ПРОДАКШН

ПРЕСЕНТ

«МЕТОД СЕКУЩИХ»

powered by

Prokof’ev Anton

Sizyh Danil

Slastnikova Anna

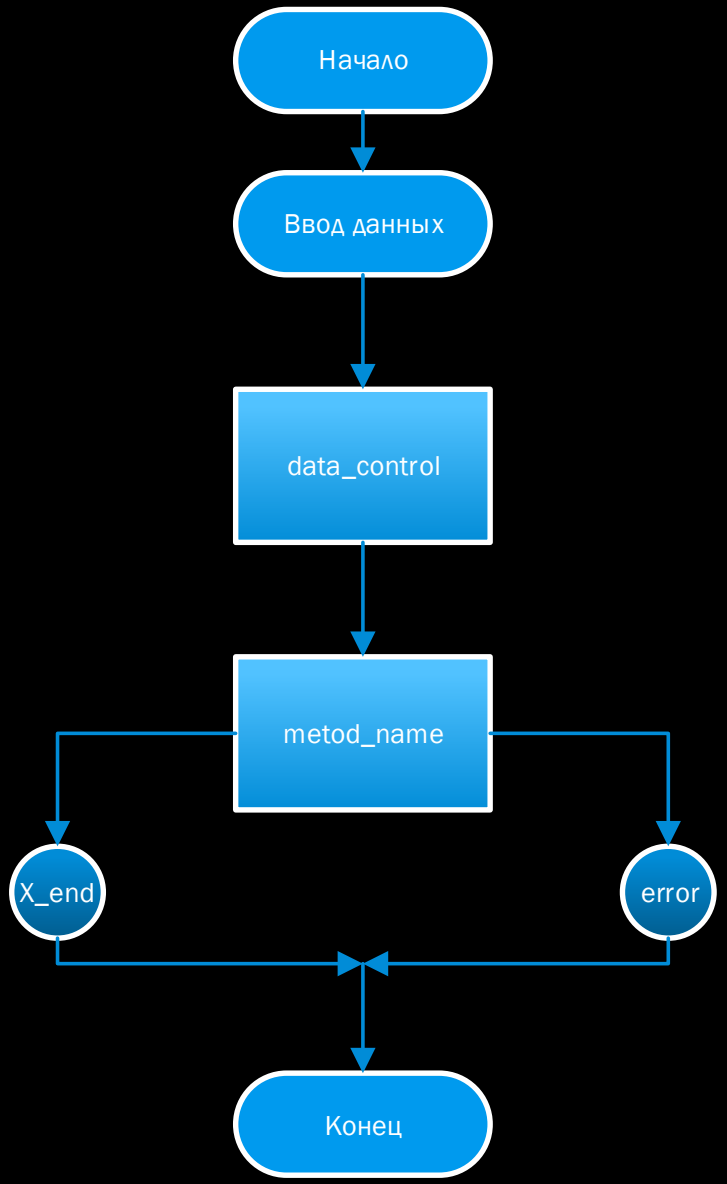
ЗАДАЧА:

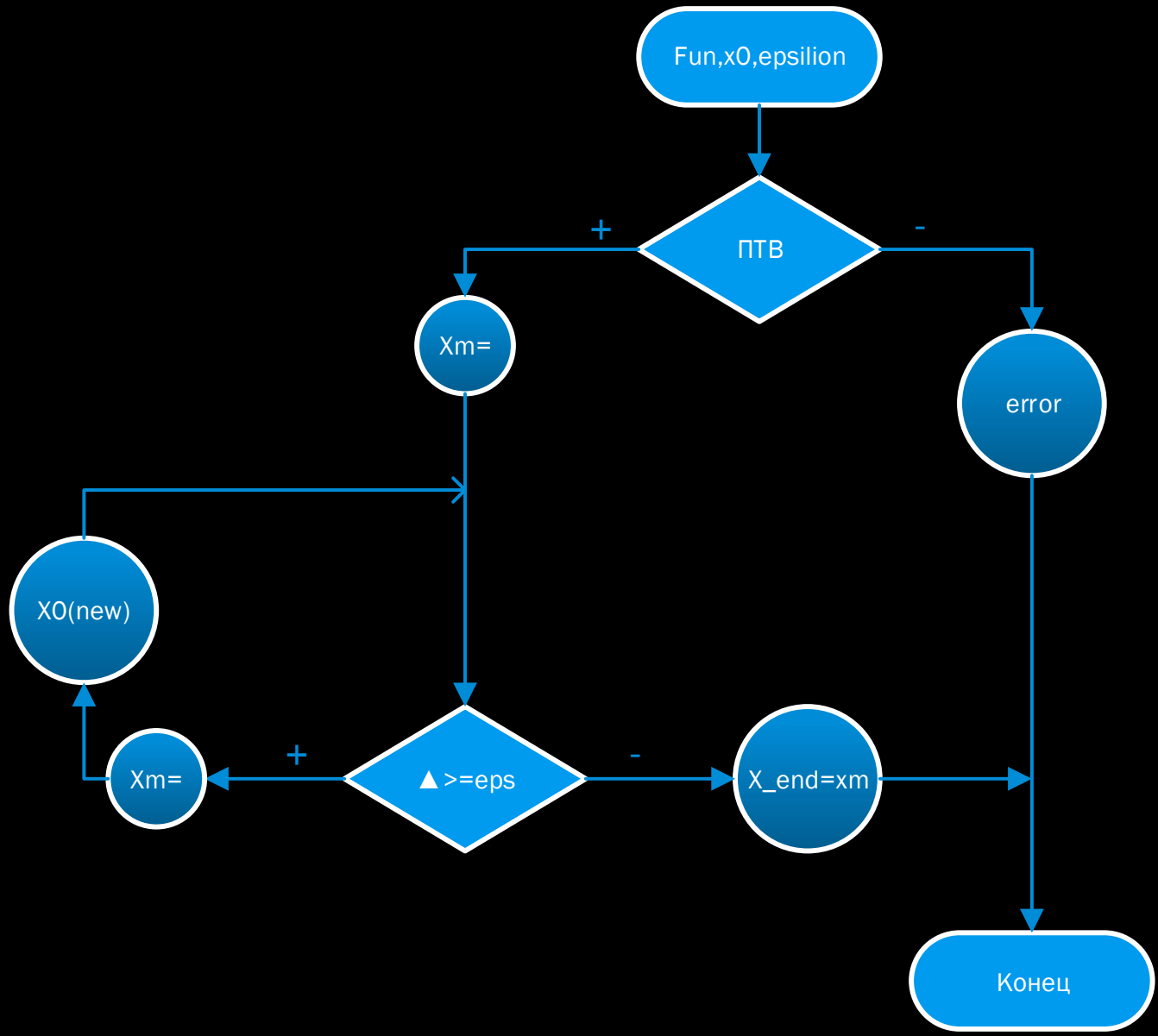
Найти корень уравнения на отрезке [a;b] при помощи метода секущих (второй модификации метода Ньютона).

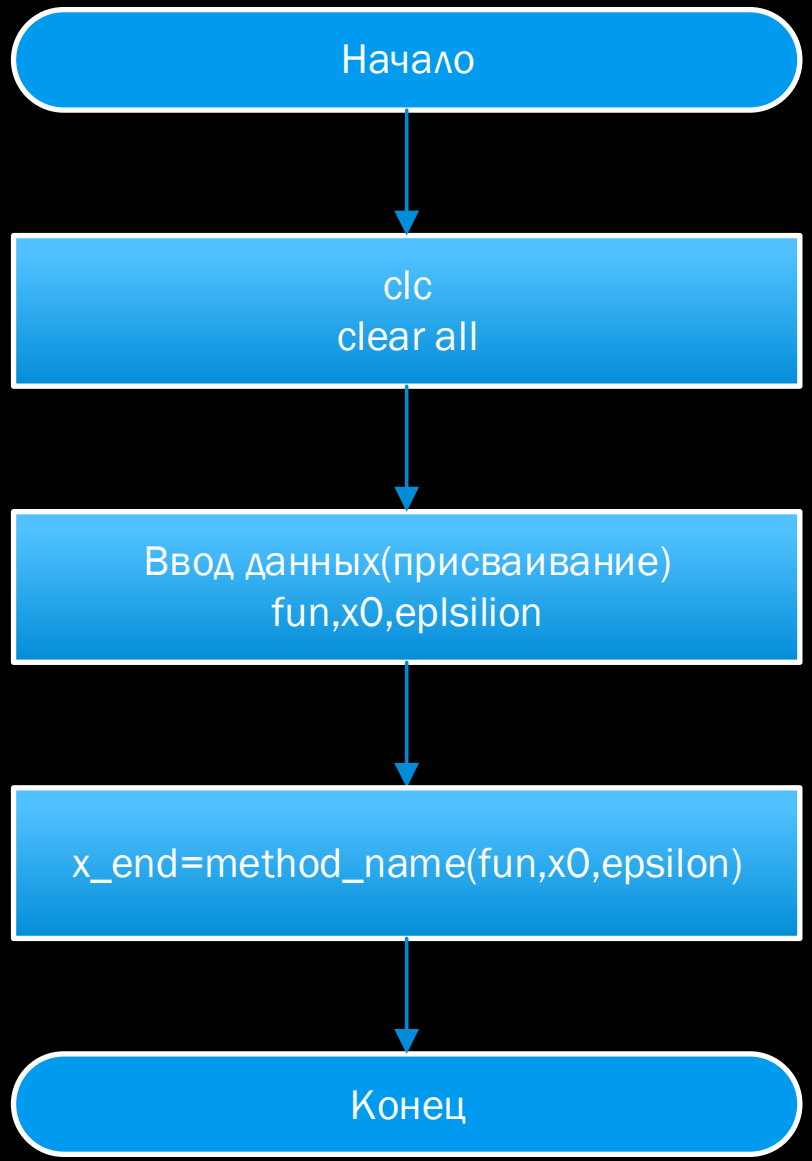
ОСНОВНЫЕ ПОНЯТИЯ:

- Теория Вейерштрасса
- Производная функции
- Секущая

ОПРЕДЕЛЕНИЕ МЕТОДА

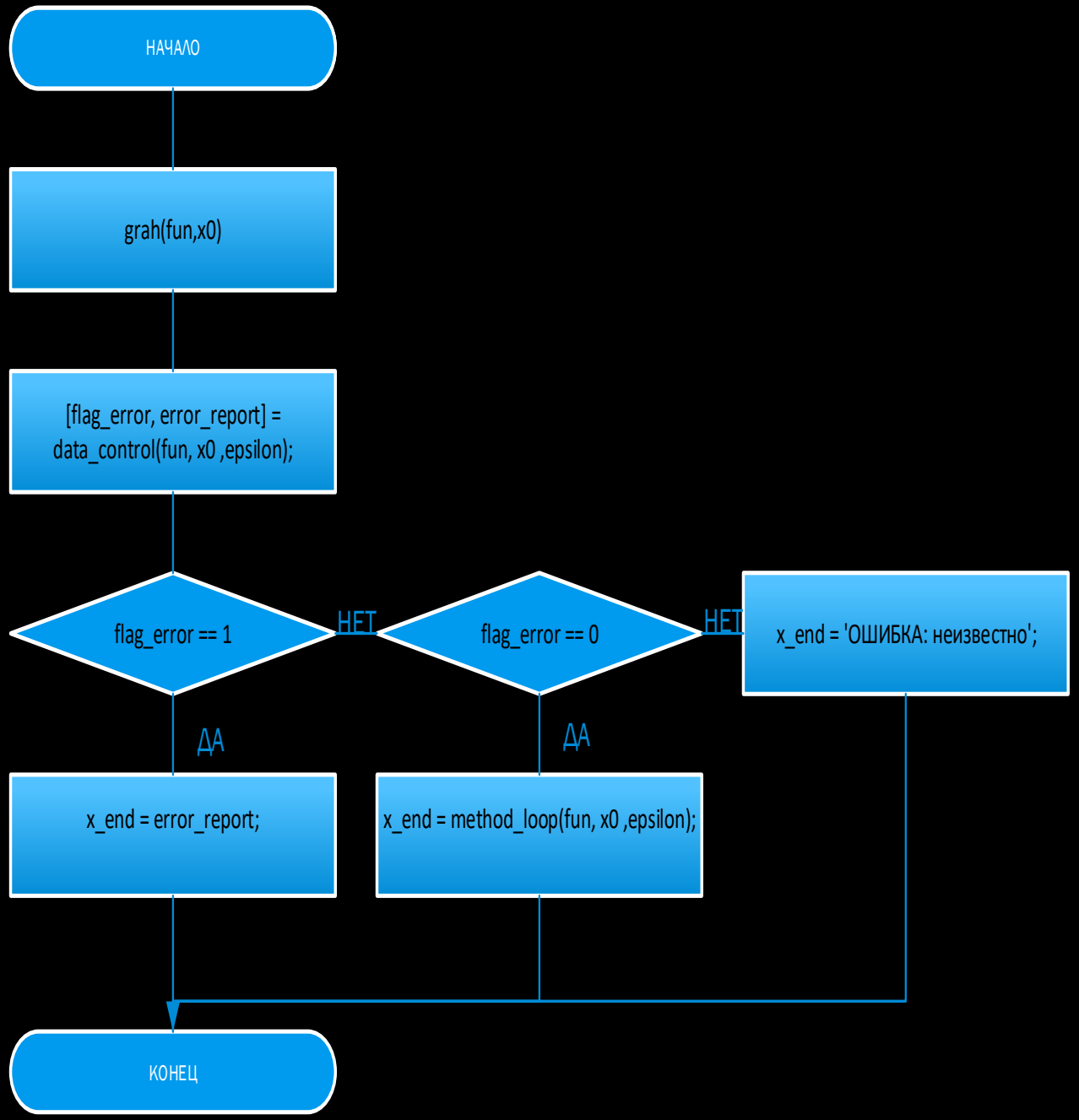

Начало
Ввод данных
data_control
metod_name
X_end
error
Конец


Fun,x0,epsilion
ПТВ
+
-
Xm=
error
X0(new)
Xm=
+
▲>=eps
-
X_end=xm
Конец



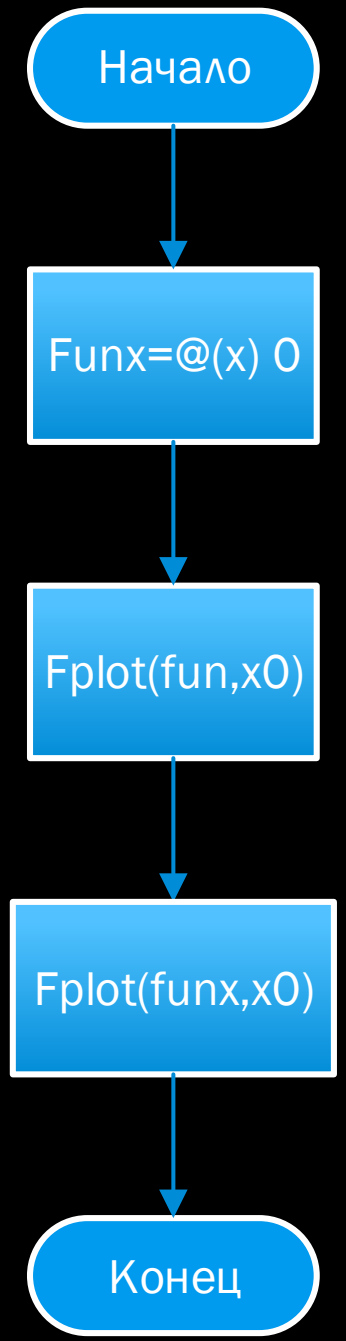
main.m

```
clc
clear ALL
%ввод данных
fun = @(x) x.^2-0.1;
x0 = [0,1];
epsilon=1e-2;
%нахождение приближенного корня
x_end = method_name(fun, x0 ,epsilon)
```



method_name.m

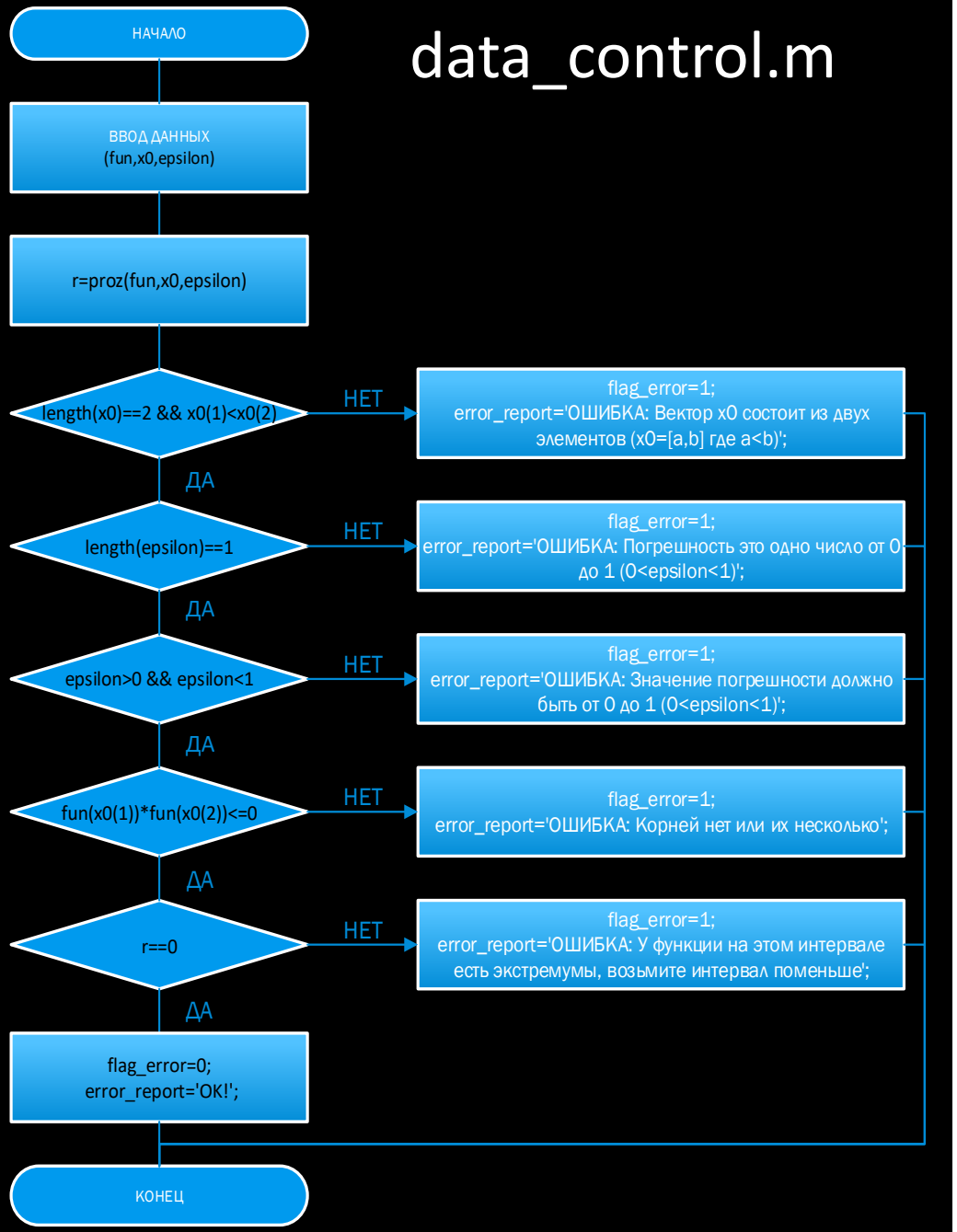
```
function x_end = method_name(fun, x0 ,epsilon)
 %построение графика для наглядности
 grah(fun,x0)
 %вызов первой проверки
 [flag_error, error_report] = data_control(fun, x0 ,epsilon);
 %основное тело
 if flag_error == 1
 %вместо результата выводим ошибку
 x_end = error_report;
 elseif flag_error == 0
 %а здесь результат
 x_end = method_loop(fun, x0 ,epsilon);
 else
 x_end = 'ОШИБКА: неизвестно';
 end
end
```



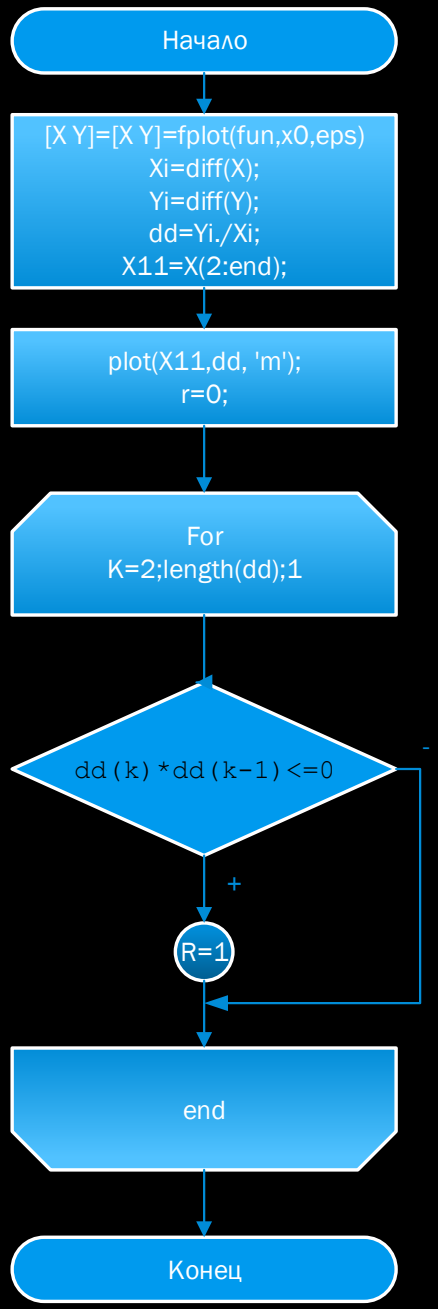
grah.m

```
function grah(fun,x0)
%функция для оси OX
funx = @(x) 0;
%построение графика на заданном участке
fplot(fun,x0,'b')
grid on
grid minor
hold on
%построение оси OX
fplot(funx,x0,'r')
title('График для наглядности')
end
```

data_control.m



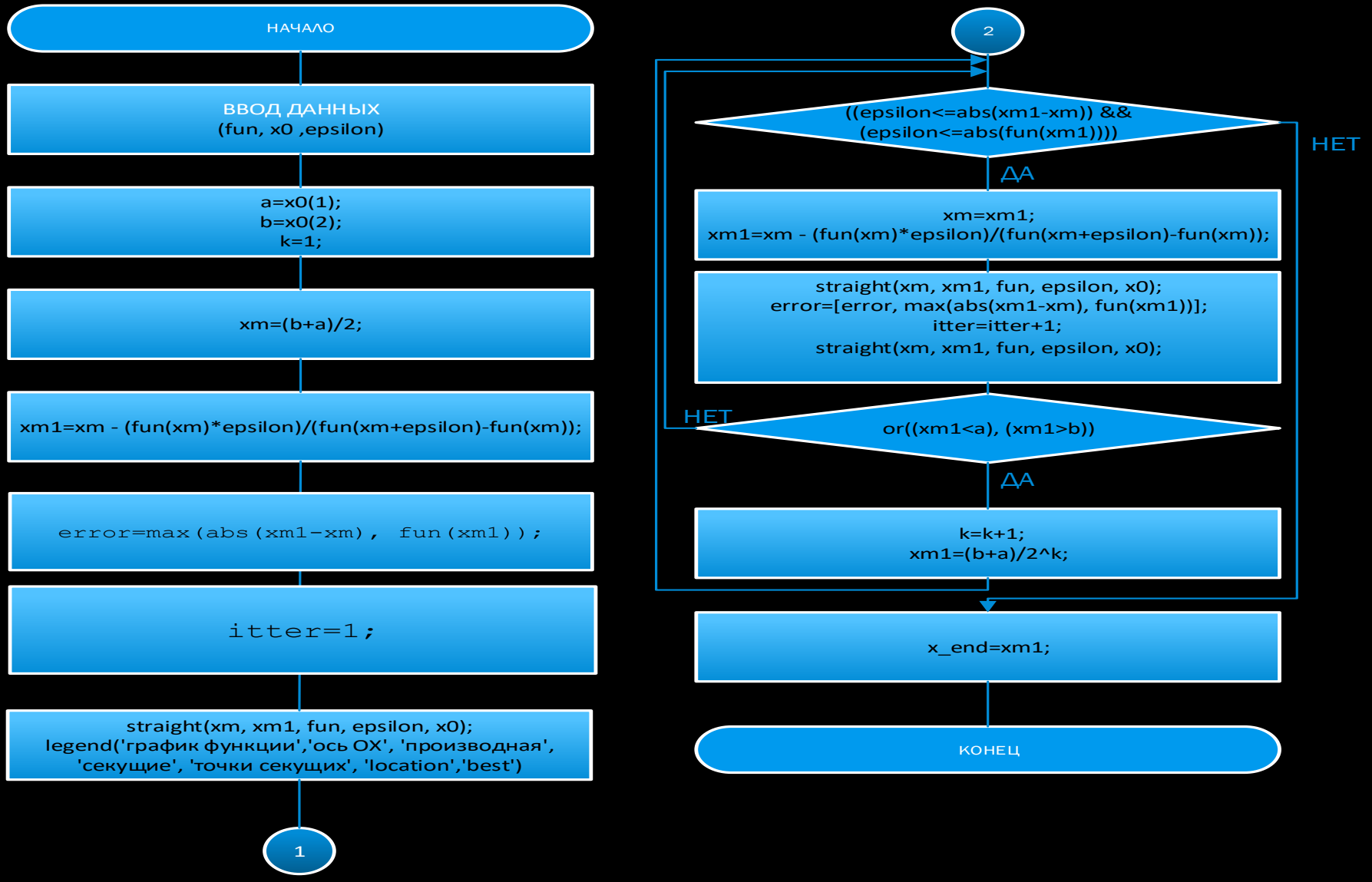

```
function [flag_error, error_report] = data_control(fun, x0 ,epsilon)
%проверка на "дурака"
r=proz(fun,x0,epsilon);
if length(x0)==2 && x0(1)<x0(2)
    if length(epsilon)==1
        if epsilon>0 && epsilon<1
            %проверка Вейерштрасса
            if fun(x0(1))*fun(x0(2))<=0
                if r==0
                    flag_error=0;
                    error_report='OK!';
                else
                    flag_error=1;
                    error_report='ОШИБКА: У функции на этом интервале есть экстремумы, возьмите интервал поменьше';

                end
            else
                flag_error=1;
                error_report='ОШИБКА: Корней нет или их несколько';
            end
        else
             flag_error=1;
             error_report='ОШИБКА: Значение погрешности должно быть от 0 до 1 (0<epsilon<1)';
        end
    else
         flag_error=1;
         error_report='ОШИБКА: Погрешность это одно число от 0 до 1 (0<epsilon<1)';
    end
else
     flag_error=1;
     error_report='ОШИБКА: Вектор x0 состоит из двух элементов (x0=[a,b] где a<b)';
end
end
```



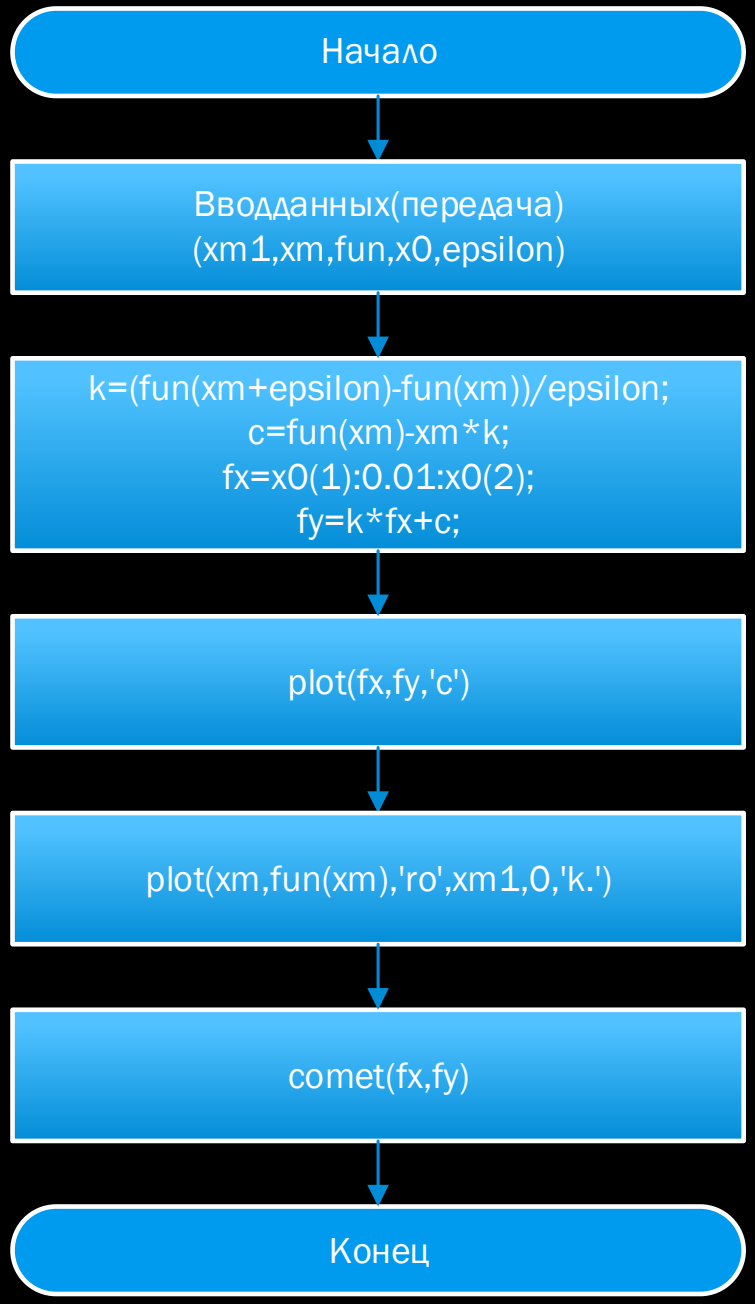
proz.m

```
function r=proz(fun,x0,epsilon)
[X Y]=fplot(fun,x0,eps);
Xi=diff(X);
Yi=diff(Y);
dd=Yi./Xi;
X11=X(2:end);
plot(X11,dd, 'm');
r=0;
for k=2:length(dd)
    if dd(k)*dd(k-1)<=0
        r=1;
    end
end
```

method_loop.m



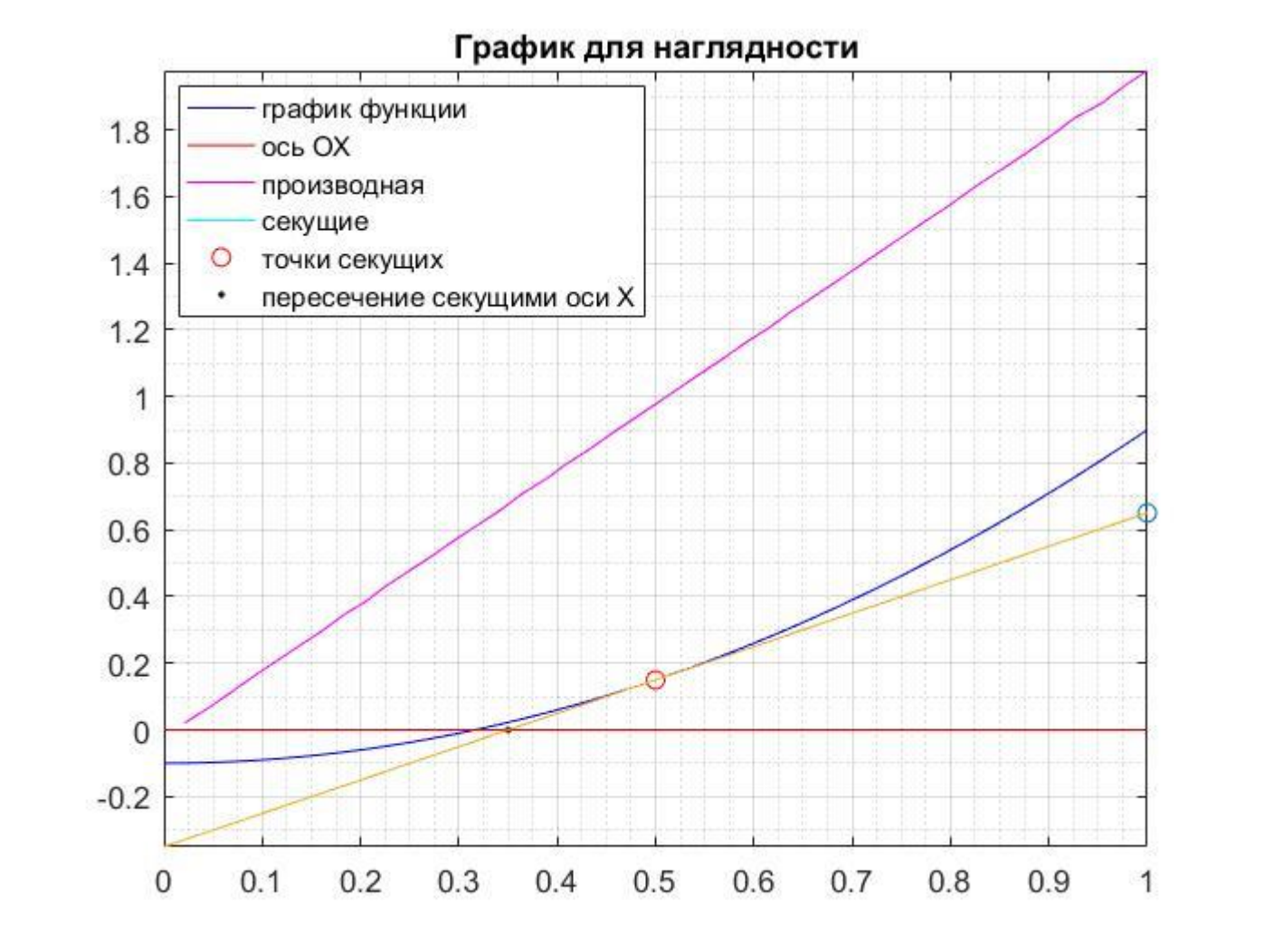
```
function x_end = method_loop(fun, x0 ,epsilon)
%переменные для упрощения расчетов
  a=x0(1);
  b=x0(2);
  k=1;
  xm=(b+a)/2;
  xm1=xm - (fun(xm)*epsilon)/(fun(xm+epsilon)-fun(xm));
  error=max(abs(xm1-xm), fun(xm1));
  itter=1;
  straight(xm, xm1, fun, epsilon, x0);
  legend('график функции','ось OX', 'производная', 'секущие','точки секущих','пересечение секущими оси X','location','best')
  %цикл с расчетами
while ((epsilon<=abs(xm1-xm)) && (epsilon<=abs(fun(xm1))))%здесь идет проверка дельта икс и дельта игрек
   xm=xm1;
   xm1=xm - (fun(xm)*epsilon)/(fun(xm+epsilon)-fun(xm));
   straight(xm, xm1, fun, epsilon, x0);
   figure
   grah(fun,x0);
   proz(fun,x0,epsilon);
   error=[error, max(abs(xm1-xm), fun(xm1))];
   itter=itter+1;
   straight(xm, xm1, fun, epsilon, x0);
   legend('график функции','ось OX', 'производная', 'секущие','точки секущих','пересечение секущими оси X','location','best')
    if or((xm1<a), (xm1>b))
        k=k+1;
        xm1=(b+a)/2^k;
    end
end
figure
loglog([1:itter], error);
grid on
x_end=xm1;
end
```



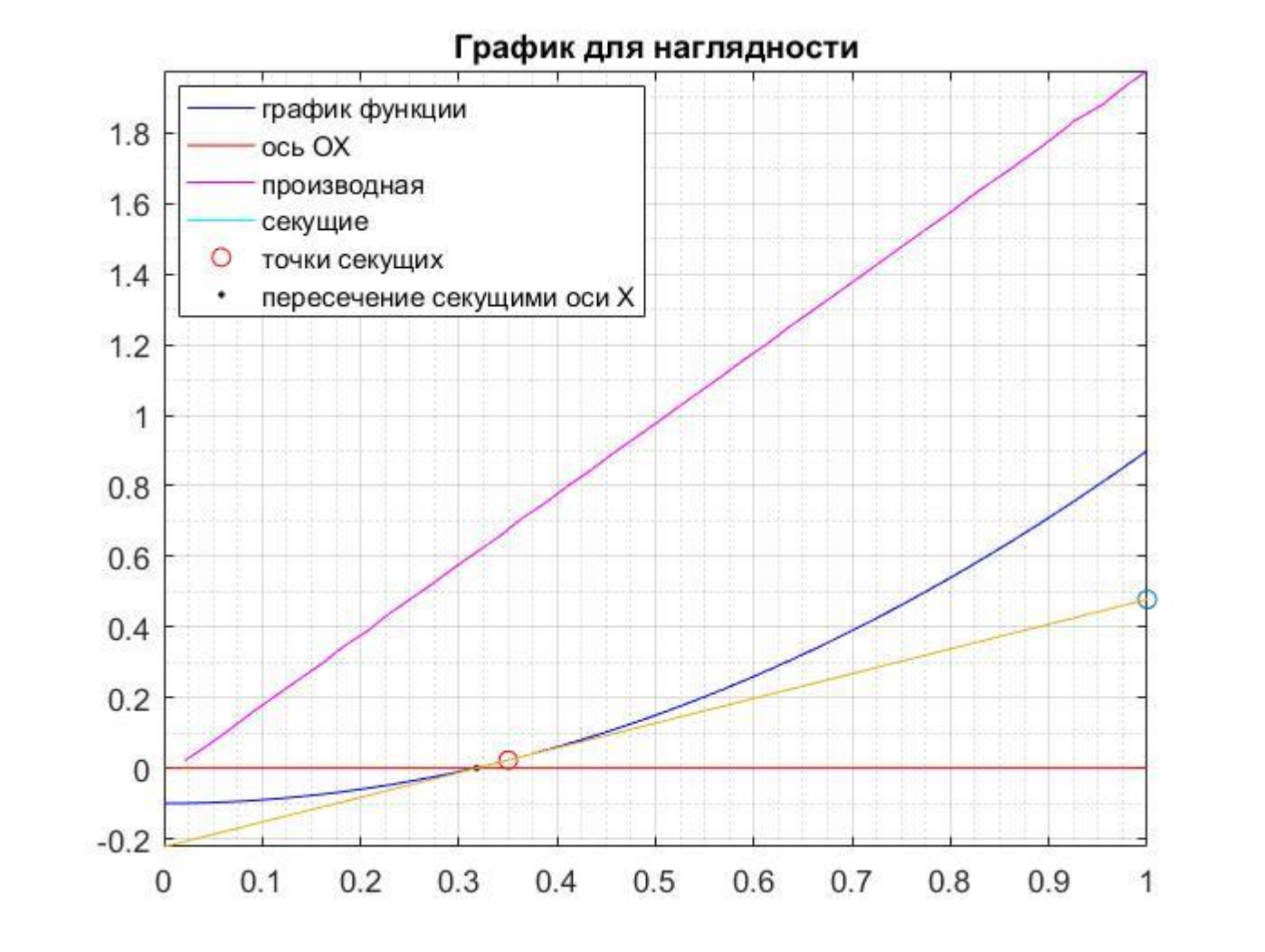
straight.m

```
function straight(xm, xm1, fun, epsilon, x0)
    k=(fun(xm+epsilon)-fun(xm))/epsilon;
    c=fun(xm)-xm*k;
    fx=x0(1):0.01:x0(2);
    fy=k*fx+c;
    plot(fx,fy, 'c' )
    plot(xm, fun(xm),'ro',xm1,0,'k.')
    comet(fx, fy)
end
```

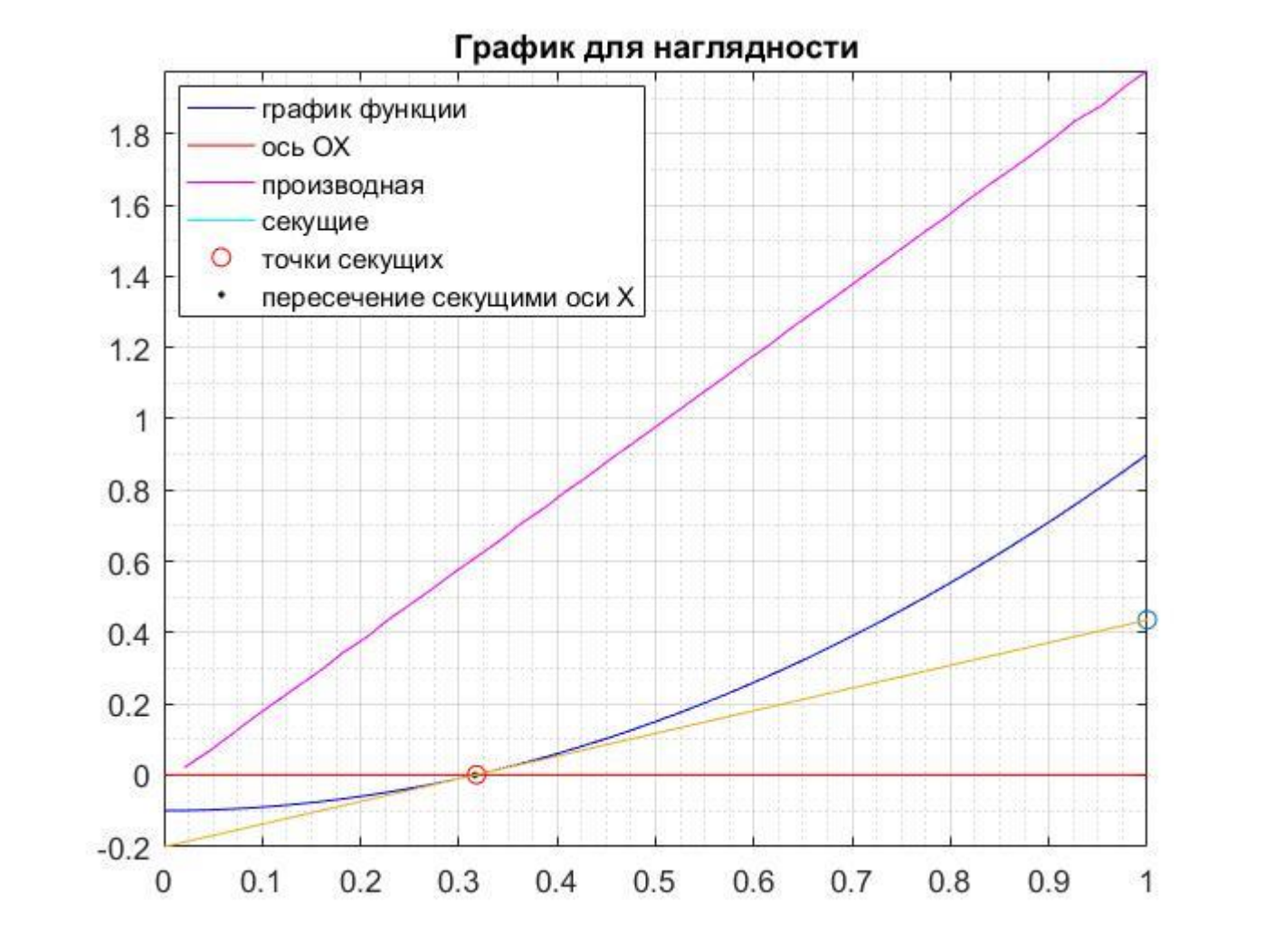
Первая итерация



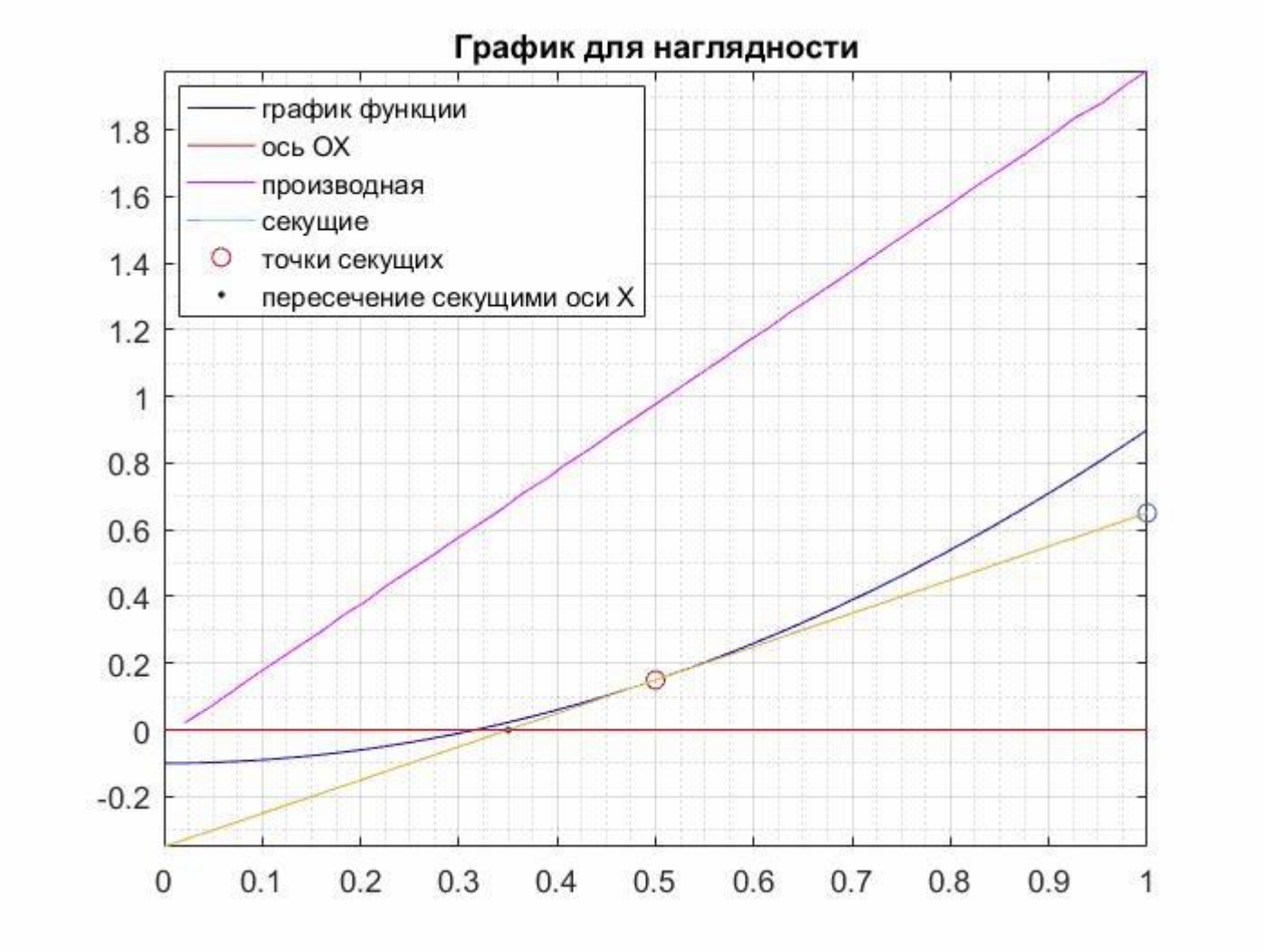
Вторая итерация



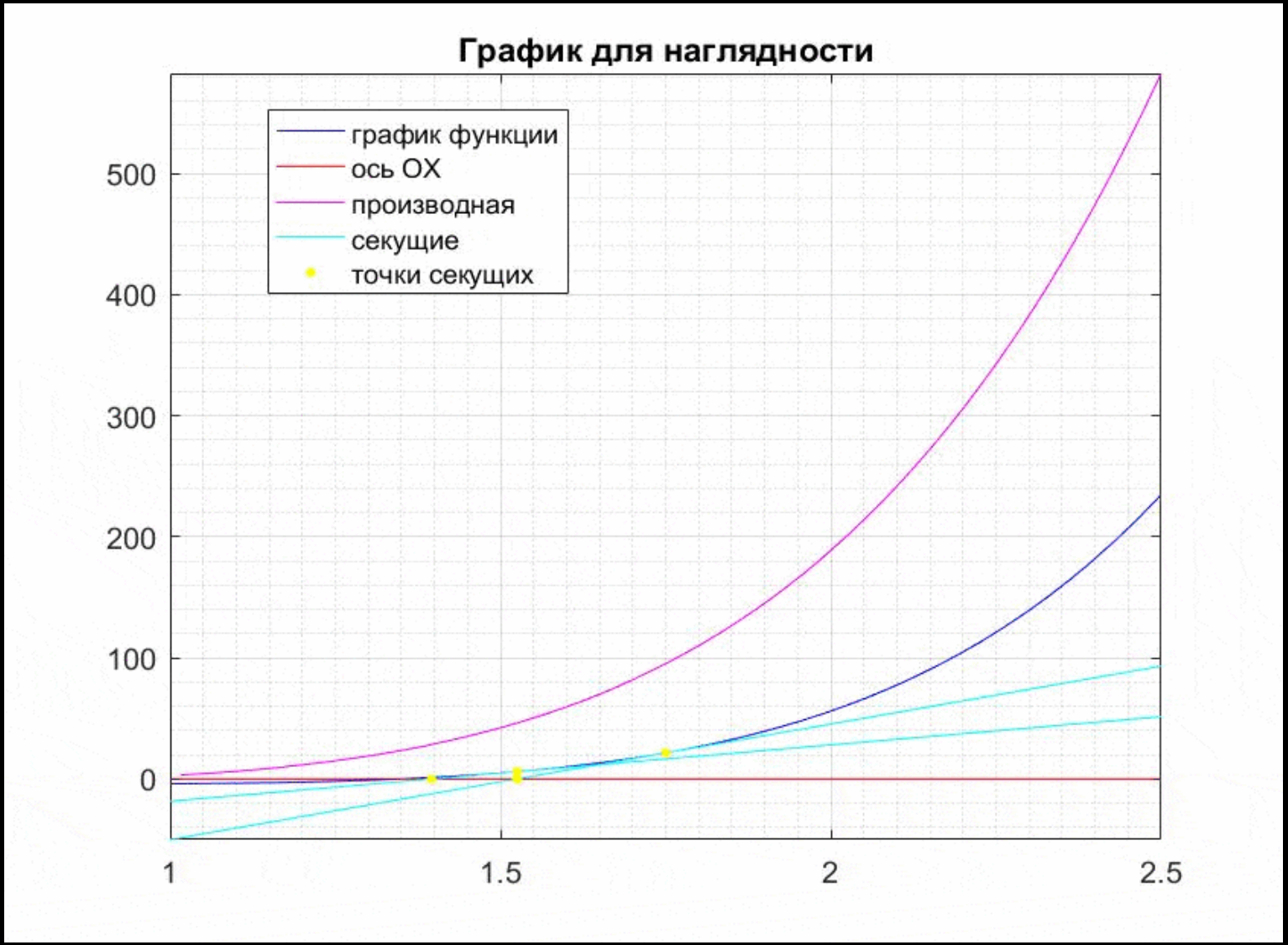
Третья итерация



Gif итерационного процесса



Gif итерационного процесса



fun=x^6-3x-2
[1;2.5]
epsilion=1e-4

ЗЕ ЭНД