```
# -*- coding: utf-8 -*-
"""
Created on Wed Mar 22 12:28:16 2017

@author: willie
"""

# Exemple d'équation non linéaire à résoudre
def f(x):
    return (x-2)**2-1

#Recherche de la solution par la méthode de la fausse position
def fausse_position(f,a,b,epsilon):
    #on vérifie si une solution existe dans l'intervalle [a;b]
    if f(a)*f(b)>0:
        print("Il n'est pas sûr qu'il y ait une solution dans l'intervalle.")
        return None
    i=0 #initialisation compteur du nombre d'iteration

    #on note c la position de découpe
    distance = b-a #initialisation de la distance entre deux positions de découpe
    c=b #initialisation de la première position de découpe

    while abs(distance)>epsilon and i<1000:
    #on teste si la distance entre deux positions de découpe
    #est supérieure à la précision cherchée
    #et si le nombre d'iteration n'est pas trop grand
        c_prec=c #on sauvegarde la position de découpe précédente
        c=a-f(a)*(b-a)/(f(b)-f(a)) #on calcule la nouvelle position de découpe
        if f(a)*f(c)<0: #le zéro est dans l'intervalle [a;nouvelle position]
            print 'b'
            b=c #le nouvel intervalle [a;b] est donc [a;nouvelle position]
        else: #le zéro est dans l'intervalle [nouvelle position;b]
            a=c #le nouvel intervalle [a;b] est donc [nouvelle position;b]
        distance=c_prec-c #on recalcule la distance entre deux positions de découpe
        i=i+1 #on incémente le nombre d'iterations
    return i,c #on renvoie le nombre d'itérations total et la dernière position
calculée

#on affiche la dernière position calculée après appel de la fonction
print fausse_position(f,0.,2.,1e-12)
```