Томский государственный университет

Факультет информатики

А.В. Скворцов, Т.Н. Поддубная

Работа с графами

(методические указания к

лабораторной работе № 4)

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**.

В данной лабораторной работе необходимо будет разработать приложение для нахождения кратчайшего маршрута на графе. В рамках этой задачи потребуется изучить новые визуальные компоненты, а также закрепить навыки работы с динамическими массивами.

Приложение будет иметь одну форму, на которой расположены три панели с закладками. Каждая из панелей содержит интерфейсные элементы для определенных задач при программировании алгоритмов с графами: ввод графа, вычисление его характеристик, нахождение путей в графе.

Цель работы

Закрепление навыков работы с динамическими массивами и изучение новых интерфейсных элементов Delphi.

Задачи работы

1. Ознакомление с компонентами класса `TMemo` и `TPageControl`, их свойствами и методами.
2. Изучение и программирование алгоритмов с графами: ввод графа в виде списка ребер, построение матрицы смежности, поиск простого пути алгоритмом Дейкстры.

Описание работы

Как обычно, перед началом работы необходимо создать новый проект и сохранить его в отдельном каталоге Lab4 в соответствии с рекомендациями, приведенными в предыдущих лабораторных работах.

В данном приложении требуется создать одну форму, на которую необходимо поместить компоненту `TPageControl` со страницы Win32 палитры компонент Delphi. Эта компонента содержит несколько перекрывающих друг друга панелей с закладками типа `TTabSheet`. На рис. 13-15 приведен внешний вид создаваемого приложения. Рассмотрим подробнее некоторые из этапов создания экранной формы.

Непосредственно внутри формы необходимо разместить два компонента – один типа `TPageControl` и кнопку закрытия окна. Чтобы добавить новые закладки внутрь компонента `TPageControl`, необходимо на нем с помощью правой кнопкой мышки вызвать локальное меню и выбрать в нем команду New Page. После этого можно приступать к наполнению соответствующих страниц элементами управления.

Первая страница с закладкой «Ввод» предназначена для ввода списка ребер графа. Каждое ребро будет характеризоваться номерами начальной и конечной вершин, а также своим весом. Поэтому на первой странице необходимо разместить компонент `TStringGrid` с одной фиксированной и тремя редактируемыми колонками. Кроме того, справа можно добавить необязательные кнопки для вставки и удаления ребер. Ниже таблицы необходимо разместить компонент типа `TCheckBox` для указания, является ли граф ориентированным или нет.

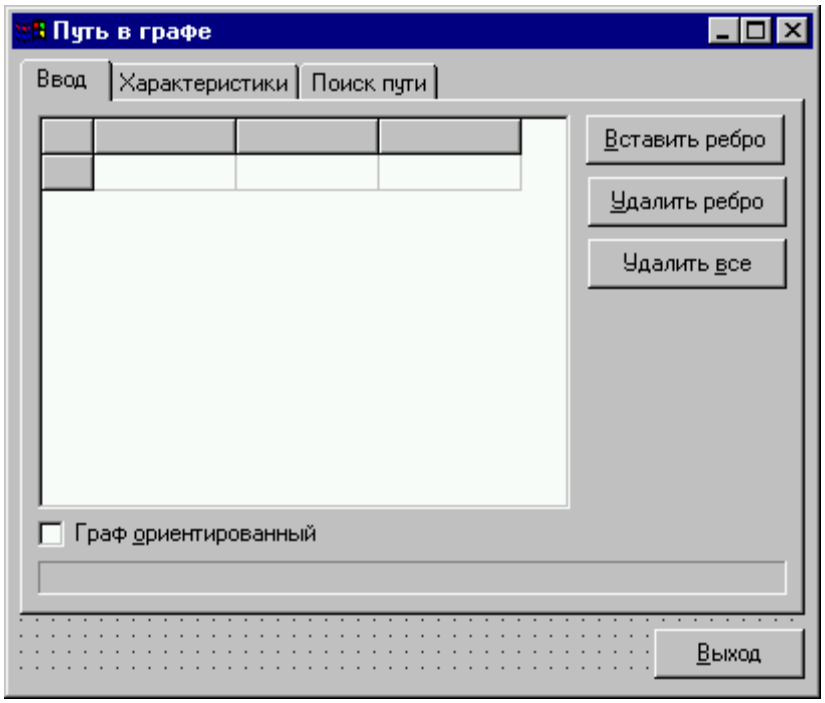

Рис. 13. Страница для ввода ребер графа

При вводе номеров вершин и весов ребер в таблице будем сразу же проверять правильность ввода и при ошибке сигнализировать пользователю. Чтобы не надоедать частыми сигналами об ошибках, будем выдавать их не в диалоговом окне, а в специальной строке статуса, которую необходимо разместить внизу под таблицей.

Замечание

Не надоедайте пользователю частыми сообщениями об ошибках. Если они могут возникать в процессе ввода символов, лучше выводите ошибки в строку статуса. Если ошибки могут возникать регулярно в цикле, то лучше выдать одно сообщение по завершении цикла.

Для удобства пользователя сделаем так, чтобы при вводе чисел в таблицу ее размер автоматически увеличивался или уменьшался. Для этого просто после каждого изменения пользователем значений в ячейках таблицы необходимо найти максимальный номер заполненной строки и общее количество доступных для ввода строк таблицы сделать на единицу больше.

Для просмотра характеристик графа и выполнения поиска в нем необходимо сформировать матрицу смежности. Для удобства пользователя свяжем этот процесс с моментом переключения с первой страницы на другие. Для этого необходимо алгоритм формирования матрицы смежности выполнить в ответ на событие `OnChange` компонента `TPageControl`. Кроме того, если пользователь переключится на страницу «Характеристики», то необходимо выдать на ней краткую информацию о количестве вершин и ребер в графе, а также выдать в таблице сформированную матрицу смежности.

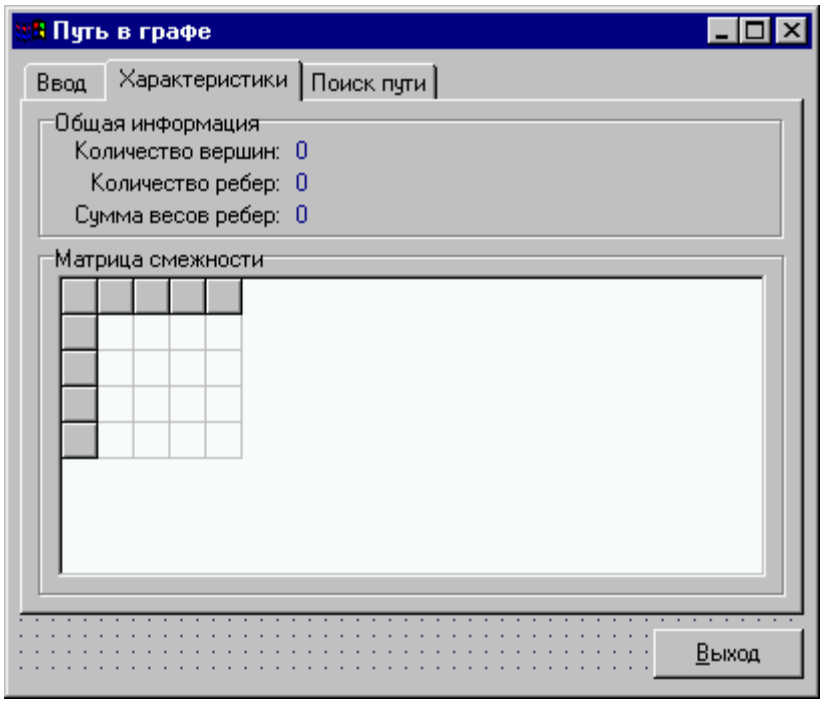

Рис. 14. Страница для вывода характеристик графа и матрицы смежности

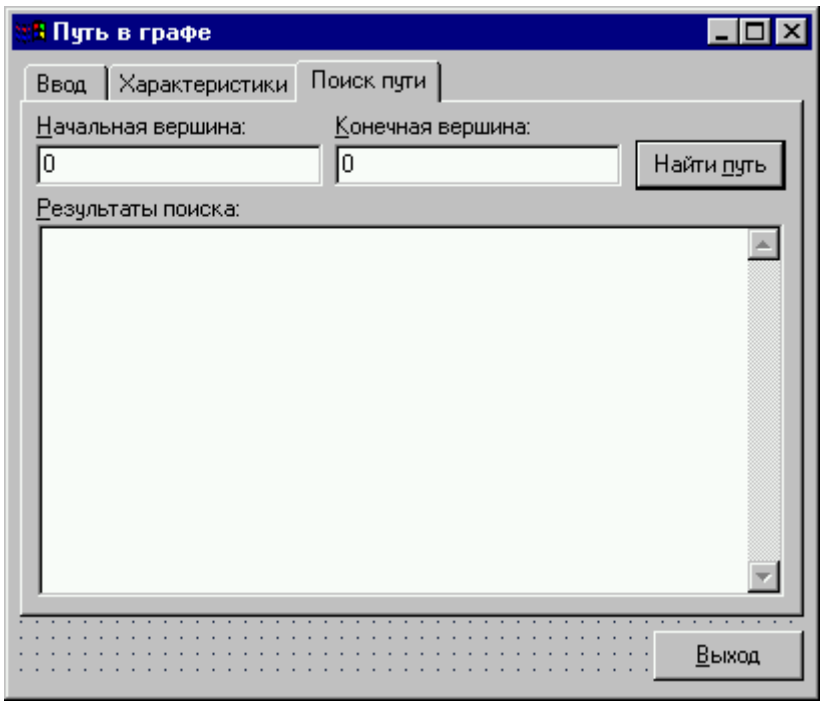

Рис. 15. Страница для поиска в графе

Третья страница «Поиск пути» должна содержать две строки ввода для задания номеров начальной и конечной вершин, кнопки для начала поиска пути в графе по алгоритму Дейкстры и большого текстового поля типа `TMemo` для выдачи результатов поиска.

В листингах 9-10 приведены тексты файлов проекта и модуля.

Листинг 9. Текст файла проекта Lab4.dpr

```
program Lab5;

uses
  Forms,
  Graphs in 'Graphs.pas' {GraphForm};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TGraphForm, GraphForm);
  Application.Run;
end.
```

Листинг 10. Текст главного модуля Graphs.pas

```
unit Graphs;

interface

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

type
  TGraphForm = class(TForm)
    PageControl: TPageControl;
    TabSheetInput: TTabSheet;
    TabSheetInfo: TTabSheet;
    TabSheetSearch: TTabSheet;
    StringGridRibs: TStringGrid;
    ButtonInsert: TButton;
    ButtonDelete: TButton;
    ButtonDeleteAll: TButton;
    ButtonExit: TButton;
    GroupBoxIncidency: TGroupBox;
    GroupBoxInfo: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    LabelVertexCount: TLabel;
    LabelRibCount: TLabel;
    LabelWeightSum: TLabel;
    StringGridAdjacency: TStringGrid;
    Label7: TLabel;
    EditStart: TEdit;
    EditFinish: TEdit;
    Label8: TLabel;
    ButtonFind: TButton;
    Label9: TLabel;
    MemoResults: TMemo;
    StatusBar: TStatusBar;
    CheckBoxOrientated: TCheckBox;
    procedure ButtonExitClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure StringGridRibsSetEditText(Sender: TObject; ACol,
      ARow: Integer; const Value: String);
    procedure ButtonInsertClick(Sender: TObject);
    procedure ButtonDeleteClick(Sender: TObject);
    procedure ButtonDeleteAllClick(Sender: TObject);
```

```
    procedure PageControlChange(Sender: TObject);
    procedure ButtonFindClick(Sender: TObject);
  private
    S: integer; // Количество вершин в матрице смежности
    W: array of array of double; // Матрица смежности (веса)
    procedure CheckRows; // Установить оптимальное количество ребер в таблице ребер
  end;

var
  GraphForm: TGraphForm;

implementation

{$R *.DFM}

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

procedure TGraphForm.FormCreate(Sender: TObject);
begin // При создании формы надо проинициализировать таблицу
  with StringGridRibs do
  begin // Формирование заголовков столбцов таблицы ребер
    Cells[0,0]:='N';
    Cells[1,0]:='Начало';
    Cells[2,0]:='Конец';
    Cells[3,0]:='Вес';
    Cells[0,1]:='1';
  end;
end;

// Установить оптимальное количество ребер в таблице ребер
procedure TGraphForm.CheckRows;
var i, maxRow: integer;
begin
  with StringGridRibs do
  begin
    MaxRow:=0; // Вычисляем максимальный номер незаполненной строки
    for i:=RowCount-1 downto 1 do
      if (Cells[1,i]<>'') or (Cells[2,i]<>'') or (Cells[3,i]<>'') then
      begin
        MaxRow:=i;
        break;
      end;
    RowCount:=MaxRow+2;
    for i:=1 to RowCount-1 do
      Cells[0, i]:=IntToStr(i); // Установить номера строк в первой колонке таблицы
  end;
end;

procedure TGraphForm.StringGridRibsSetEditText(Sender: TObject; ACol,
  ARow: Integer; const Value: String);
begin // В ячейке таблицы изменен текст
  try // Проверка введенного текста
    if Value<>'' then
      if ACol=3 then
        StrToFloat(Value) // В колонке №3 допустимы только вещественные значения
      else
        StrToInt(Value); // В колонке №3 допустимы только целочисленные значения
    StatusBar.SimpleText:='';
```

```
  except
    on E: Exception do // Текст ошибки надо вывести в строку статуса
      StatusBar.SimpleText:=E.Message;
  end;
  CheckRows; // Перевычислить количество строк в таблице
end;

procedure TGraphForm.ButtonInsertClick(Sender: TObject);
var i: integer;
begin // Нажата кнопка "Вставить ребро"
  with StringGridRibs do
  begin
    // Раздвигаем строчки
    for i:=RowCount-1 downto Row+1 do
    begin
      Cells[1,i]:=Cells[1,i-1];
      Cells[2,i]:=Cells[2,i-1];
      Cells[3,i]:=Cells[3,i-1];
    end;
    // Вставляем пустую
    Cells[1,Row]:='';
    Cells[2,Row]:='';
    Cells[3,Row]:='';
  end;
  CheckRows; // Перевычислить количество строк в таблице
end;

procedure TGraphForm.ButtonDeleteClick(Sender: TObject);
var i: integer;
begin // Нажата кнопка "Удалить ребро"
  with StringGridRibs do
    // Удаляем текущую строчку и остальные сдвигаем
    for i:=Row to RowCount-2 do
    begin
      Cells[1,i]:=Cells[1,i+1];
      Cells[2,i]:=Cells[2,i+1];
      Cells[3,i]:=Cells[3,i+1];
    end;
  CheckRows; // Перевычислить количество строк в таблице
end;

procedure TGraphForm.ButtonDeleteAllClick(Sender: TObject);
var i: integer;
begin // Нажата кнопка "Удалить все"
  with StringGridRibs do
    for i:=1 to RowCount-1 do
    begin // Очищаем всю таблицу
      Cells[1,i]:='';
      Cells[2,i]:='';
      Cells[3,i]:='';
    end;
  CheckRows; // Перевычислить количество строк в таблице
end;

procedure TGraphForm.PageControlChange(Sender: TObject);
var min, max, i, j, r: integer; wsum, ww: double;
begin // Пользователь переключился на другую закладку
  if PageControl.ActivePage<>TabSheetInput then
    with StringGridRibs do
    try // Вычислить максимальный номер вершины
      min:=MaxInt; max:=-MaxInt;
```

```
      for i:=1 to RowCount-2 do
        for j:=1 to 2 do
        begin
          r:=StrToInt(Cells[j,i]);
          if r<min then min:=r;
          if r>max then max:=r;
        end;
      if (min<0) or (max>=100) or (max<0) then
        Abort; // Если номера вершин слишком велики, то считаем, что это ошибка
      // Формируем матрицу смежности
      wsum:=0; S:=max+1; SetLength(W,0,0); SetLength(W,S,S);
      for i:=1 to RowCount-2 do
      begin
        ww:=StrToFloat(Cells[3,i]);
        if ww<=0 then
          Abort;
        W[StrToInt(Cells[1,i]), StrToInt(Cells[2,i])]:=ww;
        if not CheckBoxOrientated.Checked then // Если граф неориентированный, то делаем ребро
          W[StrToInt(Cells[2,i]), StrToInt(Cells[1,i])]:=ww; // в обратную сторону
        wsum:=wsum+ww; // Суммируем все веса рёбер
      end;
    except // Если матрица введена неверно, то считаем, что матрица имеет нулевые размеры
      wsum:=0; S:=0; SetLength(W,0,0);
    end;

  if PageControl.ActivePage=TabSheetInfo then
  begin // Выдать информацию на странице "Характеристики"
    LabelVertexCount.Caption:=IntToStr(S);
    if S=0 then
      LabelRibCount.Caption:='0'
    else
      LabelRibCount.Caption:=IntToStr(StringGridRibs.RowCount-2);
    LabelWeightSum.Caption:=FloatToStr(wsum);
    with StringGridAdjacency do
      if S=0 then
      begin // Граф не задан
        ColCount:=2; RowCount:=2;
        Cells[1,0]:=''; Cells[0,1]:=''; Cells[1,1]:='';
      end
      else
      begin // Граф введен правильно, выводим матрицу смежности
        ColCount:=S+1; RowCount:=S+1;
        for i:=0 to S-1 do
        begin // Выводим номера строк и столбцов
          Cells[i+1,0]:=IntToStr(i);
          Cells[0,i+1]:=IntToStr(i);
        end;
        for i:=1 to S do
          for j:=1 to S do // Выводим матрицу смежности
            Cells[i,j]:=FloatToStr(W[j-1,i-1]);
      end;
  end;
end;

procedure TGraphForm.ButtonFindClick(Sender: TObject);
const MaxDouble = 10E100;
var VStart, VFinish, i, j: integer; V: array of double; F: boolean; ww: double;
begin // Нажата кнопка "Найти путь"
  MemoResults.Lines.Clear;
  if S=0 then
  begin
```

```
    MemoResults.Lines.Add('Матрица не задана. Поиск невозможен.');
    exit;
  end;
  try // Получаем номер начальной вершины
    VStart:=StrToInt(EditStart.Text);
    if (VStart<0) or (VStart>=S) then
      Abort;
  except
    MemoResults.Lines.Add('Неверный номер исходной вершины.');
    exit;
  end;
  try // Получаем номер конечной вершины
    VFinish:=StrToInt(EditFinish.Text);
    if (VFinish<0) or (VFinish>=S) then
      Abort;
  except
    MemoResults.Lines.Add('Неверный номер конечной вершины.');
    exit;
  end;
  if VStart=VFinish then
  begin
    MemoResults.Lines.Add('Начальная и конечная вершины совпадают.');
    exit;
  end;
  SetLength(V, S);
  for i:=0 to S-1 do
    V[i]:=MaxDouble;
  V[VFinish]:=0; // Поиск будем вести с конца в начало
  repeat
    F:=True;
    for i:=0 to S-1 do
      if V[i]<MaxDouble/2 then
        for j:=0 to S-1 do
          if (i<>j) and (W[j,i]>0) and (V[i]+W[j,i]<V[j]) then
            begin
              V[j]:=V[i]+W[j,i];
              F:=False;
            end;
  until F; // Выполняем цикл, пока возможно улучшение пути
  if V[VStart]>MaxDouble/2 then
    MemoResults.Lines.Add('Путь не существует.')
  else
  begin
    MemoResults.Lines.Add('Кратчайший путь (длина '+FloatToStr(V[VStart])+'):');
    while VFinish<>VStart do
    begin
      i:=0;
      for j:=0 to S-1 do
        if (j<>VStart) and (W[VStart,j]>0) and (V[VStart]=V[j]+W[VStart,j]) then
        begin
          i:=j;
          break;
        end;
      MemoResults.Lines.Add('Ребро '+IntToStr(VStart)+'-'+IntToStr(i));
      VStart:=i;
    end;
  end;
end;

end.
```

Вопросы и задания для самостоятельной работы

1. На странице «Ввод» добавьте кнопку «Авто» для автоматического формирования случайного графа.
2. Сделайте возможность задания кратных ребер. При этом в матрицу смежности должен быть занесен минимальный вес из всех введенных ребер.
3. Перед выполнением поиска выдайте в окно результатов исходный набор ребер.
4. На странице «Поиск пути» добавьте кнопку «Сохранить» для сохранения результатов поиска в файл.
5. Внизу всей формы поместите кнопку «О программе» для выдачи краткой информации о программе.

Учебное издание

Алексей Владимирович Скворцов
Поддубная Тамара Николаевна

Работа с графами

(методические указания к
лабораторной работе № 4)

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