство `Align` является подмножеством возможностей свойства `Anchors`.

Свойство Anchors имеет 4 логических подсвойства: akLeft, akTop, akRight, akBottom. Например, если свойство akRight установлено в True, то правая сторона элемента будет находиться на одинаковом расстоянии от правой стороны содержащей его формы при изменении пользователем ее размера, иначе правая сторона элемента будет находиться на одинаковом расстоянии от левой стороны. При изменении свойства Align на самом деле изменяется свойство Anchors, которое впоследствии и учитывается при изменении размеров.

Таблица 11. Основные свойства объектов типа TCanvas

Свойство	Тип	Комментарий
Brush	TBrush	Параметры заливки сплошных областей
Pen	TPen	Параметры отображения линий
Font	TFont	Параметры отображения текстовых надписей
Pixels[X,Y]	TColor	Цвет точки с координатами (*X*,*Y*)

Таблица 12. Основные методы объектов типа TCanvas

Метод	Комментарий
Ellipse(X1,Y1,X2,Y2)	Изображает эллипс, вписанный в прямоугольник с заданными координатами
LineTo(X,Y)	Рисует линию из текущей позиции в новую точку
MoveTo(X,Y)	Устанавливает текущую позицию
Polyline(array of TPoint)	Рисует полилинию по набору точек
Polygon(array of TPoint)	Рисует полигон по набору точек
Rectangle(X1,Y1,X2,Y2)	Рисует прямоугольник по координатам двух противоположных углов
TextOut(X,Y,Text)	Выводит текст в заданной позиции

Таблица 13. Основные свойства объектов типа `TBrush`.

Свойство	Тип	Комментарий
`Color`	`TColor`	Цвет заливки
`Style`	`TBrushStyle`	Стиль заливки (`bsSolid` – сплошная, `bsClear` – нет, `bsHorizontal` – горизонтальные линии, `bsVertical` – вертикальные)

Таблица 14. Основные свойства объектов типа `TPen`.

Свойство	Тип	Комментарий
`Color`	`TColor`	Цвет линий
`Style`	`TPenStyle`	Стиль линий (`psSolid` – сплошная, `psDash` – штрих, `psDot` – точки, `psDashDot` – штрих-пунктир, `psDashDotDot` – штрих- штрих-пунктир, `psClear` – нет линии)
`Width`	`Integer`	Ширина сплошной линии

Таблица 15. Основные свойства объектов типа `TFont`

Свойство	Тип	Комментарий
`Color`	`TColor`	Цвет шрифта
`Name`	`String`	Имя шрифта
`Size`	`Integer`	Размер шрифта
`Style`	`TFontStyles`	Стили шрифта: жирность, наклон, подчеркивание, зачеркивание

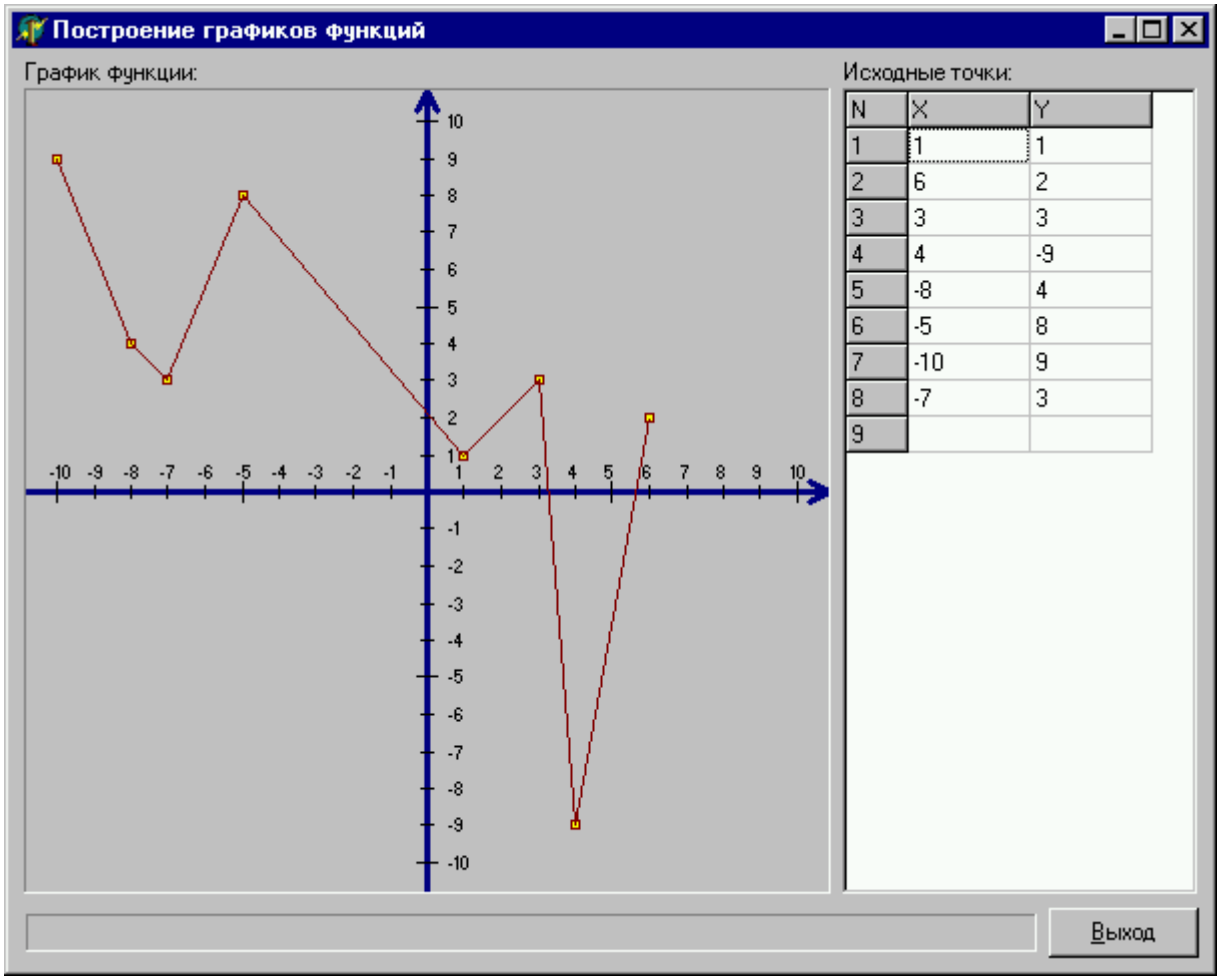

Рис. 19. Внешний вид запущенного приложения

Поэкспериментируйте с приведенными свойствами и добейтесь, чтобы в нашем приложении график растягивался в размерах при увеличении размеров формы, но в то же время таблица оставалась с правой стороны формы, а кнопка выхода в правом нижнем. Обратите внимание, что процедура рисования графика должна учитывать изменяющийся размер элемента `TPaintBox`.

На рис. 19 приведен внешний вид запущенного приложения после того, как окно растянуто. В листинге 13 приведен текст файла модуля.

Замечание

По возможности делайте любое окно растягиваемым, но при этом не забывайте отслеживать, чтобы внутренности окна изменялись при изменении его размеров. Для этого используйте свойства `Align` *и* `Anchors`.

Листинг 13. Текст главного модуля Charts.pas

```
unit Charts;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, ExtCtrls, StdCtrls, Menus;

type
  TChartForm = class(TForm)
    StringGrid: TStringGrid;
    PanelChart: TPanel;
    ButtonExit: TButton;
    Label1: TLabel;
    Label2: TLabel;
    PaintBox: TPaintBox;
    PanelStatus: TPanel;
    procedure ButtonExitClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure StringGridSetEditText(Sender: TObject; ACol, ARow: Integer;
      const Value: String);
    procedure PaintBoxPaint(Sender: TObject);
  private
    X,Y: array of double;
  end;

var
  ChartForm: TChartForm;

implementation

{$R *.DFM}

procedure TChartForm.ButtonExitClick(Sender: TObject);
begin // Выход из программы
  Close;
end;

procedure TChartForm.FormCreate(Sender: TObject);
begin // При создании формы надо проинициализировать таблицу
  with StringGrid do
  begin // Формирование заголовков столбцов таблицы
    Cells[0,0]:='N';
    Cells[1,0]:='X';
    Cells[2,0]:='Y';
    Cells[0,1]:='1';
  end;
end;

procedure TChartForm.StringGridSetEditText(Sender: TObject; ACol, ARow: Integer;
  const Value: String);
var i, maxRow: integer; f: boolean; tmp: double;
begin // В таблице изменился текст
  try
    if Value<>'' then
      StrToFloat(Value);
    PanelStatus.Caption:='';
  except
    on E: Exception do
      PanelStatus.Caption:=E.Message;
```

```
  end;
  with StringGrid do
  begin // Удаление лишних строк
    MaxRow:=0;
    for i:=RowCount-1 downto 1 do
      if (Cells[1,i]<>'') or (Cells[2,i]<>'') then
      begin
        MaxRow:=i;
        break;
      end;
    RowCount:=MaxRow+2;
    for i:=1 to RowCount-1 do
      Cells[0, i]:=IntToStr(i);
    // Заполнение массивов координат
    SetLength(X, MaxRow); SetLength(Y, MaxRow);
    for i:=0 to MaxRow-1 do
    begin
      try
        X[i]:=StrToFloat(Cells[1,i+1]);
      except
        X[i]:=0;
      end;
      try
        Y[i]:=StrToFloat(Cells[2,i+1]);
      except
        Y[i]:=0;
      end;
    end;
    // Сортировка массива по координате X методом пузырька
    repeat
      f:=True;
      for i:=0 to MaxRow-2 do
        if X[i]>X[i+1] then
        begin
          tmp:=X[i]; X[i]:=X[i+1]; X[i+1]:=tmp;
          tmp:=Y[i]; Y[i]:=y[i+1]; Y[i+1]:=tmp;
          f:=False;
        end;
    until f;
    PaintBox.Invalidate; // Перерисовать график
  end;
end;

procedure TChartForm.PaintBoxPaint(Sender: TObject);
var CX, CY, i, j, XX, YY: integer; D: double;
begin // Вызывается при необходимости перерисовки графика
  with PaintBox, Canvas do
  begin
    CX:=Width div 2; CY:=Height div 2; // Вычисляем положение начала координат в PaintBox
    if Width<Height then D:=(Width-30)/20 else D:=(Height-30)/20;
    // Рисуем координатные оси
    Pen.Color:=clNavy;
    Pen.Width:=3;
    Polyline([Point(0,CY), Point(Width,CY), Point(Width-10,CY+5), Point(Width,CY),
      Point(Width-10,CY-5)]);
    Polyline([Point(CX,Height), Point(CX,0), Point(CX+5,10), Point(CX,0), Point(CX-5,10)]);
    // Рисуем насечки на осях и подписи
    Pen.Color:=clBlack; // Цвет линий сделать черным
    Pen.Width:=1; // Толщину линий установить единичной
    Font.Name:='Small'; // Название шрифта "Small" (специальный шрифт для мелких надписей)
    Font.Size:=7; // Размер шрифта
```

```
    Brush.Style:=bsClear; // Заливку отключить
    for i:=-10 to 10 do
      if i<>0 then
      begin
        if i mod 5 = 0 then j:=5 else j:=3;
        Polyline([Point(CX+round(i*D),CY-j), Point(CX+round(i*D),CY+j+1)]);
        Polyline([Point(CX-j,CY+round(i*D)), Point(CX+j+1,CY+round(i*D))]);
        TextOut(CX+round(i*D)-4,CY-15,IntToStr(i));
        TextOut(CX+10, CY-round(i*D)-5,IntToStr(i));
      end;
    // Строим график
    Pen.Color:=clMaroon;
    Brush.Color:=clYellow;
    for i:=0 to High(X) do
    begin
      XX:=CX+round(X[i]*D); YY:=CY-round(Y[i]*D);
      Rectangle(XX-2,YY-2,XX+3,YY+3);
      if i=0 then MoveTo(XX,YY) else LineTo(XX,YY);
    end;
  end;
end;

end.
```

Вопросы и задания для самостоятельной работы

1. Как Windows перерисовывает свои окна?
2. Как заставить Windows перерисовать какое-то окно?
3. Для чего используются свойства визуальных компонентов `Align` и `Anchors`?
4. Поместите на форму компоненты для выбора стиля изображения на графике введенных точек (кружочки, квадратики или «не изображать»), толщины и цвета линий. При этом график должен автоматически перерисовываться при изменении этих параметров.
5. Добавьте кнопку для генерации случайного набора точек.
6. Добавьте возможность автоматического подбора масштаба координатных осей для отображения всех исходных данных.

Учебное издание

Алексей Владимирович Скворцов
Поддубная Тамара Николаевна

Построение графиков функций

(методические указания к
лабораторной работе № 6)

Томский государственный университет. Томск, 1999. – 12 с.