<u>Question 2:</u>

8.1

Let [X] denotes the concentration of species X. We have

$$\frac{\mathrm{d}}{\mathrm{d}t}[E] = -k_1[E][S] + (k_2 + k_3)[ES]$$

$$\frac{\mathrm{d}}{\mathrm{d}t}[S] = -k_1[E][S] + k_2[ES]$$

$$\frac{\mathrm{d}}{\mathrm{d}t}[ES] = k_1[E][S] - (k_2 + k_3)[ES]$$

$$\frac{\mathrm{d}}{\mathrm{d}t}[P] = k_3[ES]$$

8.2

```
## This code is implemented with Python

import numpy as np
import matplotlib.pyplot as plt


# The rate constants
k1 = 100
k2 = 600
k3 = 150

# The initial concentrations
E = 1
S = 10
ES = 0
P = 0

# Time steps
t = np.linspace(0, 1, 1000)

# The rate of changes
def roc(concentrations, t):
    '''
    This function computes the rate of changes of the four species.
    :param concentrations: a list containing concentration of each species,
[E,S,ES,P]
    :param t: time steps
    :return: the rate of changes of the 4 species
    '''
    E, S, ES, P = concentrations
    dEdt = -k1*E*S + k2*ES + k3*ES
    dSdt = -k1*E*S + k2*ES
    dESdt = k1*E*S - (k2 + k3)*ES
    dPdt = k3*ES
    return np.array([dEdt, dSdt, dESdt, dPdt])

# The fourth-order Runge-Kutta method
def rk4(f, concentrations, t):
```

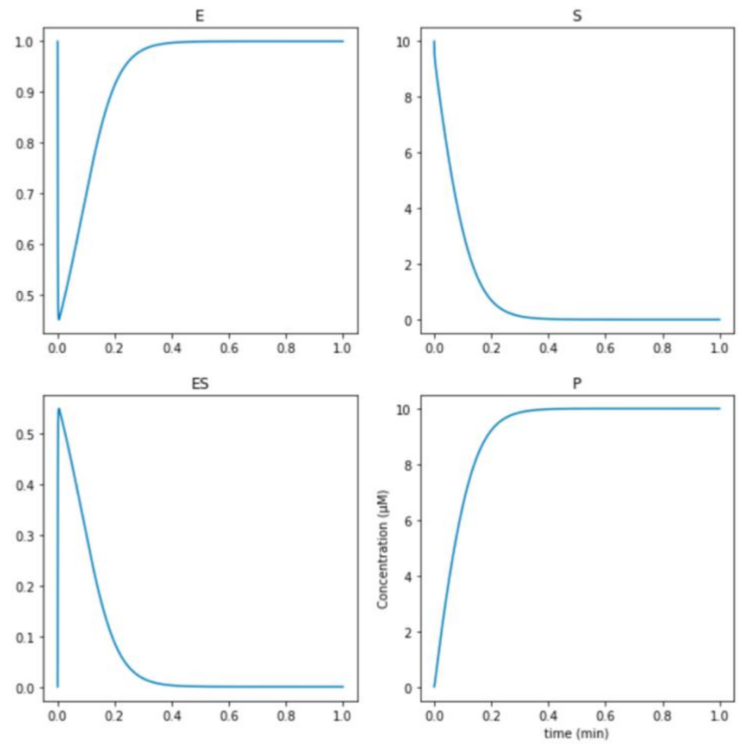
```python
    '''
    This function implements the fourth-order Runge-Kutta method
    :param f: the function
    :param concentrations: a list containing concentration of each species,
[E,S,ES,P]
    :param t: time steps
    :return: the results, [E_new,S_new,ES_new,P_new]
    '''
    n = len(t)
    y = np.zeros((n, len(concentrations)))
    v = np.zeros(n)
    y[0] = concentrations
    v[0] = 0
    for i in range(n-1):
        h = t[i+1] - t[i] #step
        k1 = h * f(y[i], t[i])
        k2 = h * f(y[i] + 0.5*k1, t[i] + 0.5*h)
        k3 = h * f(y[i] + 0.5*k2, t[i] + 0.5*h)
        k4 = h * f(y[i] + k3, t[i+1])
        y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
        v[i] = f(y[i], t[i])[3]
    return y,v

concentrations_new, v = rk4(roc, [E, S, ES, P], t) #new concentrations of
species at each time point

# Plot
fig, axs = plt.subplots(2, 2)
fig.set_size_inches(10,10)
plt.xlabel("time (min)")
plt.ylabel("Concentration (µM)")
# Plot E
axs[0, 0].plot(t, concentrations_new[:,0])
axs[0, 0].set_title('E')
# Plot S
axs[0, 1].plot(t, concentrations_new[:,1])
axs[0, 1].set_title('S')
# Plot ES
axs[1, 0].plot(t, concentrations_new[:,2])
axs[1, 0].set_title('ES')
# Plot P
axs[1, 1].plot(t, concentrations_new[:,3])
axs[1, 1].set_title('P')
plt.show()

print('E = ', concentrations_new[999,0], 'µM')
print('S = ', concentrations_new[999,1], 'µM')
print('ES = ', concentrations_new[999,2], 'µM')
print('P = ', concentrations_new[999,3], 'µM')
```

Program output:

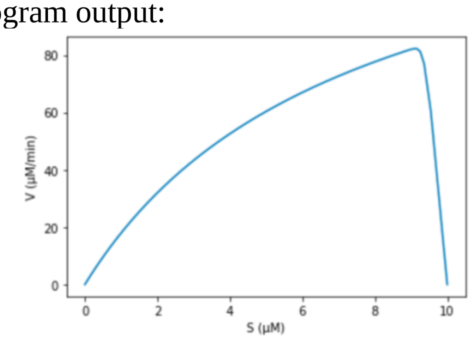



```
E =  0.9999999442267385 μM
S =  4.082438425947938e-07 μM
ES =  5.5773260053835604e-08 μM
P =  9.999999535982885 μM
```

8.3

```
plt.plot(concentrations_new[:,1], v)
plt.xlabel('S (μM)')
plt.ylabel('V (μM/min)')
plt.show()
print('V_max = ', max(v),'μM/min')
```

Program output:




```
V_max =  82.21714214034107 μM/min
```