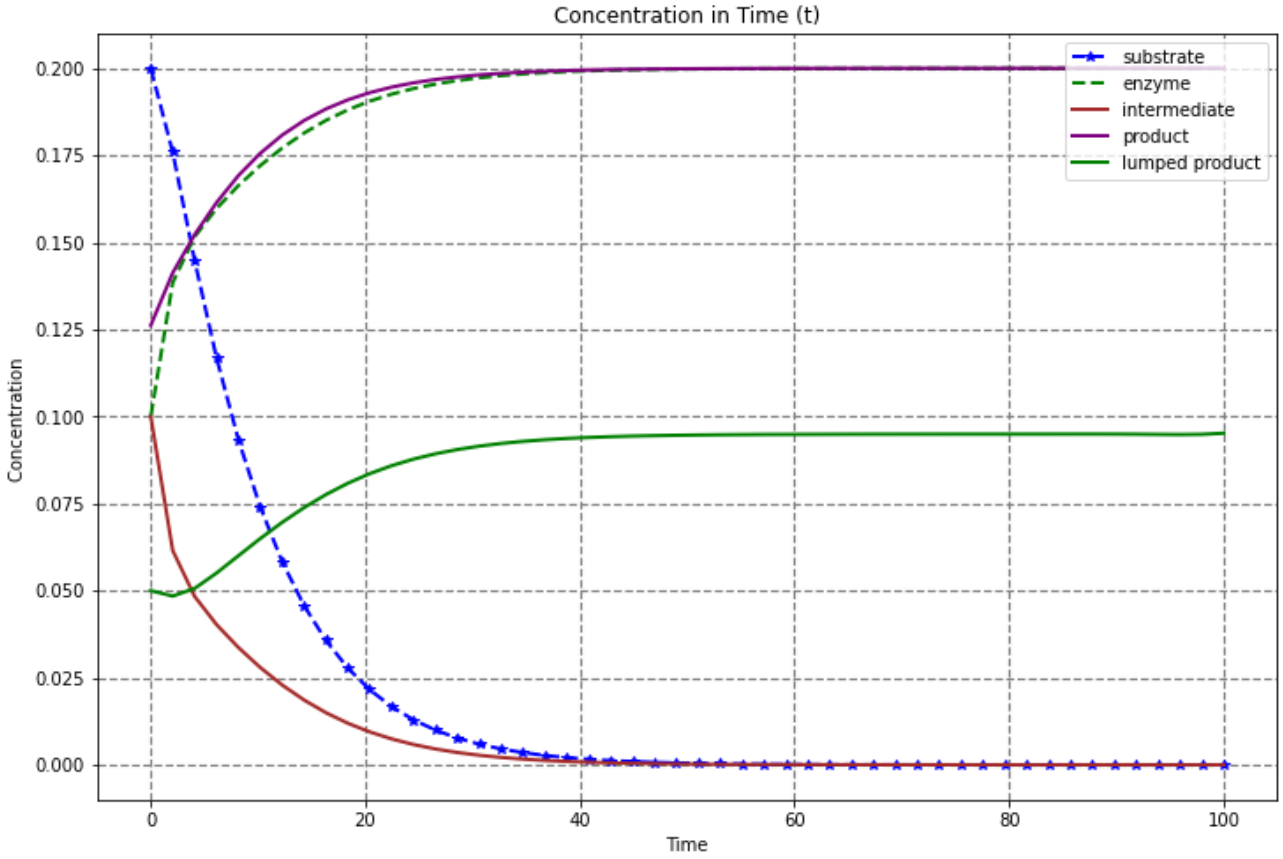
```
#hello
#sojor here,welcome!!!
#lets solve this pde functions in python --->...

#first we begin by importing important libraries
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

def ode_func_using_odeint(y,t):

    #defining the variables
    k0=1
    k1=0.15
    k2=0.4

    #assigning each ODE a vector element (Conversion from scalar to vector components)
    A=y[0]
    B=y[1]
    C=y[2]

    #derivertives
    dSdt=-k0*A*B+k1*C
    dEdt=-k0*A*B+k1*C+k2*C
    dIdt=k0*A*B-k1*C-k2*C
    dPdt=k2*C

    return np.array([dSdt,dEdt,dIdt,dPdt])

#initializing any initial conditions t0 test thefunction

#setting the initial conditions
y0=[0.2,0.1,0.1,0.1]

t=np.linspace(0,100) #setting the kinetic time frame

#solving the ODE using ODEINT function
sol=odeint(ode_func_using_odeint,y0,t)
A=sol[:,0]
B=sol[:,1]
C=sol[:,2]
D=B**A/5 #product  ***this is subject to change based on your specifications
E=C*B/A #lumped product  ***this is subject to change based on your specifications

# you can plot the function for the 1st,2nd,3rd and 4th order derivertives soln.
plt.figure(figsize=(12,8))
plt.plot(t,A, '--*',linewidth = 2,color='blue',label='substrate')
plt.plot(t,B, '--',linewidth = 2,color='green', label='enzyme')
plt.plot(t,C, '-',linewidth = 2,color='brown', label='intermediate')
plt.plot(t,D, '-',linewidth = 2,color='purple', label='product')
plt.plot(t,E, '-', linewidth = 2,color='green',label='lumped product')
plt.title('Concentration in Time (t)')
plt.xlabel('Time')
plt.ylabel('Concentration ')
```

```
plt.legend(loc='upper right')
plt.grid(color = 'grey', linestyle = '--', linewidth = 1.2)
plt.show
```

```
<function matplotlib.pyplot.show>
```



✓ 0s completed at 2:19 PM