

# Углубленный курс информатики

---

Итерационные методы решения  
нелинейных уравнений

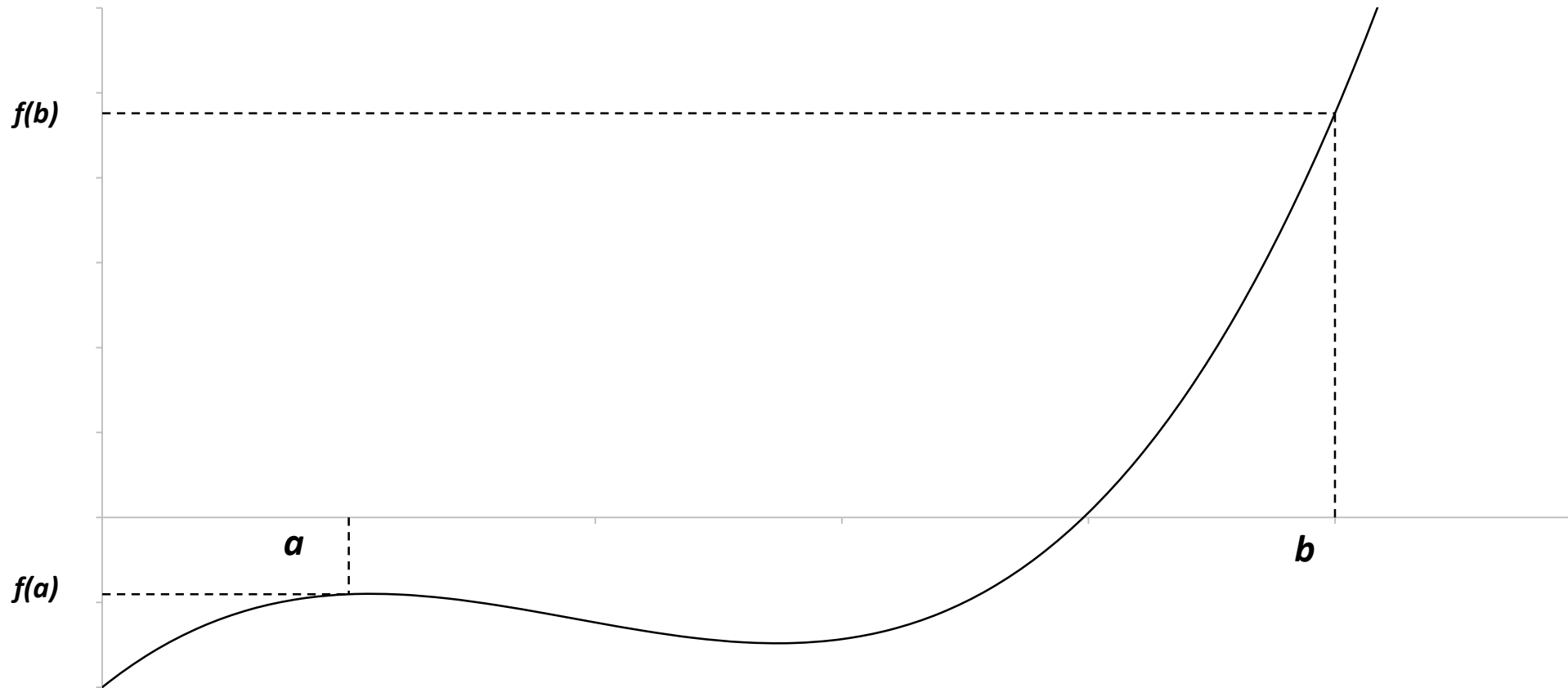


Чузов Вячеслав Алексеевич

к.т.н., доцент ОХИ ИШПР

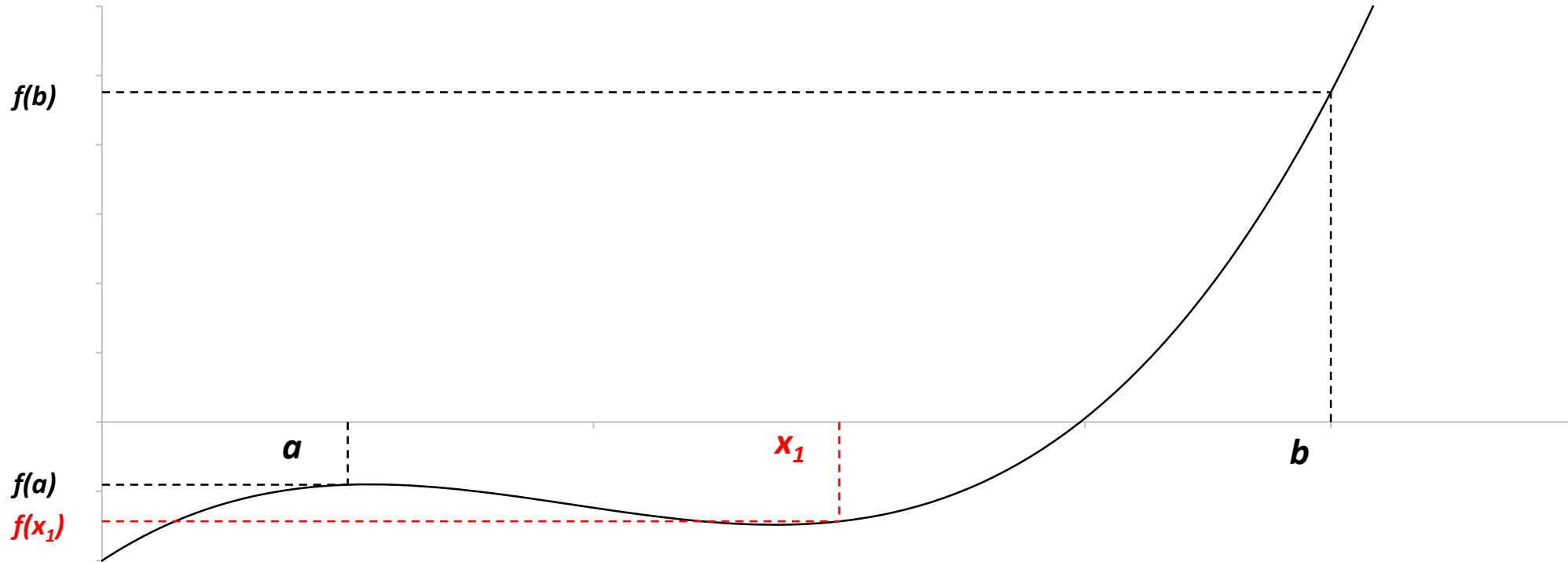
# МЕТОД ДЕЛЕНИЯ ОТРЕЗКА ПОПОЛАМ

- Пусть дана функция  $f(x)$ , непрерывная на интервале  $[a, b]$ .
- $f(a) * f(b) < 0$  – условие наличия корня на данном интервале.



# МЕТОД ДЕЛЕНИЯ ОТРЕЗКА ПОПОЛАМ

- Поделим отрезок  $[a, b]$  пополам точкой  $x_1$  с координатой  $x_1 = (a + b) / 2$  и вычислим значение функции  $f(x_1)$ .

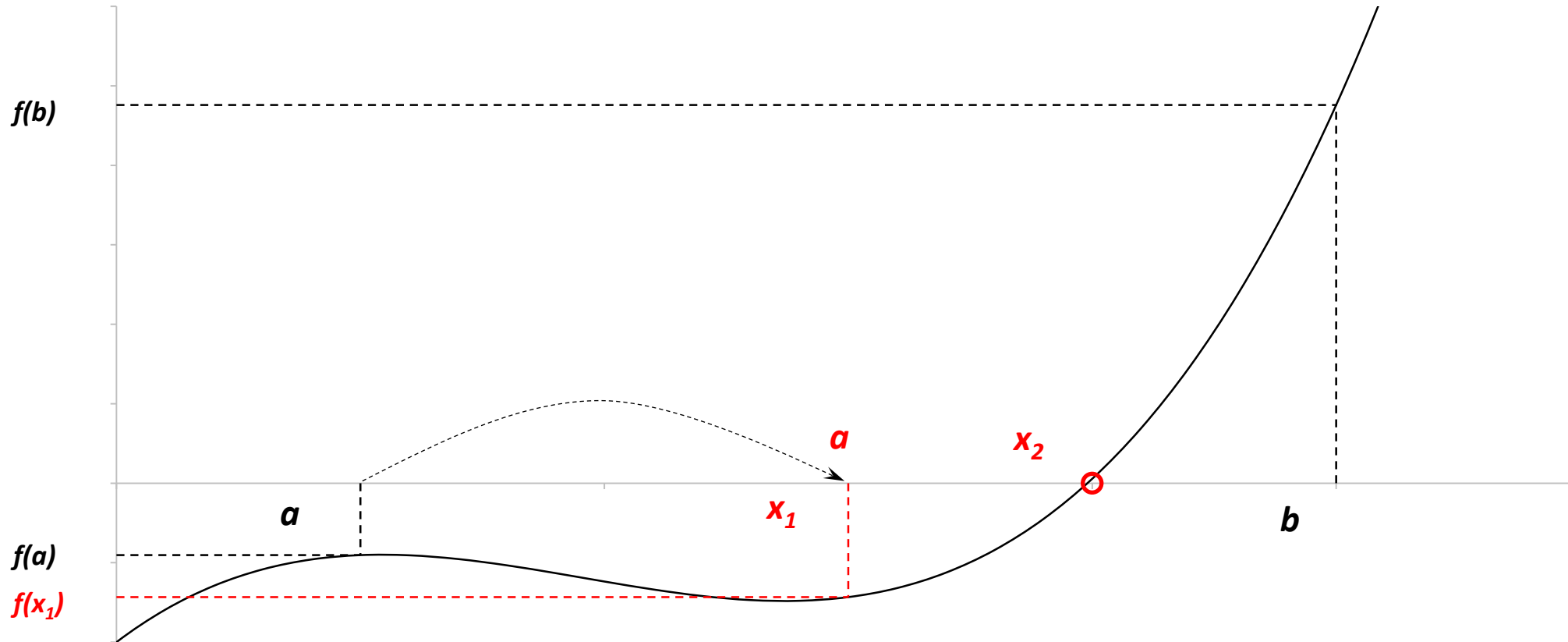


Возможны два случая:

1.  $f(a) * f(x_1) > 0 \rightarrow$  корень находится на отрезке  $[x_1, b]$ , исключаем отрезок  $[a, x_1]$ ;
2.  $f(a) * f(x_1) < 0 \rightarrow$  корень находится на отрезке  $[a, x_1]$ , исключаем отрезок  $[x_1, b]$ .

# МЕТОД ДЕЛЕНИЯ ОТРЕЗКА ПОПОЛАМ

- Продолжаем деление пополам до тех пор, пока длина оставшегося интервала  $[a, b]$  не станет меньше некоторой заданной малой величины  $eps$ , т.е.  $(b - a) < eps$ , и тогда любое значение аргумента из отрезка  $[a, b]$  можно считать корнем с погрешностью  $eps$ . Обычно принимают в качестве корня середину отрезка.



## Пример

Дано нелинейное уравнение:

$$e^x - 6 * x - 3 = 0$$

$$[-3, 1]$$

Точность  $\text{eps} = 0.0001$

Необходимо найти корень данного уравнения на данном отрезке с заданной точностью, используя метод деления отрезка пополам.

```
function equation(x: real): real;
begin
    result := exp(x) - 6 * x - 3
end;
```

```
function HalfDivision(f: function(x: real): real;
                      a, b, eps: real): real;
```

```
begin
    if f(a) * f(b) > 0 then
        begin
            Println('Half Division Method, Not solved!');
            exit
        end;
```

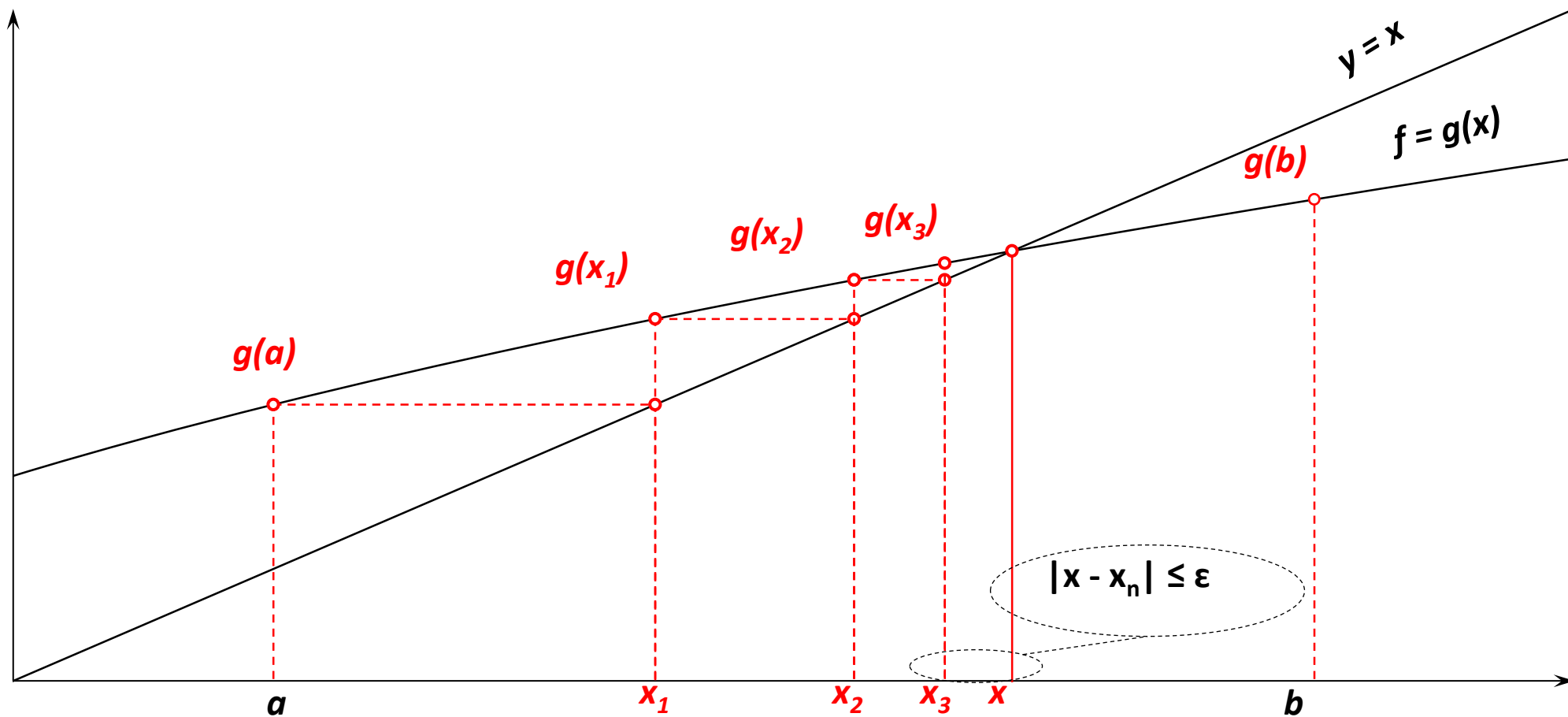
```
    result := (a + b) / 2;
    while abs(a - b) >= eps do
        begin
            if f(a) * f(result) > 0 then
                a := result
            else
                b := result;
            result := (a + b) / 2
        end;
    end;
```

```
begin
    HalfDivision(equation, -3, 1, 1e-4).Print
end.
```

-0.38677978515625

# МЕТОД ПРОСТЫХ ИТЕРАЦИЙ

- Заменим исходное уравнение  $f(x) = 0$ , на эквивалентное  $g(x) = x$  и будем строить итерации по правилу  $x_{n+1} = g(x_n)$ .
- Условие сходимости:  $|g'(x)| < 1$



## Пример

$$e^x - 6 * x - 3 = 0$$

$$[-3, 1]$$

Точность eps = 0.0001

- Запишем выражение для эквивалентной функции:

$$g(x) = \frac{\exp(x) - 3}{6}$$

- Первое приближение  $x_1$  найдем как  $g(-3)$ :

$$x_1 = g(-3) = \frac{\exp(-3) - 3}{6} = -0.4917$$

- Следующее приближение  $x_3$  найдем как  $g(x_2)$ :

$$x_3 = g(x_2) = \frac{\exp(-0.3981) - 3}{6} = -0.3881$$

- Следующее приближение  $x_2$  найдем как  $g(x_1)$ :

$$x_2 = g(x_1) = \frac{\exp(-0.4917) - 3}{6} = -0.3981$$

- Аналогичным образом находим

$$x_4 = -0.3869; x_5 = -0.3868 \text{ и } x_6 = -0.3868.$$

*Поскольку разность  $x_5$  и  $x_6$  по модулю меньше заданной точности, то принимаем, что решением является значение  $x = -0.3868$ .*



# РЕАЛИЗАЦИЯ

```
function EquivalentFunc(x: real): real;
begin
    result := (exp(x) - 3) / 6
end;

function Iterations(f: function(x: real): real; a, eps: real): real;
begin
    var x: real;
    result := f(a);

    repeat
        x := f(result);
        result := f(x);
    until abs(result - x) < eps
end;

begin
    Iterations(EquivalentFunc, -3, 1e-4).Print
end.
```

-0.386795147605899

# МЕТОД НЬЮТОНА (КАСАТЕЛЬНЫХ)

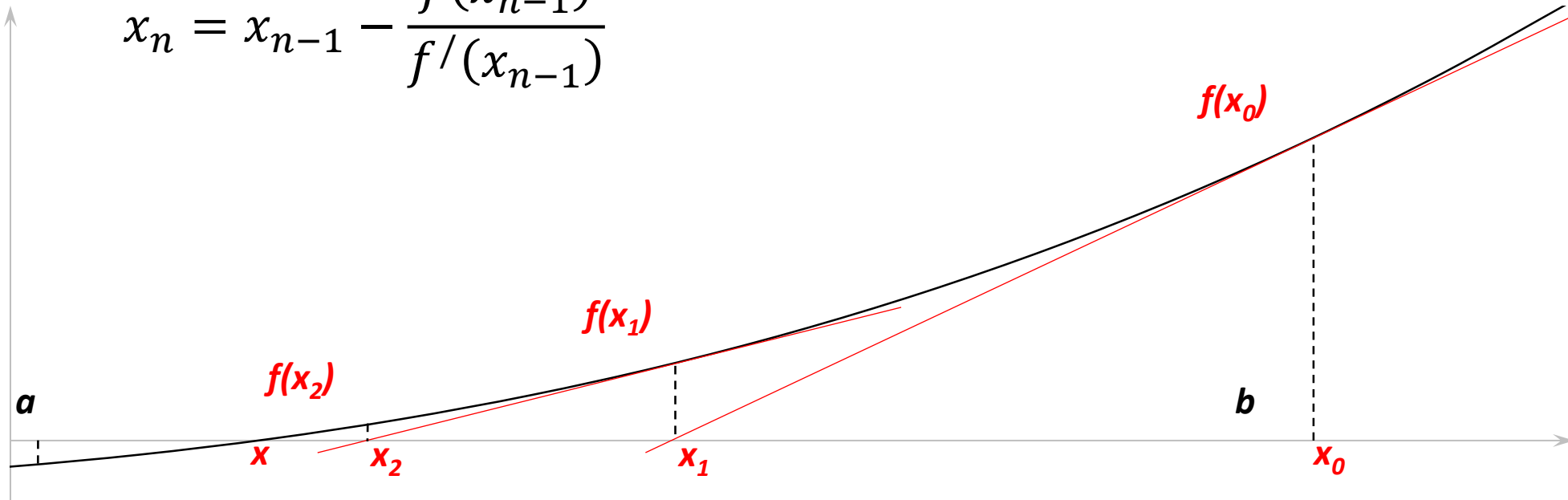
- Последовательность приближений строится следующим образом:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

.....

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$



- Выбор начальных приближений:

- если  $f(a) * f''(a) > 0$ , то точка  $a$ ;
- если  $f(b) * f''(b) > 0$ , то точка  $b$ ;

*Если неравенства не выполняются – метод не применим.*

# РЕАЛИЗАЦИЯ

```
function equation(x: real): real;  
begin  
    result := exp(x) - 6 * x - 3  
end;
```

```
function equation1(x: real): real;  
begin  
    result := exp(x) - 6  
end;
```

```
function equation2(x: real): real;  
begin  
    result := exp(x)  
end;
```

```
function Newton(f, f1, f2: function(x: real): real;
               a, b, eps: real): real;
begin
  var x: real;
  if f(a) * f2(a) > 0 then x := a
  else
    if f(b) * f2(b) > 0 then x := b
    else
      begin
        Print('Newton Method, Not Solved!');
        exit
      end;

  repeat
    x := result - f(result) / f1(result);
    result := x - f(x) / f1(x);
  until abs(result - x) < eps
end;

begin
  Newton(equation, equation1, equation2, -3, 1, 1e-4).Print
end.
```

-0.386794939318652

# Задание

*Составьте программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона:*

1.	$x^4 + 3 \cdot x - 20 = 0$	Интервал [1; 2], допустимая точность $10^{-2}$
2.	$e^x + x - 2 = 0$	Интервал [0; 1], допустимая точность $10^{-3}$
3.	$\ln(x) + x = 0$	Интервал [0.5; 1.5], допустимая точность $0.2 \cdot 10^{-4}$
4.	$2 \cdot x - e^{-0.1 \cdot x} = 0$	Интервал [0.2; 1.5], допустимая точность $0.5 \cdot 10^{-4}$





# КОНТАКТНАЯ ИНФОРМАЦИЯ


---

**ЧУЗЛОВ ВЯЧЕСЛАВ АЛЕКСЕЕВИЧ**

к.т.н., доцент ОХИ ИШПР

 Учебный корпус №2, ауд. 136

 +7-962-782-66-15

 [chuva@tpu.ru](mailto:chuva@tpu.ru)