1. **Перечислите и опишите основные свойства объектов.**

Свойства отвечают за внешний вид или поведение объектов в программе во время выполнения.

Каждый компонент, помещенный на форму, отражается в инспекторе объектов. Каждый объект имеет свои собственные свойства. Один объект может обладать свойством, а другой-нет. Например, поле ввода (TEdit) имеет свойство ReadOnly, которое отвечает за изменение текста в этом поле. Логично, что кнопка (TButton) не имеет и не может иметь этого свойства.

Некоторые простые взаимодействия уже"встроены"в стандартные компоненты и могут быть легко изменены с помощью инспектора объектов. Все остальное нужно создавать и программировать вручную. Основной способ создания взаимодействий компонентов-это почти написание реакций на события.

Тип свойств

Свойство (атрибут) -это поле в структуре, содержащее определенное значение. Начните с того, как эти значения задаются и какие именно "свойства", т. е. структура имеет возможность идентифицировать несколько типов свойств.

1. Простое свойство. Простое свойство-это свойство со значением числа или строки (текст). Примерами таких свойств являются Left и Top формы. Эти свойства определяют положение формы на экране (особенно ее левый верхний угол). Значения этих свойств являются числами. Пример свойства, имеющего значение-строка-Caption формы. Это свойство сохраняет заголовок формы и имеет значение обычный текст.

2. Перечисляемые свойства. Эти свойства не могут быть заданы явно, но могут быть выбраны только из списка. Список возможных значений заранее определен. Примером такого свойства является свойство AutoSize формы. Он отвечает за то, будет ли форма автоматически изменять свой размер, чтобы показать все компоненты, размещенные на ней. Значение является истинным или ложным. Другой пример-свойство BorderStyle. Это свойство отвечает за внешний вид формы и поведение ее границ, то есть можно ли изменять размер формы во время работы программы. Это свойство принимает одно из шести значений.

3. Вложенные свойства. В этих свойствах есть несколько других свойств. В инспекторе объектов слева от этих имен свойств отображается маленькая кнопка со знаком"+". Щелчок по нему откроет это свойство (символ изменится на" -"). Снова щелкните свойство свернуть. Существует два основных типа вложенных свойств: коллекции и комбинированные значения. Набор-это набор значений, каждое из которых "включено" или "выключено".

Комбинированное значение-это набор нескольких атрибутов, которые могут иметь различные типы данных. Примером набора является свойство BorderIcons формы-оно отвечает за кнопки, которые будут отображаться в строке заголовка окна. Понятно, что либо кнопка может отображаться на экране, либо нет-этот набор очень удобно настроить с помощью набора. Примером комбинированного значения является свойство Font (которым обладает большинство визуальных компонентов), задающее шрифт элемента. Он включает в себя несколько других свойств-имя шрифта, цвет,стиль,размер и т. д.

**Управление свойствами**

Вы можете управлять свойствами двумя способами, изменяя их значения: в режиме программирования(во время разработки) и во время выполнения (во время выполнения). Вы все еще можете изменять свойства в том же инспекторе объектов. Если это простое свойство, просто нажмите на строку имени свойства и введите новое значение. Если это перечисляемое свойство, вы можете выбрать его значение из списка. Некоторые свойства(например, left, top, width и height) можно изменить простым перетаскиванием и изменением размера с помощью мыши.

Следует отметить, что в режиме конструктора все свойства выбранного компонента могут быть недоступны через инспектор объектов. Эти свойства доступны в режиме выполнения и изменяются программно.

1. **Дайте описание компонентов для установки связи с базой данных. Алгоритм установки связи в базе данных.**

Большинство визуальных компонентов для работы с данными похожи на соответствующие компоненты для обычных приложений. Основное отличие заключается в том, что компоненты для работы с базами данных содержат свойства , позволяющие указать источник данных (DataSource) и поле, из которого компонент получает данные (DataField).

Названия компонентов начинаются с букв DB. Большинство компонентов предназначено для отображения текущего значения одного поля: DBEdit, DBCheckBox, DBRadioGroup, DBText и др. Для вывода и редактирования данных поля Мемо используются компоненты DBMemo и DBRichEdit. Для просмотра изображений предназначен компонент DBImage. Отображать данные графически позволяет компонент DBChart.

Кроме того, имеются компоненты, предназначенные для использования только в приложениях баз данных. Это DBNavigator и компоненты DBLookupComboBox, DBLookupListBox для синхронного просмотра данных из связанных таблиц.

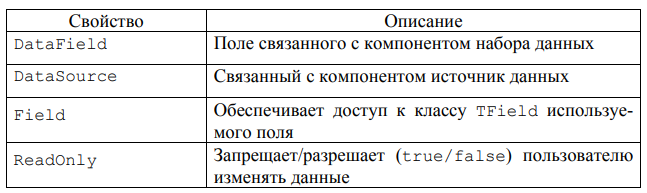


Рис. 1 – Описание основных свойств компанента

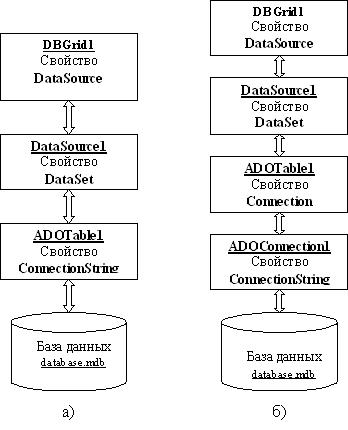
[](https://www.bestprog.net/wp-content/uploads/2016/01/01_02_00_008_04r.jpg)

Рис. 2 - Способы подключения к файлу базы данных

Для создания приложения необходимо выполнить приведённые ниже действия

1. Разместить на форме компоненты Table, DataSource, DBGrid.

2. Связать компонент Table с таблицей БД, которая находится на диске. Для этого в Инспекторе объектов:

* в свойстве DatabaseName выбрать в выпадающем списке псевдоним или указать путь к базе данных;
* в свойстве TableName указать название таблицы;
* свойство Active установить в значение true.

3. Связать компоненты DataSource и Table: в свойстве DataSet компонента DataSource установить значение Table1.

4. Подсоединить DBGrid к источнику данных: в свойстве DataSource компонента DBGrid задать значение DataSource1.

Для удобства работы можно добавить на форму навигатор DBNavigator и связать его с источником данных: в свойстве DataSource задать значение DataSource1.

1. **Опишите алгоритм нахождения потока минимальной стоимости.**

***Шаг 0.*** Определяем для данной сети начальное базисное допустимое решение (множество дуг). Переходим к шагу 1.

***Шаг 1.*** С помощью условия оптимальности симплекс-метода определяем вводимую в базис переменную (дугу). Если на основе условия оптимальности определяем, что последнее решение оптимально, вычисления заканчиваются. В противном случае переходим к шагу 2.

***Шаг 2.*** На основе условия допустимости симплекс-метода определяем исключаемую из базиса переменную (дугу). Изменив базис, возвращаемся к шагу 1.

Сети с **n** узлами и нулевым результирующим потоком (т.е. при выполнении равенства  ) соответствуют **n - 1** ***независимым*** ограничениям в виде равенств. Поэтому базисное решение должно содержать  переменных. Можно показать, что базисное решение соответствует ***остовному дереву*** данной сети.

Вводимая переменная (дуга) на шаге 1 определяется путем вычисления разностей для всех текущих небазисных дуг **(i, j)**. Если для всех разностей , тогда текущее базисное решение оптимально. Иначе в качестве вводимой в базис переменной выбираем дугу, которой соответствует наибольшее положительное значение разности .

Вычисление разностей  основано на соотношениях двойственности, точно так же как в транспортной модели. Обозначим через **wi** переменную задачи, двойственной к задаче линейного программирования, которая (переменная) соответствует ограничению узла **i**. Тогда данная двойственная задача формулируется следующим образом.

Максимизировать

при выполнении условий

переменные **wi** произвольного знака ().

Из теории линейного программирования следует, что  для любой базисной дуги **(i, j)**. Поскольку исходная задача линейного программирования по определению имеет одно избыточное ограничение, мы можем присвоить произвольное значение одной из переменной двойственной задачи. Для определенности положим **w1 = 0**. Затем следует решить (базисную) систему уравнений  для нахождения остальных переменных двойственной задачи. Далее вычисляем разности  для небазисных переменных согласно формуле

1. **Введите три числа. В зависимости от выбранного переключателя RadioButton вычислить объем или площадь поверхности параллелепипеда с такими сторонами.**

В приложении Delphi 7 создаем новый проект Application. Из раздела компонентов переносим нужные нам в нужном количестве, а именно: Button, Edit, Label, RadioButton, MainMenu. Расставляем их.

Получилась следующая форма

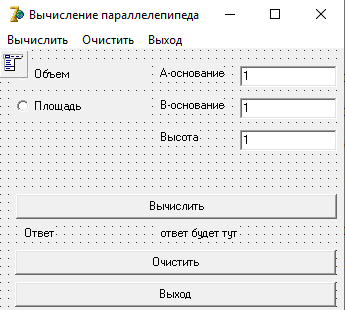


Рис.3 – форма приложения

Далее переходим к написанию кода. Для написания событий щелчков мыши по кнопкам, два раза нажимаем на нужную кнопку и пишем код. Пример кода, отвечающий за вычисление отражен ниже.

Листинг 1 – вычисление объема и площади

procedure TForm1.Button1Click(Sender: TObject);

begin

if(Edit1.GetTextLen>0) AND (Edit2.GetTextLen>0) AND (Edit3.GetTextLen>0) then

begin

A:=StrToFloat(Edit1.Text);

B:=StrToFloat(Edit2.Text);

H:=StrToFloat(Edit3.Text);

//V

if(RadioButton1.Checked) then

begin

Answer:=A\*B\*H;

end;

//S

if(RadioButton2.Checked) then

begin

Answer:=2\*(A\*B+B\*H+H\*A);

end;

Label5.Caption:=FloatToStr(Answer);

end;

end;

1. **Введите два положительных числа – координаты Х и У. При нажатии на кнопку она должна принимать положение в соответствии с введенными координатами. Добавить защиту от некорректных чисел и меню дублирующее кнопки.**

В приложении Delphi 7 создаем новый проект Application. Из раздела компонентов переносим нужные нам в нужном количестве, а именно: Button, Edit, Label, MainMenu. Расставляем их.

Получилась следующая форма

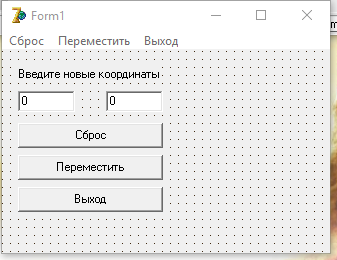


Рис.4 – форма приложения

Далее переходим к написанию кода. Для написания событий щелчков мыши по кнопкам, два раза нажимаем на нужную кнопку и пишем код. Пример кода, отвечающий за перемещение кнопки, отражен ниже.

Листинг 2 – перемещение кнопки

procedure TForm1.Button2Click(Sender: TObject);

begin

if(Edit1.GetTextLen>0) AND (Edit2.GetTextLen>0) then

begin

newX:=StrToInt(Edit1.Text);

newY:=StrToInt(Edit2.Text);

Relocate();

end;

end;

procedure TForm1.Relocate();

begin

Button2.Top:=newY;

Button2.Left:=newX;

end;

1. **Напишите программу, определяющую среднее количество ребер в графе, заданном матрицей смежности вершин? Указание. Среднее арифметическое элементов матрицы смежности.**

В приложении Delphi 7 создаем новый проект Application. Из раздела компонентов переносим нужные нам в нужном количестве, а именно: Button, Edit, Label, MainMenu, OpenDialog, String grid. Расставляем их.

Получилась следующая форма

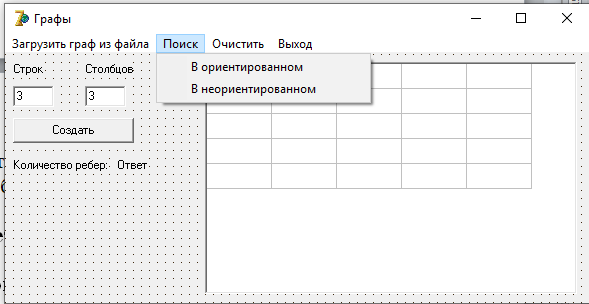


Рис.4 – форма приложения

Далее переходим к написанию кода. Для написания событий щелчков мыши по кнопкам, два раза нажимаем на нужную кнопку и пишем код.

Для чтения матрицы смежности из файла, текстовый файл должен следовать следующим правилам:

1. Каждое число должно идти с новой строки
2. Первая строка файла является количеством строк матрицы
3. Вторая строка файла является количеством столбцов матрицы
4. Все последующие строки должны представлять значения матрицы слева-направо сверху-вниз без пробелов, каждое с новой строки.

Пример кода, отвечающий за чтение из фала, отражен ниже.

Листинг 3 – чтение из файла

procedure TForm1.N1Click(Sender: TObject);

var

f:textfile;

temp, x, y: integer;

tempstr: string;

begin

If OpenDialog1.Execute then

begin

assignfile(f, OpenDialog1.FileName);

reset(f);

readln(f, temp);

stringgrid1.colcount := temp;

readln(f, temp);

stringgrid1.rowcount := temp;

for X := 0 to stringgrid1.colcount - 1 do

for y := 0 to stringgrid1.rowcount - 1 do

begin

readln(F, tempstr);

stringgrid1.cells[x, y] := tempstr;

end;

closefile(f);

end;

end;

Пример кода, отвечающего за поиск количество ребер графа в ориентированном графе представлен ниже

Листинг 4 – поиск количества ребер в ориентированном графе

procedure TForm1.N6Click(Sender: TObject);

var

countOrientReb,i,j,temp:Integer;

ArrayLengthRow, ArrayLengthCol: Integer;

myArr:Array of Array of Integer;

begin

countOrientReb:=0;

getArrayFromTable();

ArrayLengthRow := Length(arrMatrix);

ArrayLengthCol := Length(arrMatrix[0]);

for i:=0 to ArrayLengthRow do

for j:=0 to ArrayLengthCol do

begin

temp:=0;

if(Length(stringgrid1.cells[i, j])>0) then

temp:=strToInt(stringgrid1.cells[i, j]);

countOrientReb:=countOrientReb+temp;

end;

Label4.Caption:=IntToStr(countOrientReb);

end;

**Список использованных источников**

1. Работа с БД [Электронный ресурс]. – Режим доступа: https://www.delphiplus.org/programirovanie-baz-dannih-v-delphi/praktika-raboti-s-bd-ms-access-iz-delphi.html. – Дата доступа: 09.10.2020.
2. Работа с данными [Электронный ресурс]. – Режим доступа: http://www.snkey.net/books/delphi/ch4-4.html. – Дата доступа: 09.10.2020.
3. Свойства объектов [Электронный ресурс]. – Режим доступа: https://life-prog.ru/view\_zam.php?id=199&cat=12&page=1. – Дата доступа: 09.10.2020.
4. Массивы [Электронный ресурс]. – Режим доступа: https://www.bestprog.net/en/2016/09/29/arrays/. – Дата доступа: 09.10.2020.
5. Компонент StringGrid [Электронный ресурс]. – Режим доступа: http://www.delphi-manual.ru/stringgrid.php?com=yes. – Дата доступа: 09.10.2020.
6. Очистка таблиц [Электронный ресурс]. – Режим доступа: http://www.stringgrid-delphi.ru/cleartable.php. – Дата доступа: 09.10.2020.
7. Работа с функцией Mod [Электронный ресурс]. – Режим доступа: https://delphisources.ru/pages/faq/faq\_delphi\_basics/Mod.php.html. – Дата доступа: 09.10.2020.