Практическая работа № 15, НАХОЖДЕНИЕ КОРНЯ ФУНКЦИИ НА ОТРЕЗКЕ

Постановка задачи

Разработать программу, которая выполняет следующие действия:

1. вводит значения начала и конца отрезка, точность;
2. вычисляет значение корня заданной функции с помощью трех методов приближенного вычисления (метод половинного деления, метод хорд, метод касательных);
3. определяет, какой метод наиболее оптимален для заданной функции (по количеству итераций).

В программе необходимо предусмотреть:

1. контроль вводимой информации.

Ограничение:

1. формула функции задается непосредственно в программе;
2. точность вычислений 0,000001;
3. на отрезке существует один корень и функция непрерывна.

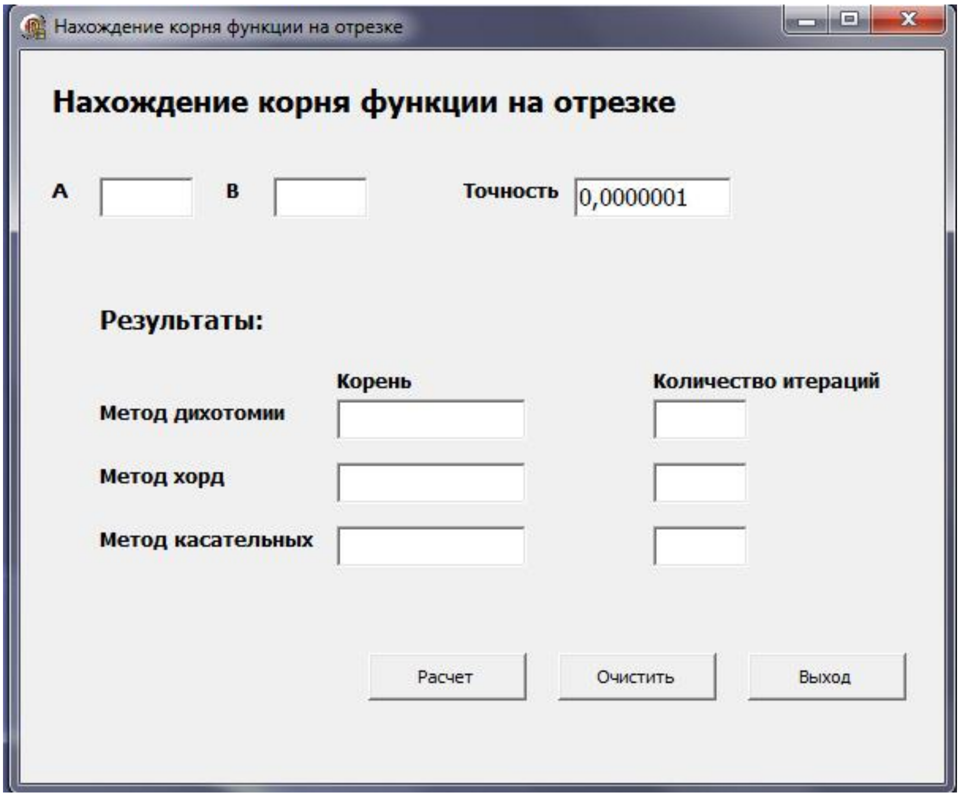


Рис.1

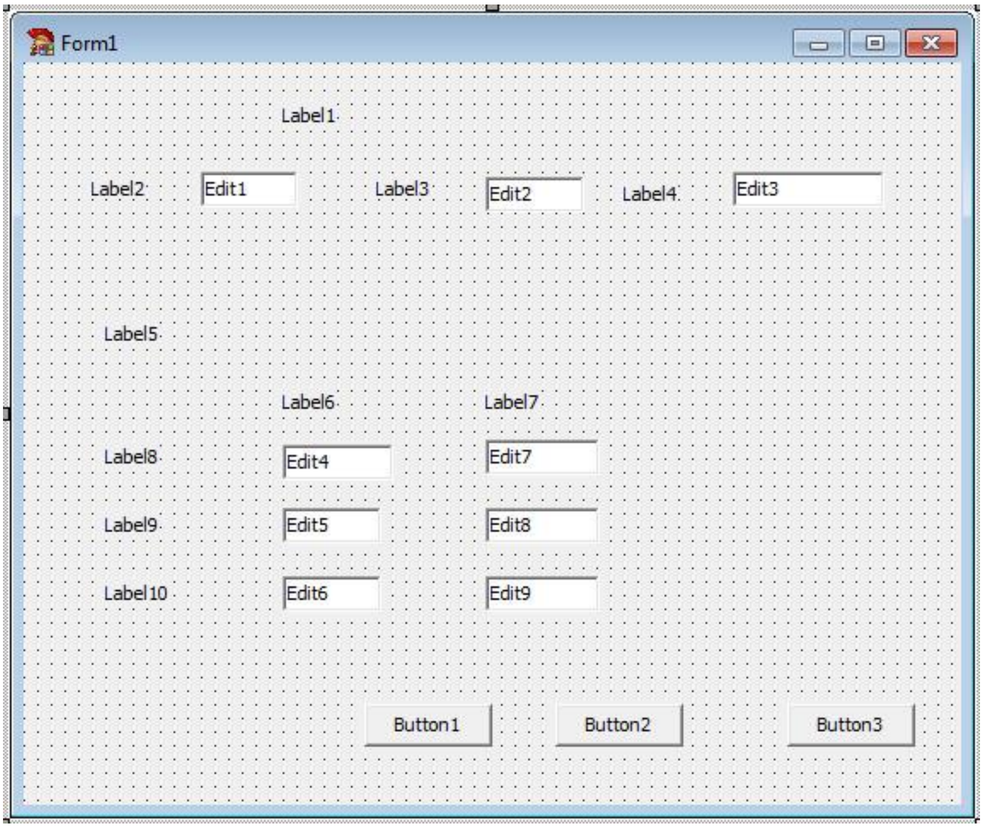


Рис.2

Новым в этой работе являются:

- методы приближенного вычисления (метод половинного деления, метод хорд, метод касательных);
- функция **FindComponent**, позволяющая работать с большим количеством компонентов одного типа.

План разработки программы

1. Откройте новый проект.

2. Сохраните код программы и проект под именами, например, **Unit_17.pas** и **Pr_17.dpr**. В последующем сохраняйте проект и проверяйте его работоспособность после каждого изменения.

3. Разместите в форме объекты в соответствии с рис.2.

Не стремитесь располагать компоненты на свои места. Это мы сделаем в программе при создании формы. Для этого выделите объект **Form1**, перейдите на вкладку **Events Инспектора объектов** (**Object Inspector**), найдите событие **OnCreat**, справа от него дважды щелкнуть левой кнопкой мыши. Попав в код программы, надо написать следующий код:

```
// Описание формы
Form1.Width:=600;
Form1.Height:=500;
Form1.Caption:='Нахождение корня функции на отрезке';

// Описание метки - заголовка
Label1.Caption:='Нахождение корня функции на отрезке';
Label1.Left:=20;
```

```
Label1.Top:=20;
Label1.Font.Style:=[fsBold];
Label1.Font.Size:=14;

// Описание метки – начала отрезка
Label2.Caption:='A';
Label2.Left:=20;
Label2.Top:=80;
Label2.Font.Style:=[fsBold];
Label2.Font.Size:=10;
...
```

Подобным образом необходимо описать все расположенные компоненты на форме.
Для упрощения записи представим все свойства компонентов в виде трех таблиц.

Таблица 1

Компонента	**Свойства**				
	Caption	**Left**	**Top**	**Font.Style**	**Font.Size**
Label1	Нахождение корня функции на отрезке	20	20	fsBold	14
Label2	A	20	80	fsBold	10
Label3	B	130	80	fsBold	10
Label4	Точность	280	80	fsBold	10
Label5	Результаты	50	160	fsBold	12
Label6	Корень	200	200	fsBold	10
Label7	Количество итераций	400	200	fsBold	10
Label8	Метод дихотомии	50	220	fsBold	10
Label9	Метод хорд	50	260	fsBold	10
Label10	Метод касательных	50	300	fsBold	10

Таблица 2

Компонента	**Свойства**					
	Text	**Left**	**Top**	**Width**	**Height**	**Font.Size**
Edit1	''	50	80	60	20	11
Edit2	''	160	80	60	20	11
Edit3	''	350	80	100	20	11
Edit4	''	200	220	120	20	11
Edit5	''	200	260	120	20	11
Edit6	''	200	300	120	20	11
Edit7	''	400	220	60	20	11
Edit8	''	400	260	60	20	11
Edit9	''	400	300	60	20	11

Таблица 3

Компонента	Свойства				
	Caption	Left	Top	Width	Height
Button1	Расчет	220	380	100	30
Button2	Очистить	340	380	100	30
Button3	Выход	460	380	100	30

4. Компоненты **Edit4** – **Edit9** должны быть недоступны для редактирования, т.е. свойство **Enabled** принимает значение **False**. Для этого необходимо записать соответствующие операторы присваивания, но для упрощения записи воспользуемся функцией **FindComponent**, позволяющей работать с большим количеством компонентов одного типа (найти тот или иной компонент по маске его названия). Имена компонентов определяются как Имя_Компонента + Уникальный_Индекс.

В нашем случае для изменения свойства **Enabled** компонентам **Edit4** – **Edit9** необходимо внести дополнения в процедуру **TForm1.FormCreate**:

```
Var I: Integer;
...
For I:=4 To 9 do
 TEdit(FindComponent('Edit'+IntToStr(I))).Enabled:=False;
```

Немного теории

Численные методы нахождения корней

В научных и инженерных задачах часто требуется решить уравнение вида

$$f(x) = 0,$$

где $f(x)$ – заданная непрерывная нелинейная функция.

Методы решения нелинейного уравнения можно разделить на ***точные*** (аналитические) и ***приближенные*** (итерационные, численные). В точных методах корень представляется некоторой алгебраической формулой, но это можно сделать только для простейших уравнений. В большинстве случаев приходится решать эту задачу с использованием приближенных методов.

Приближенное решение уравнения проводят в два этапа:

1 этап – поиск интервалов, на которых расположен только один корень,

2 этап – уточнение значения корней с заданной точностью.

Примечание

При приближенных вычислениях нельзя найти *точное* значение корня и добиться обращения функции в ноль. Критерием уточнения может служить *абсолютная* или *относительная погрешность* корня. Если корень близок к нулю, то лишь относительная погрешность даст необходимое число значащих цифр. Если же он весьма велик по абсолютной величине, то критерий абсолютной погрешности часто дает совершенно излишние верные цифры. Для функций, быстро изменяющихся в окрестности корня, может быть привлечен и критерий *абсолютная величина значения функции* не превышает заданной допустимой погрешности.

Для уточнения корней можно воспользоваться различными методами: методом половинного деления (дихотомии), методом хорд (секущих), методом Ньютона (касательных).

Постановка задачи

Пусть непрерывная функция $f(x)$ на концах отрезка $[a,b]$ имеет значения разных знаков, т.е. $f(a)\times f(b) < 0$ и на отрезке существует один корень. Найти корень на заданном отрезке.

Метод половинного деления (дихотомия)

Возьмем середину отрезка $c = (a + b)/2$.

Если $f(a)\times f(c) \leq 0$, то корень принадлежит отрезку от a до $(a + b)/2$.

Если $f(c)\times f(b) \leq 0$, то корень принадлежит отрезку от $(a + b)/2$ до b.

Выбираем подходящий отрезок, вычисляем значение функции в его середине. Итерации продолжаются до тех пор, пока длина очередного отрезка не окажется меньше заданной предельной абсолютной погрешности $|b - a| < \varepsilon$ или $f(c) < \varepsilon$.

Геометрически этот метод можно представить следующим образом:

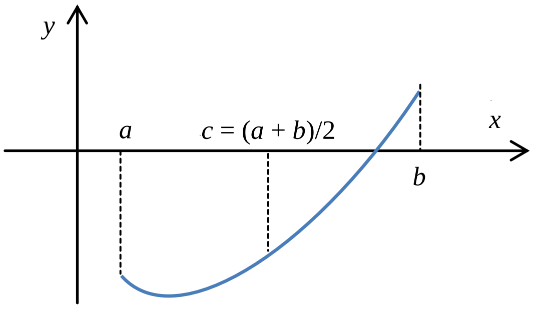


Рис.3, Метод половинного деления (1-ый шаг)

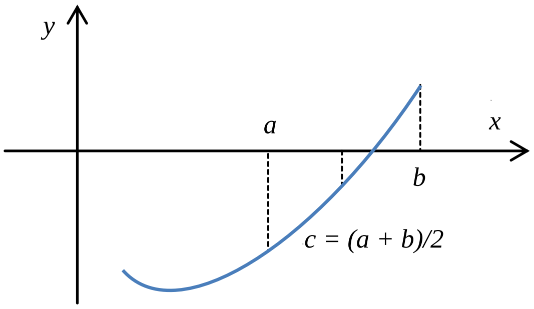


Рис.4, Метод половинного деления (2-ой шаг)

Метод хорд (секущих)

Между точками a и b строится хорда, стягивающая $f(x)$. Очередное приближение берется в точке c, где хорда пересекает ось абсцисс. В качестве нового интервала для продолжения итерационного процесса выбирается тот, на концах которого функция имеет разные знаки.

Для вывода итерационной формулы процесса найдем точку пересечения хорды (описываемой уравнением прямой) с осью абсцисс.

Уравнение прямой:

$y = k\times x + z$

Для того, чтобы найти значения коэффициентов k и b составим систему уравнений:

$$\begin{cases} f(a) = k \times a + z \\ f(b) = k \times b + z \end{cases}$$

Вычтем из первого уравнение второе:

$f(a) - f(b) = k\times(a - b)$, тогда если $a \neq b$

$$k = \frac{f(a) - f(b)}{a - b}$$

$$z = f(a) - \frac{f(a) - f(b)}{a - b} \times a$$

Тогда уравнение прямой (хорды):

$$y = \frac{f(a) - f(b)}{a - b} \times x + f(a) - \frac{f(a) - f(b)}{a - b} \times a$$

$$y = f(a) + \frac{f(a) - f(b)}{a - b} \times (x - a)$$

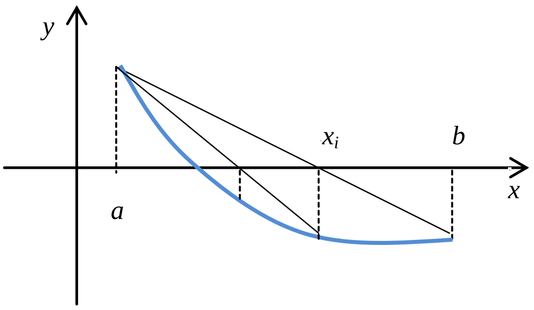
В точке пересечения прямой (хорды) с осью абсцисс (точка c) полученное уравнение равно 0. После преобразований получим формулу для вычисления значения c:

$$c = a - \frac{f(a) \times (a - b)}{f(a) - f(b)}$$

Ограничения: $a \neq b$ и $f(a) \neq f(b)$

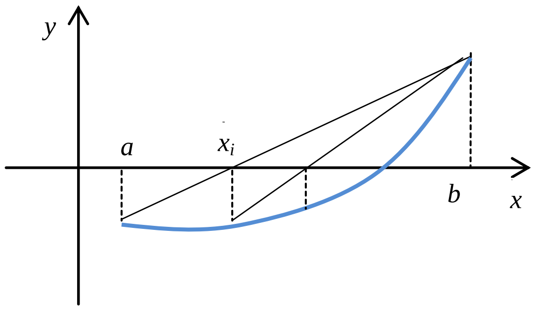
Если первая и вторая производные имеют разные знаки, т.е. $f'(x) \times f''(x) < 0$, то все приближения к корню выполняются со стороны правой границы отрезка [a, b] и вычисляется по формуле

$$x_{i+1} = x_i - \frac{x_i - a}{f(x_i) - f(a)} \times f(x_i)$$



Если первая и вторая производные имеют одинаковые знаки, т.е. $f'(x) \times f''(x) > 0$, то все приближения к корню выполняются со стороны правой границы отрезка [a, b] и вычисляется по формуле

$$x_{i+1} = x_i - \frac{b - x_i}{f(b) - f(x_i)} \times f(x_i)$$



Метод Ньютона (касательных)

В качестве начального приближения выбираем тот конец отрезка $[a, b]$, на котором функция $f(x)$ и ее вторая производная $f''(x)$ имеют одинаковые знаки: $f(b) \times f''(b) > 0$.

Пусть b – начальное приближение к корню, а $f(x)$ имеет непрерывную производную. Следующее приближение к корню найдем в точке c, где касательная к функции $f(x)$, проведенная из точки $(b, f(b))$, пересекает ось абсцисс. Затем точно так же обрабатываем точку$(c, f(c))$, организуя итерационный процесс. Выход из итерационного процесса по условию $f(c) < \varepsilon$.

Для вывода итерационной формулы процесса найдем точку пересечения касательной (описываемой уравнением прямой) с осью абсцисс.

Геометрически этот метод можно представить следующим образом:

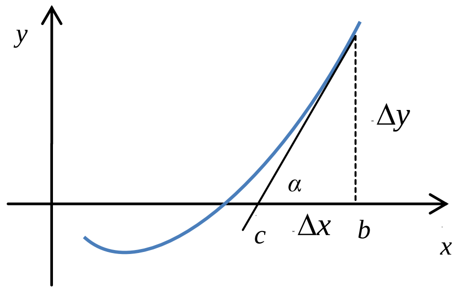


Рис.7, Метод Ньютона (1-ый шаг)

$$tg(\alpha) = f'(b)$$

$$tg(\alpha) = \frac{\Delta y}{\Delta x} = \frac{f(b)}{b-c}$$

$$f'(b) = \frac{f(b)}{b-c}$$

$$c = b - \frac{f(b)}{f'(b)}$$

Ограничение: $f'(b) \neq 0$

5. Описание функции, для которой будут выполняться все вычисления, по условию поставленной задачи должно находиться в самой программе. Для метода касательных нам понадобится еще и производная функции. Внесем изменения в программу, добавив описания функции и производной функции. Оформим это в виде отдельных функций.

Для примера возьмем $f(x) = x^3 + 2.25 \times x^2 + 1.25 \times x - 1.5$

```
function FX (X: Real): Real;
  // Значение функции
  begin
    FX:=X*X*X+0.25*X*X+1.25*X-1.5;
  end;

function FX_P (X: Real): Real;
  // Значение производной
  begin
    FX_P:=3*X*X+0.5*X+1.25;
  end;
```

6. Для выполнения расчетов нам понадобится ввести переменные:

```
Var EPS : Real;    // Точностьрасчета
    A, B : Real;   // Концы отрезка
    C : Real;      // Расчетное значение переменной функции
    NX : Integer; // Количество итераций
```

7. Для ввода исходных значений *A*, *B*, *EPS* предназначены компоненты **Edit1**, **Edit2**, **Edit3**. Особенностью вводимой информации является то, что это числовая информация вещественного типа. Для контроля вводимой информации в окно **Edit1** можно воспользоваться следующей процедурой обработки события **OnKeyPress**:

```
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
case Key of
  #8,'0'..'9' :  ;                  // цифры и клавиша <Back Space>
  '.', ',' :
      begin
      // точку и запятую заменим допустимым разделителем
         Key := DecimalSeparator;
     // запрет ввода второго разделителя
      if Pos(DecimalSeparator, Edit1.Text)<> 0 then Key := Chr(0);
       end;

  '-':    // минус можно ввести только первым символом,
          // т.е. когда ячейка пустая
        if Length(Edit1.Text) <> 0 then Key := Chr(0);
  #13:   Edit2.SetFocus;            // <Enter> переход в окно Edit2

 else key := Chr(0);  // остальные символы заменяются пустым символом
end;
end;
```

По материалам книги Никиты Культина «Основы программирования в Delphi 7»

8. Оформите процедуры обработки Edit2 и Edit3 самостоятельно и протестируйте полученную программу.

9. Для выполнения самого расчета у нас предназначена кнопка **Button1** (**Расчет**), поэтому оформим процедуру обработки события **OnClick**.

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

EPS:=StrToFloat(Edit3.Text); // точность расчета

A:=StrToFloat(Edit1.Text);   // начальное значение отрезка
B:=StrToFloat(Edit2.Text);   // конечное значение отрезка

Metod_1;                     // процедура расчета по выбранному методу
Edit4.Enabled:= True;
Edit4.Text:=floatToStrF(C,ffFixed, 10, 7);
Edit4.Enabled:= False;
Edit7.Enabled:= True;
Edit7.Text:=IntToStr(NX);
Edit7.Enabled:= False;
...
end;
```

Комментарии

В начале процедуры заполняются значения переменных *EPS*, *A*, *B*, которые будут использованы в расчетах. Значения берутся из соответствующих компонент (**Edit1**, **Edit2**, **Edit3**).

После обращения к процедуре расчет по выбранному методу (Metod_1), которая не использует параметры, заполняются соответствующие графы формы полученными результатами (*C* – приближенное значение корня, *NX* – количество итераций). Для заполнения необходимо сделать компоненты Edit4 и Edit7 доступными, а после заполнения сделать недоступными.

10. Процедура, выполняющая расчет по методу половинного деления выглядит так:

```
procedure Metod_1;
  // Метод половинного деления
  begin
  NX:=0;

  Repeat
  NX:=NX+1;
  C:=(A+B)/2;
  If FX(A)*FX(C) <= 0 Then B:=C;
  If FX(B)*FX(C) <= 0 Then A:=C;
  Until (abs(B - A)<EPS) or (abs(FX(C))<EPS);
  end;
```

Протестируйте полученную программу, введя исходные данные:

$A = -1.5$, $B = 1$, $EPS = 0{,}0001$

У вас должен получиться следующий результат:

$C = 0.7499847$, $NX = 15$

Задание для самостоятельного выполнения

1. Вставьте обработку нажатия кнопки **Button3** (**Выход**).

2. Вставьте обработку нажатия кнопки **Button2** (**Очистить**), которая очищает все компоненты **Edit1-Edit9**. Для этого воспользуйтесь функцией **FindComponent**, позволяющей работать с большим количеством компонентов одного типа (см.п.4).

3. Дополните программу возможностью нахождения корня методом хорд и методом касательных.

4. Что будет с программой, если ввести $A = 1$, $B = -1$? Внесите необходимые изменения для того, чтобы программа работала корректно.

5. Протестируйте полученную программу для нахождения корня функции $y = x^2 - 4$ на отрезке [–3, –1]. Внесите необходимые изменения для того, чтобы программа работала корректно.

6. Выберите из таблицы вариантов заданий свое задание. Внесите необходимые изменения в программу для того, чтобы выполнить расчеты и получить значение корня. Проверьте правильность расчетов с помощью Excel.

Варианты заданий

№ варианта	Функция	Отрезок	
		Начало	Окончание
1.	$x^4 + 3x - 20$	-3	-2
2.	$2 - \lg x - x$	1	2
3.	$e^x + x - 2$	0	1

4.	$2^x - x - 3$	-3	-2
5.	$8 \ln x + 5x$	0,1	1,1
6.	$\sqrt{x} - \cos 0{,}378x$	0,5	1
7.	$\mathrm{tg}\,(0{,}5\,x + 0{,}1) - x^2$	-1	0
8.	$\lg x - 10^{-x}$	0,5	2
9.	$\sin^2 x - \ln x$	1	3
10.	$\sin x - 1/x$	0,5	2
11.	$e^x - 1/\sin x$	0,1	1,5
12.	$\cos x - \ln(x + 1)$	0,5	2