

Тема урока: Язык программирования как средство информационного компьютерного моделирования.

Разработка моделирующих компьютерных программ.

Теоретическая часть.

Графические и численные методы решения уравнений

На языке алгебры формальные модели записываются с помощью уравнений, точное решение которых основывается на поиске равносильных преобразований алгебраических выражений, позволяющих выразить переменную величину с помощью формул.

Точные решения существуют только для некоторых уравнений определенного вида (линейные, квадратные, тригонометрические, ...), поэтому для большинства уравнений приходится использовать методы приближенного решения с заданной точностью (графические или численные).

Например, нельзя найти корень уравнения $x^3 - \cos(x) = 0$ путем равносильных алгебраических преобразований. Однако такие уравнения можно решать приближенно графическими и численными методами.

Графический метод решения уравнений

Построение графиков функций может использоваться для грубо приближенного решения уравнений. Для уравнений вида $f(x)=0$, где $f(x)$ – некоторая непрерывная функция, корень (или корни) этого уравнения являются точкой (или точками) пересечения графика функции с осью X.

Графическое решение таких уравнений можно осуществить путем построения компьютерной модели.

Численные методы решения уравнений

Для решения уравнений с заданной точностью можно применить разработанные в вычислительной математике численные методы решения уравнений путем последовательных приближений. Самый простой из них – метод половинного деления. Если мы определим числовой отрезок аргумента x , на котором существует корень, и функция на краях этого отрезка принимает значения разных знаков, то можно использовать метода половинного деления.

Практическая часть

Графический метод

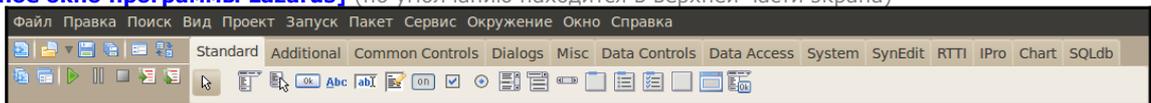
На основании формальной модели, описывающей движение тела, брошенного под углом к горизонту, создадим компьютерную модель с использованием системы программирования Lazarus.

Для этого:

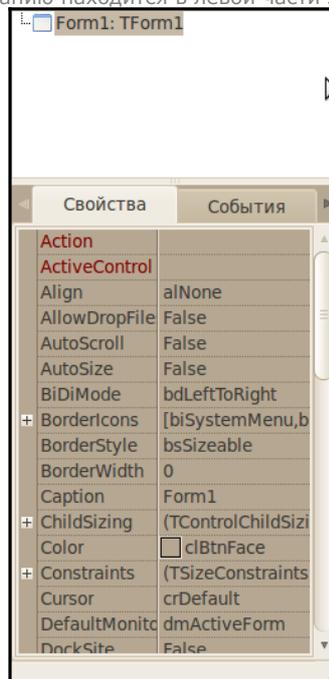
1) Запустить среду разработки приложений (систему программирования) Lazarus

- 1.1) Приложения – Программирование – Lazarus
- 1.2) На экране появятся окна данной программы:

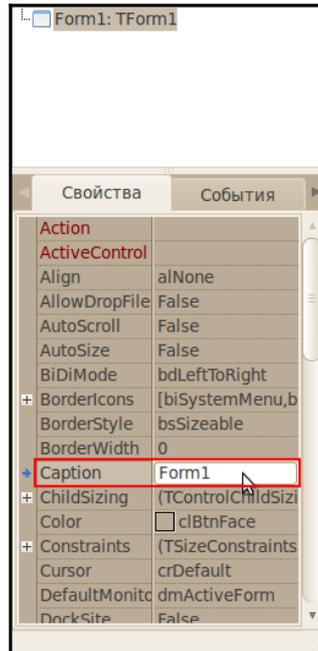
[Главное окно программы Lazarus] (по умолчанию находится в верхней части экрана)



[Окно «Инспектор объектов»] (по умолчанию находится в левой части экрана)

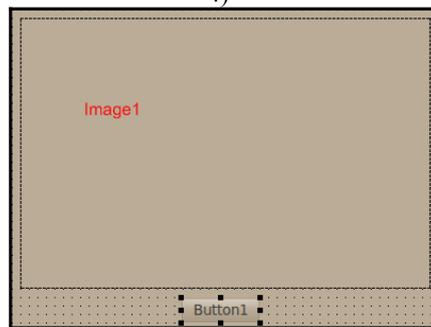


- 2.1) Делаем щелчок левой кнопкой мыши по форме **Form1** и в окне «**Инспектор объектов**» (**Object Inspector**) находим свойство **Caption** (Название, Заголовок) и меняем его значение с **Form1** на **График функции** (для подтверждения изменений нужно нажать кнопку **Enter**), после этого название формы окна изменится на то, что вы ввели:

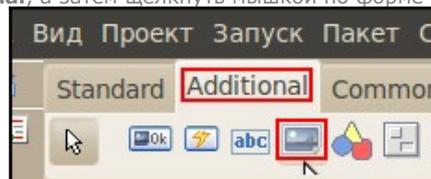


- 3) **Разместить** на форме элементы согласно рисунку ниже:

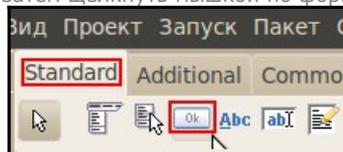
4)



- 3.1) **Добавить** компонент **Image1** (контейнер для изображения) можно щелкнув мышкой по значку **Image** на панели инструментов на вкладке **Additional**, а затем щелкнуть мышкой по форме



- 3.2) **Добавить** компонент **Button1** (кнопка обыкновенная, батон) можно щелкнув мышкой по значку **Button** на панели инструментов на вкладке **Standard**, а затем щелкнуть мышкой по форме



- 3.3) **Сделать** настройки для каждого из элементов, расположенных на форме в соответствии с теми, что приведены ниже:

Настройка свойств производится следующим образом:

- на форме щелкаем левой кнопкой мыши по нужному объекту (он выделится квадратиками),
- далее переходим к окну «**Инспектор объектов**» и изменяем свойства так, как указано ниже:

Форма окна (**Form1**)

Caption -> **График функции** (название окна, заголовок окна)
 Height -> **448** (высота формы)
 Width -> **415** (ширина формы)

Кнопка (**Button1**)

Caption -> **Построить график** (название кнопки)
 Left -> **144** (отступ кнопки от левого края формы)
 Top -> **416** (отступ кнопки от верхнего края формы)
 Width -> **144** (ширина кнопки)

Изображение (**Image1**)

Height -> **400** (высота картинки)
Left -> **8** (отступ картинки от левого края формы)
Top -> **8** (отступ картинки от верхнего края формы)
Width -> **400** (ширина картинки)

4) Отредактировать код программы в соответствии с тем, как указано ниже:

4.1) Перейти к окну «Редактор исходного кода»:

4.2) Находим следующий код:

```
uses  
Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs;
```

...и дописываем к нему **Math** (модуль, который обеспечивает выполнение математических функций в создаваемой программе). Должно получиться следующее:

```
uses  
Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs, Math;
```

4.3) Далее находим следующий код:

```
var  
Form1: TForm1;
```

...и дописываем к нему переменные, которые будут использоваться во время работы программы

```
var  
Form1: TForm1;  
x, y : real ;  
n : integer ;
```

4.4) Далее напишем код программы, который будет выполняться при нажатии на кнопку **Построить график**. Для этого

делаем двойной щелчок по этой кнопке. На первый план перейдет окно «Редактор исходного кода» со следующим кодом:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
  
end;
```

...приводим его к следующему виду:

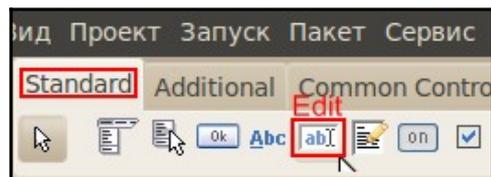
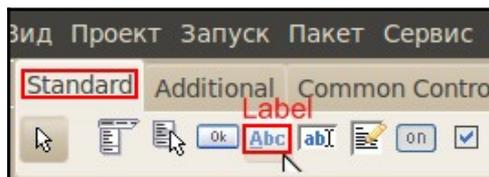
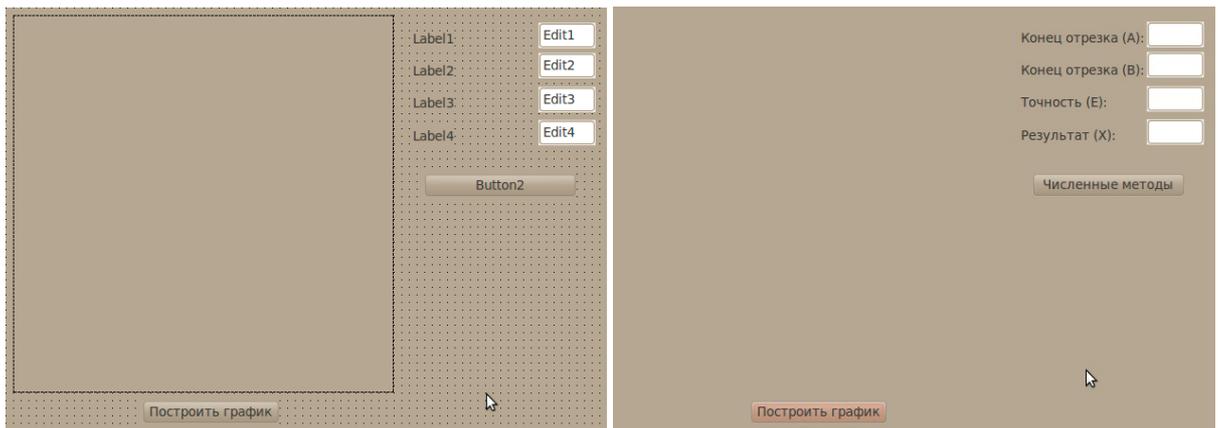
```
procedure TForm1.Button1Click(Sender: TObject);  
Begin  
With Image1.Canvas do  
begin  
//график функции  
x := -3;  
while x < 3 do  
begin  
x := x + 0.001;  
y := x*x*x - Cos(x);  
Pixels[Round(100*x) + 200, 200-round(20*y)] := clBlue;  
end;  
Pen.Color := clGreen;  
MoveTo(0, 200); Lineto(500, 200); //ось X  
MoveTo(250, 0); LineTo(250, 500); //ось Y  
  
//шкала оси X  
N := 0;  
Pen.Color := clGreen;  
while n < 500 do  
begin  
n := n + 100;  
MoveTo (n, 190); LineTo (n, 210);  
TextOut (n, 200, FloatToStr(Round(n - 250)/50));  
end;  
//шкала оси Y  
N := 0;  
Pen.Color := clGreen;  
while n < 400 do  
begin  
n := n + 100;  
MoveTo(245, 400-n); LineTo(255, 400-n);  
TextOut(245, 400-n, FloatToStr(Round(n - 200)/10));  
end;  
end;  
end;
```

5) Нажать кнопку **F9** (произойдет компиляция и запуск программы на выполнение), нажать в окне программы кнопку «Построить график» и проанализировать полученные результаты

График функции пересекает ось X один раз, следовательно, уравнение имеет один корень. По графику грубо приближенно можно определить, что **x = 0,8**.

6) Добавить в программу Численный метод половинного деления

6.1) Необходимо привести нашу программу к следующему виду:



...для этого делаем следующее:

Форма окна (Form1)

Width -> **632** (ширина формы)

Надпись (Label1)

Caption -> **Конец отрезка (A):** (название надписи)

Left -> **428** (отступ надписи от левого края формы)

Top -> **25** (отступ надписи от верхнего края формы)

Надпись (Label2)

Caption -> **Конец отрезка (B):** (название надписи)

Left -> **428** (отступ надписи от левого края формы)

Top -> **59** (отступ надписи от верхнего края формы)

Надпись (Label3)

Caption -> **Точность (E):** (название надписи)

Left -> **428** (отступ надписи от левого края формы)

Top -> **93** (отступ надписи от верхнего края формы)

Надпись (Label4)

Caption -> **Результат (X):** (название надписи)

Left -> **428** (отступ от левого края формы)

Top -> **128** (отступ от верхнего края формы)

Текстовое поле (Edit1)

Left -> **560** (отступ от левого края формы)

Top -> **16** (отступ от верхнего края формы)

Width -> **60** (ширина текстового поля)

Text -> (оставить поле пустым)

Текстовое поле (Edit2)

Left -> **560** (отступ от левого края формы)

Top -> **48** (отступ от верхнего края формы)

Width -> **60** (ширина текстового поля)

Text -> (оставить поле пустым)

Текстовое поле (Edit3)

Left -> **560** (отступ от левого края формы)

Top -> **84** (отступ от верхнего края формы)

Width -> **60** (ширина текстового поля)

Text -> (оставить поле пустым)

Текстовое поле (Edit4)

Left -> **560** (отступ от левого края формы)

Top -> **119** (отступ от верхнего края формы)

Width -> **60** (ширина текстового поля)

Text -> (оставить поле пустым)

Кнопка (Button2)

Caption -> **Численные методы** (название кнопки)

Left -> **440** (отступ от левого края формы)

Top -> **176** (отступ от верхнего края формы)

Width -> **160** (ширина кнопки)

6.2) Далее необходимо написать код программы, который будет выполняться при нажатии на кнопку «Численные методы». Для этого делаем двойной щелчок мышкой по кнопке. На первый план перейдет окно «Редактор исходного кода» со следующим кодом:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
end;
```

...приводим его к следующему виду:

```
procedure TForm1.Button2Click(Sender: TObject);
Var a,b,c,e:real;
begin
a := strtfloat(edit1.text);
b := strtfloat(edit2.text);
e := strtfloat(edit3.text);
Repeat
c := (a+b)/2;
if (a*a*a - cos(a))*(c*c*c - cos(c)) < 0 then b := c else a := c;
Until (b-a) / 2 < e;
Edit4.text := floattostr((a+b)/2);
end;
```

7) Нажать кнопку F9 (произойдет компиляция и запуск программы на выполнение)

7.1) Нажать в окне программы кнопку «Построить график», затем ввести в поле «Начало отрезка (а)» значение **0**, ввести в поле «Конец отрезка (В)» значение **1**, ввести в поле «Точность (Е)» значение **0.001** и нажать кнопку «Численные методы» и посмотреть на результат, расположенный в поле «Результат (Х)». Это и будет точный корень решения нашего уравнения

