```python
import numpy as np
from matplotlib import pyplot as plt

def f1(x):
    return x**5 - 6*x**3 + np.pi*x - 1

def derivf1(x):
    return 5*x**4 - 18*x**2 + np.pi

def f2(x):
    return np.exp(-x/3) + x**4 - 7*(x**2)*np.log(x)

def derivf2(x):
    return (-1/3)*np.exp(-x/3) + 4*x**3 - 14*x*np.log(x) + 7*x

def f3(x):
    return x**12 + 2**x - 2.5

def derivf3(x):
    return 12*x**11 + np.log(2)*2**x

def ROOT_FALSEPOSITION(b,a,f,derivf,err):
    #Find point where the line between the two bounds crosses x axis
    c = b - ((f(b))*(a-b))/((f(a))-(f(b)))
    iteration = 0
    clist = []
    rootlist=[]
    multiroot=[]
    #Start loop that depends on how close f(c) is to 0
    while np.abs(f(c)) > err:
        #Loop choosing to replace a or b with c value depending on
        #which side has both a positive and negative f(x)
        if f(a)*f(b) < 0:
            b=c
        else:
            a=c
        #Create a list containing all of the c values to analyze later
        clist = np.append(clist,c)
        c = b - ((f(b))*(a-b))/((f(a))-(f(b)))
        iteration += 1
    rootlist=np.append(rootlist,c)
    if np.abs(derivf(c)) <= err:
        multiroot=np.append(multiroot,c)
    iterations = np.arange(1,iteration+1,1)
    rel_err = np.abs(clist/c - 1)*100

    plt.figure(plotloop)
    plt.subplot(211)
    plt.plot(iterations, clist, 'bo--')
    plt.xlabel("Number of Iterations", fontsize=10, fontstyle='italic')
    plt.ylabel("Estimate of Root", fontsize=10, fontstyle='italic')
    plt.title("How False Position Method Approaches Root {}".format(plotloop),fontsize=14)
    plt.subplots_adjust(hspace = .7)

    plt.subplot(212)
    plt.plot(iterations, rel_err, 'ro--')
    plt.xlabel("Number of Iterations", fontsize=10, fontstyle='italic')
    plt.ylabel("Relative Error", fontsize=10, fontstyle='italic')
    plt.title("Relative Error Progression",fontsize=14)
    plt.show()
    plt.savefig("function4_r{}".format(plotloop))
    return rootlist, multiroot, iteration

def manyintervals(a,b,f,derivf,err,inte):
    rootlist=[]
    multiroot=[]
    intsize=np.abs((b-a))/inte
    minimum=a
    maximum=a+intsize
    loopend=0
    global plotloop
    plotloop=1
    while loopend<inte:
        if f(minimum)*f(maximum)<0:
            r1, r2, iteration = ROOT_FALSEPOSITION(minimum,maximum,f,derivf,err)
            rootlist = np.append(rootlist, r1)
            isitmulti=np.size(r2)
            plotloop += 1
            if isitmulti>0:
                multiroot=np.append(r2)
        minimum=minimum+intsize
        maximum=maximum+intsize
        loopend=loopend+1
    print(rootlist, multiroot, iteration)

#manyintervals(1.5,2.5,f1,derivf1,0.0001,10) #root = 2.3467, iterations = 4
    #This sped up the iterations slightly
#manyintervals(0,3,f1,derivf1,0.0001,10) #root = 2.3467, iterations = 4
    #Greatly sped up the iterations by a factor of around 30
#manyintervals(0,3,f2,derivf2,0.0001,10) #Does not blow up,
    #Found roots at 1.3736 with 2 iterations, and 2.54314 with 6 iterations
#manyintervals(0,3,f3,derivf3,0.0001,10) #root = .95333 after 31 iterations
```