

Научно-практическая конференция учащихся Республики Тыва

«Шаг в будущее»

11-12 марта 2016 года, г. Кызыл

Разработка игры-программы
«Угадыватель мыслей»
в среде Delphi 7 Personal
с использованием
динамически создаваемых объектов

Работу выполнила:

Хепчи Татьяна Сергеевна

ученица 11 класса

ГБОУ Аграрный лицей Республики Тыва

Научный руководитель:

Куулар Чойгана Радийевна

Учитель информатики 1 категории

ГБОУ Аграрный лицей Республики Тыва

г. Кызыл – 2016

Разработка игры-программы
«Угадыватель мыслей»
в среде Delphi 7 Personal
с использованием динамически создаваемых объектов

Хепчи Татьяна Сергеевна
ученица 11 класса ГБОУ Аграрный лицей Республики Тыва

АННОТАЦИЯ

Объектом данной работы является объектно-ориентированная среда программирования Delphi 7 Personal.

Предметом работы технология программирования на Delphi 7 Personal.

Цель данной работы это раскрыть объектно-ориентированное программирование на примере Delphi 7 Personal.

Задачи, решение которых необходимо для достижения поставленной цели:

1. Проанализировать основные составляющие Delphi 7 Personal.
2. Собрать необходимую информацию.
3. Изучение принципов объектно-ориентированного программирования, в котором основными концепциями являются понятия классов и объектов.
4. Разработать игру-программу в среде Delphi 7 Personal.

Актуальность данной работы заключается в том, что ООП помогает решить множество классических задач. Но для задач нетривиальных, всегда необходимо пробовать и применять новые подходы. Полученные знания помогут программистам для написания более качественного и гибкого кода.

Методы и приемы: В идее программы заложен математический фокус, основанный на том, что, если от любого числа (в нашем случае от 0 до 99) вычесть его составляющие цифры, то всегда получим число, делящееся на 9. Для чисел делящихся на 9 используется одинаковый, заданный случайным образом, символ или знак, а для всех остальных разные случайные символы или значки.

Полученные данные: Рабочее приложение magic.exe работающее в среде Windows x86/x64 не отличающееся программными требованиями от каких либо приложений созданных в Delphi. Для использования подойдет Pentium 2 или выше, Windows XP или выше, к видеопамяти особых требований нет. Разработанный пользовательский интерфейс позволит с лёгкостью пользоваться этим продуктом. Программа содержит минимальное меню, а с помощью кнопок главного окна приложения можно управлять всей программой.

Вывод: С помощью среды объектно-ориентированного программирования Delphi 7 Personal можно создавать вполне работоспособные учебные программы, в которых можно использовать динамически создаваемые объекты, в нашем случае надписи.

Разработка игры-программы
«Угадыватель мыслей»
в среде Delphi 7 Personal
с использованием динамически создаваемых объектов

Хепчи Татьяна Сергеевна
ученица 11 класса ГБОУ Аграрный лицей Республики Тыва

ПЛАН ИССЛЕДОВАНИЙ

1. Проанализировать основные составляющие Delphi 7 Personal.
2. Собрать необходимую информацию.
3. Изучение принципов объектно-ориентированного программирования, в котором основными концепциями являются понятия классов и объектов.
4. Продумать интерфейс будущей программы.
5. Продумать какие компоненты целесообразнее использовать
6. Разработать игру-программу в среде Delphi 7 Personal.

Разработка игры-программы «Угадыватель мыслей» в среде Delphi 7 Personal с использованием динамически создаваемых объектов

Программы в объектно-ориентированном программировании (ООП) состоят из двух частей (описание и сама программа). Любая программа может быть концептуально организована либо вокруг её кода кодовое воздействие на данные, либо вокруг данных управляемый данными доступ к коду.

При первом процедурном подходе программу определяет последовательность операторов её кода. Второй подход организует программу вокруг данных, т.е. вокруг объектов и набора хорошо организованных интерфейсов.

ООП - это методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса; а классы образуют иерархию наследования.

- ООП использует в качестве базовых элементов объекты, а не алгоритмы.
- Каждый объект будет экземпляром какого-либо определенного класса.
- Классы реализованы (организованы) иерархически.

Основное достоинство ООП - сокращение числа межмодульных связей, изменение объемов информации, которая передается между модулями и возможность повторного использования кодов.

Программирование — это процесс создания (разработки) программы, который может быть представлен последовательностью следующих шагов:

1. Спецификация (определение, формулирование требований к программе).
2. Разработка алгоритма.
3. Кодирование (запись алгоритма на языке программирования).
4. Отладка.
5. Тестирование.
6. Создание справочной системы.

Спецификация, определение требований к программе – один из важнейших этапов, на котором подробно описывается исходная информация, формулируются требования к результату, поведение программы в особых случаях (например, при вводе неверных данных), разрабатываются диалоговые окна, обеспечивающие взаимодействие пользователя и программы.

На этапе разработки алгоритма необходимо определить последовательность действий, которые надо выполнить для получения результата. Если задача может быть решена несколькими способами и, следовательно, возможны различные варианты алгоритма решения, то программист, используя некоторый критерий, например, скорость решения алгоритма, выбирает наиболее подходящее решение. Результатом этапа разработки алгоритма является подробное словесное описание алгоритма или его блок-схема.

После того как определены требования к программе и составлен алгоритм решения, алгоритм записывается на выбранном языке программирования. В результате получается исходная программа.

Отладка — это процесс поиска и устранения ошибок. Ошибки в программе разделяют на две группы: синтаксические (ошибки в тексте) и алгоритмические. Синтаксические ошибки — наиболее легко устранимые. Алгоритмические ошибки обнаружить труднее. Этап отладки можно считать законченным, если программа правильно работает на одном-двух наборах входных данных.

Этап тестирования особенно важен, если вы предполагаете, что вашей программой будут пользоваться другие. На этом этапе следует проверить, как ведет себя программа на как можно большем количестве входных наборов данных, в том числе и на заведомо неверных.

Если разработчик предполагает, что программой будут пользоваться другие, то он обязательно должен создать справочную систему и обеспечить пользователю удобный доступ к справочной информации во время работы с программой. В современных программах справочная информация представляется в форме CHM - или HLP - файлов. Помимо справочной информации, доступ к которой осуществляется из программы во время ее работы, в состав справочной системы включают инструкцию по установке (инсталляции) программы, которую оформляют в виде Readme-файла в одном из форматов: TXT, DOC или HTM.

На первом этапе создания программы программист должен определить последовательность действий, которые необходимо выполнить, чтобы решить поставленную задачу, т. е. разработать алгоритм. Алгоритм — это точное предписание, определяющее процесс перехода от исходных данных к результату.

Мы задумали, что интерфейс нашей программы будет содержать окно, где располагаются 100 символов или значков, рядом с каждым символом будет расположена надпись содержащая его порядковый номер (номера мы задумали от 0 до 99).

0	⚡	1	📁	2	📁	3	⌚	4	📁	5	🔍	6	⌚	7	📁	8	📄	9	⚡
10	↶	11	⚡	12	←	13	←	14	↶	15	⬇	16	←	17	↶	18	⚡	19	↷
20	📄	21	📁	22	🕒	23	🕒	24	📄	25	🗑	26	📄	27	⚡	28	📄	29	📄
30	🔍	31	📁	32	📄	33	📄	34	📁	35	📄	36	⚡	37	📄	38	📁	39	📁
40	—	41	📄	42	▶▶	43	—	44	📄	45	⚡	46	⏮	47	⏮	48	📄	49	▶▶
50	⬆	51	↷	52	↷	53	→	54	⚡	55	↶	56	↶	57	↶	58	→	59	↷
60	📄	61	📄	62	📄	63	⚡	64	📄	65	📄	66	📄	67	📄	68	📄	69	📄
70	⌚	71	📄	72	⚡	73	📁	74	📁	75	🔍	76	📁	77	🔍	78	📄	79	📄
80	▲	81	⚡	82	▶	83	▲	84	📄	85	—	86	▶▶	87	⏮	88	▼	89	◀
90	⚡	91	—	92	⏮	93	▲	94	📄	95	📄	96	▶	97	📄	98	⏮	99	⚡

Пользователь должен задумать число от 1 до 109, далее вычесть его составляющие цифры, отыскать полученное число в таблице и запомнить соответствующий символ, нажать на кнопку, и он должен увидеть запомненный символ. Например, игрок задумал число 17, тогда $17 - 1 - 7 = 9$. На самом деле фокус заключается в том, что какое бы число ни задумали, после вычитания составляющих его цифр всегда обязательно получим число, делящееся на 9.

Тогда, если для чисел делящихся на 9 использовать одинаковый, заданный случайным образом, символ или знак, а для всех остальных разные случайные символы или значки, то можно достичь эффекта магии.

После такого небольшого вступления, приступим к описанию создания программы.

Итак, для нашего проекта мы создали основную форму на ней разместили одну надпись, две кнопки, компонент меню, две панели, на вторую панель размещаем кнопку и компонент многостраничного текста memo1. Основное окно программы, то есть Form1 имеет размеры 940x460 пунктов. Панели создаем почти одинакового размера и размещаем одна на другую. На первой панели нет элементов и ее размещаем сзади а во второй панели расположенной впереди размещаем многостраничный текст правил игры и кнопку которая закрывает панель. Панели занимают основную часть окна программы и оставляют маленькую полоску внизу для двух кнопок, расположенных по обеим краям внизу окна. Первая кнопка у нас будет кнопка “Обновить” а вторая кнопка “Проверить/Посмотреть”.

На первой панели будут располагаться 200 элементов типа надпись. Для того чтобы к ним можно было обращаться по индексу и одновременно изменять их значение располагать их будем программным способом. В разделе объявления переменных “Var”

объявляем два массива, следующим способом: sannar: array [0..99] of TLabel; и nadpis: array [0..99] of TLabel;

Там же объявляем пять целочисленных переменных: i, j, k, numerfont, answer; и одну строковую переменную: namefont; Переменная i нужна для индекса массивов и начинается от 0 до 99, переменная $j = i \text{ div } 10$ (целая часть от деления на 10) для обозначения строк таблицы, а переменная $k = i \text{ mod } 10$ (остаток от деления на 10) для обозначения столбцов таблицы.

В операционной системе Windows имеется четыре встроенных символьных шрифта: Webdings, Wingdings, Wingdings 2, Wingdings 3. Переменная numerfont нужна для присвоения случайного шрифта из этих четырех шрифтов. Переменная answer нужна для обозначения одинакового символа для чисел, делящихся на 9. А строковая переменная namefont нужна для обозначения одинакового шрифта для чисел делящихся на 9.

В процедуре создания формы procedure TForm1.FormCreate объявляем заголовок окна: form1.Caption:='Угадыватель мыслей'; и для объявляем надписи для кнопок: Button1.Caption:='Обновить'; Button2.Caption:='Проверить';.

С помощью оператора повторения For формируем надписи с числами: используем команды: sannar[i]:=TLabel.Create(self) – создание надписей; sannar[i].Parent:=Panel1 – указываем объект-родитель где создаются надписи; ширину, высоту, размер, шрифт, выравнивание, стиль объявляем одинаковыми. Для расположения надписи сверху от окна используем формулу: sannar[i].Top:=5+35*(j); а расположения надписи слева от окна используем формулу: sannar[i].Left:=5+90*k; Для присвоения надписи используем оператор перевода целого числа в строку: sannar[i].Caption:=inttostr(i); Для объявления фона надписей, чтобы цвет фона у надписей через столбец чередовался, используем оператор ветвления,:

```
if odd(k)
then
  sannar[i].Color:=rgb(240,240,120)
else
  sannar[i].Color:=rgb(120,240,240);
```

Таким же образом, с помощью оператора повторения с параметром For формируем надписи с символами. Для расположения надписи сверху от окна используем формулу: nadpis[i].Top:=5+35*(j); а расположения надписи слева от окна используем формулу nadpis[i].Left:=50+90*k;

Если один и тот же код используется более двух раз, целесообразнее использовать процедуры, то есть код пишется в отдельной процедуре и по мере надобности обращаются к этой процедуре по ее имени. Для присвоения надписи к символам используем отдельную процедуру procedure update1; и ее используем в процедуре создания формы и еще при каждом нажатии на кнопку “Обновить”.

В процедуре update1 сначала для того, чтобы каждый раз использовать разные случайные числа используем процедуру randomize. Далее с помощью функции random переменной numerfont присваивается случайное число в диапазоне от 0 до 80 включительно. Далее с помощью оператора ветвления для строковой переменной namefont присваивается имя из четырех возможных шрифтов, если случайное число от 0 до 19 присваивается шрифт Wingdings 3, если случайное число от 20 до 39 присваивается шрифт Webdings, если случайное число от 40 до 59 присваивается шрифт Wingdings 2, если случайное число от 60 до 80 присваивается шрифт Wingdings. Это используется для присвоения случайного из 4 шрифтов для нужных чисел 0, 9, 18, ... , 99.

Для установления отдельного шрифта для каждой из 10 строк используем оператор выбора, для надписей с символами 1-й и 5-й строки используем шрифт Wingdings 3, для надписей с символами 2-й и 6-й строки используем шрифт Wingdings 2, для надписей с символами 3-й, 7-й и 10-й строки используем шрифт 'Wingdings, для надписей с символами 4-й, 8-й и 9-й строки используем шрифт Webdings.

Для переменной answer используем случайное число от 0 до 9: answer:=random(10);

Для установления символов к надписям используем оператор ветвления, если порядковый номер надписи делится нацело на 9, для имени фона надписи используется строковая переменная `namefont` и для свойства надписи используется целочисленная переменная `answer`, предварительно переведенная в строковую переменную с помощью функции `inttostr`. А для всех остальных надписей, порядковый номер надписи не делящийся нацело на 9, для свойства надписи используется случайное число в диапазоне от 0 до 9, переведенная в строковую переменную с помощью функции `inttostr`.

Все описание процедуры `update1` на этом заканчивается, как было ранее сказано эти не хитрые действия используем в процедуре создания формы и еще при каждом нажатии на кнопку “Обновить”.

В процедуре нажатии на кнопку “Обновить” кроме процедуры `update1` используем две команды `label100.Font.Name:=namefont; label100.Caption:=inttostr(answer);`, то есть к надписи появляющейся при нажатии на кнопку “Проверить” присваиваем нужный шрифт и нужный символ.

В процедуре нажатии на кнопку “Проверить/Посмотреть” с помощью оператора ветвления то скрываем Первую Панель, то показываем. Вместе с этой панелью одновременно скрываются все надписи, так как их объект-родителем является эта панель. Ну и при нажатии на кнопку чередуется надпись на кнопке надпись будет то Проверить то Посмотреть.

Меню программы содержит две опции “Файл” и “Справка”. Опция “Файл” имеет подопцию “Выход”. А опция “Справка” имеет две подопции “Правила” и “О программе”.

При нажатии на опцию “Выход” срабатывает процедура выхода, использующая оператор ветвления и диалоговое окно `MessageDlg` с обработчиком нажатия на кнопки “Да” и “Нет”, с сообщением “Вы действительно хотите выйти из программы”.

При нажатии на опцию “Правила” появляется Панель с правилами а при нажатии на кнопку “ОК” на панели с правилами, панель с правилами скрывается с помощью свойства `Visible` панели.

При нажатии на опцию “О программе” с помощью команды `form2.showmodal` появляется новая форма “Form2” содержащая сведения об авторе программы, оформленная в виде отдельного модуля “About”.

Форма “О программе” содержит кнопку, для закрытия формы, рисунок для фотографии автора программы, компонент многостраничного текста `Memo1` для сведений о программе и надпись для имени программы. Модуль “О программе” содержит только одну процедуру, которая закрывает эту форму с помощью команды `close`.

Две модули программы сохранены в отдельном каталоге “`pas`” и содержат шесть файлов типа `dcu` – двоичный скомпилированный модуль, `dfm` – текстовый файл с описанием компонентов формы, `pas` – текстовый файл с исходным кодом, а файлы с фото автора и иконкой программы сохранены в каталоге “`files`”.

В основном каталоге сохраняются пять файлов: `magic.cfg` – конфигурационный файл, `magic.dof` – служебный файл, `magic.dpr` – файл проекта, `magic.res` – файл со встраиваемыми ресурсами приложения (например, иконками), `magic.exe` – скомпилированный исполнимый файл проекта, может работать автономно вне других файлов проекта.

Все описание создания программы мы закончили.

С помощью среды объектно-ориентированного программирования `Delphi 7 Personal` можно создавать вполне работоспособные учебные программы, в которых можно использовать динамически создаваемые объекты, в нашем случае надписи.

Ниже приведены листинги модулей программы.

ЛИСТИНГ ОСНОВНОГО МОДУЛЯ

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, WinTypes, WinProcs, Variants,
  Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Menus, OleCtrls, ShellApi, jpeg;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Label100: TLabel;
    Panel1: TPanel;
    Panel2: TPanel;
    Memo1: TMemo;
    MainMenu1: TMainMenu;
    N1: TMenuItem;
    N2: TMenuItem;
    N3: TMenuItem;
    N4: TMenuItem;
    N5: TMenuItem;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure N2Click(Sender: TObject);
    procedure N4Click(Sender: TObject);
    procedure N5Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses about;

{$R *.dfm}
var
  nadpis: array[0..99] of TLabel;
  sannar: array[0..99] of TLabel;
  i, j, k, numerfont, answer: integer;
  namefont: string;
```



```

procedure updatel;
// Процедура для обновления

begin
    randomize;

    // Присвоение случайного из 4 шрифтов для нужных чисел 0, 9, 18,

    numerfont:=random(81);
    if numerfont > 40
    then
        begin
            if numerfont > 60
            then
                namefont:='Wingdings'
            else
                namefont:='Wingdings 2'
            end
        else
            begin
                if numerfont < 20
                then
                    namefont:='Wingdings 3'
                else
                    namefont:='Webdings'
                end;
            end;

    // Установление отдельного шрифта для каждой из 10 строк
    answer:=random(10);
    for i:=0 to 99 do
        begin
            j:=i div 10; // порядок строки
            k:=i mod 10; // порядок столбца
            case j of
                1,5 : nadpis[i].Font.Name:='Wingdings 3';
                2,6 : nadpis[i].Font.Name:='Wingdings 2';
                3,7,0: nadpis[i].Font.Name:='Wingdings'
            else nadpis[i].Font.Name:='Webdings';
            end;

            // Установление рисунков
            if (i mod 9) = 0
            then
                begin
                    nadpis[i].Font.Name:=namefont;
                    nadpis[i].Caption:=inttostr(answer)
                end
            else
                nadpis[i].Caption:=inttostr(random(10));
            end;
        end;
    end;
end;

```

```

procedure TForm1.FormCreate(Sender: TObject);
begin
    form1.Caption:='Угадыватель мыслей';
    Button1.Caption:='Обновить';
    Button2.Caption:='Проверить';

    // Формирование надписей с числами
    for i:=0 to 99 do
    begin
        j:=(i div 10);
        k:=(i mod 10);
        sannar[i]:=TLabel.Create(self);
        sannar[i].Parent:=Panell1;
        sannar[i].Wordwrap:=True;
        sannar[i].AutoSize:=false;
        sannar[i].Top:=5+35*(j);
        sannar[i].Left:=5+90*k;
        sannar[i].Width:=40;
        sannar[i].Height:=30;
        sannar[i].font.Size:=16;
        sannar[i].font.Name:='Times New Roman';
        sannar[i].font.Color:=clWindowText;
        sannar[i].font.Style:=[fsBold];
        sannar[i].Alignment:=taCenter;
        sannar[i].Layout:=tlCenter;
        sannar[i].Caption:=inttostr(i);
        if odd(k)
        then
            sannar[i].Color:=rgb(240,240,120)
        else
            sannar[i].Color:=rgb(120,240,240);
        sannar[i].Visible:=enabled;
    end;

    // Формирование надписей с рисунками
    for i:=0 to 99 do
    begin
        j:=trunc(i div 10);
        k:=trunc(i mod 10);
        nadpis[i]:=TLabel.Create(self);
        nadpis[i].Parent:=Panell1;
        nadpis[i].Wordwrap:=True;
        nadpis[i].AutoSize:=false;
        nadpis[i].Top:=5+35*(j);
        nadpis[i].Left:=50+90*k;
        nadpis[i].Width:=40;
        nadpis[i].Height:=30;
        nadpis[i].font.Size:=20;
        nadpis[i].font.Style:=[fsBold];
        nadpis[i].Alignment:=taCenter;
        nadpis[i].Layout:=tlCenter;
        nadpis[i].font.Color:=clWindowText;
        nadpis[i].Visible:=enabled;
    end;

```

```

        if odd(k)
            then
                nadpis[i].Color:=rgb(240,240,120)
            else
                nadpis[i].Color:=rgb(120,240,240);
        end;

    update1;

    label100.Top:=150;
    label100.Width:=400;
    label100.Height:=100;
    label100.Left:=400;
    label100.Font.Size:=50;
    label100.Layout:=tlCenter;
    label100.Alignment:=taCenter;
    label100.Font.Name:=namefont;
    label100.Caption:=inttostr(answer);

end;

procedure TForm1.Button2Click(Sender: TObject);
// Кнопка Проверить / Посмотреть

begin
    if panel1.Visible=true
        then
            begin
                panel1.Visible:=false;
                label100.Visible:=true;
            end
        else
            begin
                panel1.Visible:=true;
                label100.Visible:=false;
            end;
    if Button1.visible=true
        then
            begin
                button1.Visible:=false;
                button2.Caption:='Посмотреть';
            end
        else
            begin
                button1.Visible:=true;
                button2.Caption:='Проверить';
            end;
end;
end;

```

```

procedure TForm1.Button1Click(Sender: TObject);
// Кнопка Обновить
begin
    update1;
    label100.Font.Name:=namefont;
    label100.Caption:=inttostr(answer);

end;

procedure TForm1.N2Click(Sender: TObject);
// Меню Файл - Выход
begin
    if MessageDlg('Вы действительно хотите выйти из
программы?!',mtConfirmation,[mbYes,mbNo],0)= mrYes
    then
        close;
end;

procedure TForm1.N4Click(Sender: TObject);
// Показать панель правил
begin
    //ShellExecute(Handle,'open','.\files\magic.mht',0,0,3);
    panel2.Visible:=true;

end;

procedure TForm1.N5Click(Sender: TObject);
// Показать окно О программе
begin
    form2.showmodal;
end;

procedure TForm1.Button3Click(Sender: TObject);
// Скрыть панель правил
begin
    panel2.Visible:=false;
end;

end.

```

Листинг модуля about

```
unit about;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms,
  Dialogs, StdCtrls, jpeg, ExtCtrls;

type
  TForm2 = class(TForm)
    Label1: TLabel;
    Button1: TButton;
    Image1: TImage;
    Memo1: TMemo;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.dfm}

procedure TForm2.Button1Click(Sender: TObject);
begin
  close;
end;

end.
```

Список литературы

1. Бузур-оол О.Б., Далаа С.М., Ховенмей В.Б. «Информатика и вычислительная техника», Екатеринбург, издательство «Сократ», 1998г
2. Современное программирование на языке Паскаль [Электронный ресурс]. - <http://pascalabc.net/> - (дата обращения: 12.02.2016).
3. Википедия – свободная энциклопедия [Электронный ресурс]. - [https://ru.wikipedia.org/wiki/Delphi_\(язык_программирования\)](https://ru.wikipedia.org/wiki/Delphi_(язык_программирования)). - (дата обращения: 12.02.2016).
4. Портал – Borland.ru [Электронный ресурс]. - <http://borland.ru/> - (дата обращения: 12.02.2016).
5. Онлайн учебник по Delphi 7 [Электронный ресурс]. - <http://delphi.support.uz> - (дата обращения: 12.02.2016).
6. Клуб программистов [Электронный ресурс]. - <http://www.programmersclub.ru/book/> - (дата обращения: 12.02.2016).
7. Учебник Дельфи для начинающих [Электронный ресурс]. - <http://bourabai.ru/einf/Delphi/> - (дата обращения: 12.02.2016).