```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy as ipw
from scipy import integrate
```

✓ 7.8s

```
alpha = 1 #Αύξηση θηραμάτων όταν οι θηρευτές απουσιάζουν
beta = 0.3 #Πως πεθαίνουν τα θηράματα λόγω των θηρευτών
delta = 0.8 #Μετά απο πόσα θηράματα που έχουν πιαστεί θα έχουμε νεό θηρευτή
gamma = 0.75 #Πως μειόνονται οι θηρευτές όταν απουσιάζουν τα θηράματα
k1=3
x0=3
y0=2
#Σε αυτό το σημείο ορίζουμε την συνάρτηση μας με όποια απο τα a,b,g,d,k1 έχουμε επιλέξει να χρησιμοποιήσουμε.
def derivative(X, t, alpha , beta, delta , gamma,k1):
        x,y = X
        dotx=x*(alpha*(1-x/k1) -beta*y)
        doty=y*(-gamma + delta*x)
        return np.array([dotx,doty])
```

✓ 0.6s

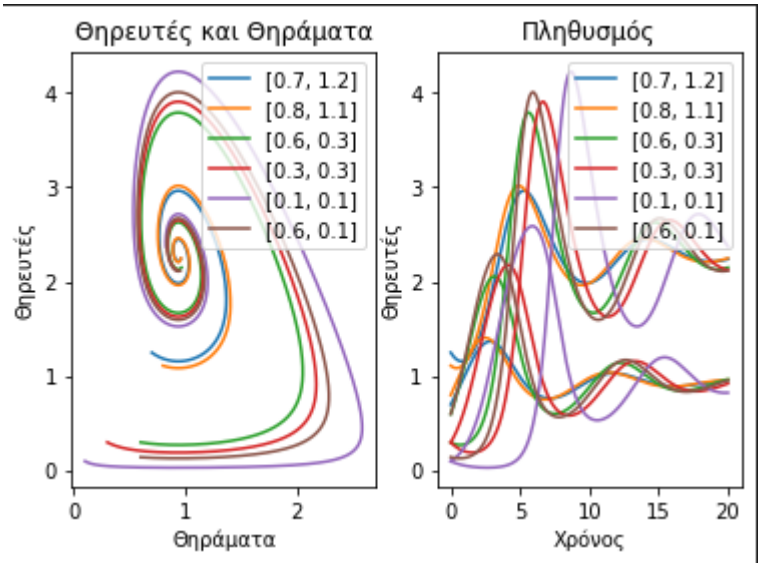
```
Nt=1000*5
tmax=20
t=np.linspace(0., tmax, Nt) #Αυτή την εντολή την χρησιμοποιούμε για να ορίσουμε τον αριθμό των δειγμάτων που στην συγκεκριμένη περίπτωση ξεκινάει απο 0
                            # φτάνει έως το tmax και παράγει Nt δείγματα , όσο λιγότερα δείγματα τόσο 'χειρότερα' θα βγούνε τα γραφήματα
X0=np.array([6,4])#Η αρχική συνθήκη αν θέλουμε να υπάρχει
res= integrate.odeint(derivative, X0, t, args=(alpha, beta , delta, gamma, k1))#Μέσω αυτής της συνάρτησης μπορούμε να λύσουμε την διαφορική που έχουμε ορίσει παραπάνω
```

✓ 0.5s

```
def yxyt(t,init, ax):     #t ο χρόνος , init θα είναι οι αρχίκες συνθήκες και το ax θα το χρησιμοποιήσουμε για να φτιάξουμε δυο διαγράμματα
    res=integrate.odeint(derivative,init, t, args=(alpha, beta , delta, gamma, k1)) #η συναρτηση που μας βοηθάει να λύσουμε την διαφορική που έχουμε ορίσει
    ax[0].plot(res[:,0], res[:,1], label='[{:.1f}, {:.1f}]'.format(*init))  #θα φτιάξουμε 2 διαγράμματα , στο πρώτο δηλαδή στο ax[0] θα εχουμε τους θηρευτες vs τα θηράματα
    l1, = ax[1].plot(t, res[:,0], label='[{:.1f}, {:.1f}]'.format(*init))
    ax[1].plot(t, res[:,1], color=l1.get_color())                   #στο δεύτερο διάγραμμα έχουμε το y με t που βλέπουμε πως εξελίσσεται το πλήθος

fh,ax=plt.subplots(1,2)
yxyt(t, [.7, 1/.8], ax)       #εδώ θέτουμε τις αρχικές συνθήκες , σε κάθε ενα διαφορετικές
yxyt(t, [.8, 1/.9], ax)
yxyt(t, [.6, .3], ax)
yxyt(t, [.3, .3], ax)
yxyt(t, [.1, .1], ax)
yxyt(t, [.6, .1/.7], ax)
ax[0].legend()                  #βάζοντας κάθε φορά διαφορετικό αριθμό στο ax , δηλαδή 0 ή 1 του λέμε σε πιο διάγραμμα θα κάνει τις αλλαγές που θέλουμε
ax[1].legend()
ax[0].set_xlabel('Θηράματα')
ax[0].set_ylabel('Θηρευτές')
ax[1].set_xlabel('Χρόνος')
ax[1].set_ylabel('Θηρευτές')
ax[0].set_title('Θηρευτές και Θηράματα')
ax[1].set_title('Πληθυσμός')
plt.show()
```

Αν βάλουμε στη συνάρτηση k>γ/δ έχουμε τα ακόλουθα διαγράμματα :



Ενώ για k>γ/δ ή k=γ/δ παίρνουμε :