Question 2

8.1

The law of mass action states that the rate of a chemical reaction is proportional to the product of the concentrations of the reactants raised to their stoichiometric coefficients. The rate of change of the concentration of species E, S, ES and P over time can be described by four differential equations, one for each species.

1. Rate of change of E:
$$\frac{d[E]}{dt} = -k1[E][S] + k2[ES] + k3[ES] = -k1[E][S] + (k2 + k3)[ES]$$
2. Rate of change of S:
$$\frac{d[S]}{dt} = -k1[E][S] + k2[ES]$$
3. Rate of change of ES:
$$\frac{d[ES]}{dt} = k1[E][S] - k2[ES] - k3[ES] = k1[E][S] - (k2 + k3)[ES]$$
4. Rate of change of P:
$$\frac{d[P]}{dt} = k3[ES]$$

where $\frac{d[X]}{dt}$ represents the rate of change of concentration of species X over time, k1, k2 and k3 are rate constants, [E], [S], [ES], and [P] are the concentrations of species E, S, ES and P respectively.

These equations describe the time evolution of the concentrations of the four species and can be used to model and simulate the enzyme reaction under different conditions, such as varying substrate and enzyme concentrations, and changes in the rate constants.

8.2

Code:

```
import numpy as np

def dEdt(E, S, ES, k1, k2):
    return -k1*E*S + k2*ES + k3*ES

def dSdt(E, S, ES, k1, k2):
    return -k1*E*S + k2*ES

def dESdt(E, S, ES, k1, k2, k3):
    return k1*E*S - k2*ES - k3*ES
```

```
def dPdt(ES, k3):
    return k3*ES

def runge_kutta(E, S, ES, P, k1, k2, k3, h, n):
    for i in range(n):
        kE1 = h*dEdt(E, S, ES, k1, k2)
        kS1 = h*dSdt(E, S, ES, k1, k2)
        kES1 = h*dESdt(E, S, ES, k1, k2, k3)
        kP1 = h*dPdt(ES, k3)

        kE2 = h*dEdt(E+kE1/2, S+kS1/2, ES+kES1/2, k1, k2)
        kS2 = h*dSdt(E+kE1/2, S+kS1/2, ES+kES1/2, k1, k2)
        kES2 = h*dESdt(E+kE1/2, S+kS1/2, ES+kES1/2, k1, k2, k3)
        kP2 = h*dPdt(ES+kES1/2, k3)

        kE3 = h*dEdt(E+kE2/2, S+kS2/2, ES+kES2/2, k1, k2)
        kS3 = h*dSdt(E+kE2/2, S+kS2/2, ES+kES2/2, k1, k2)
        kES3 = h*dESdt(E+kE2/2, S+kS2/2, ES+kES2/2, k1, k2, k3)
        kP3 = h*dPdt(ES+kES2/2, k3)

        kE4 = h*dEdt(E+kE3, S+kS3, ES+kES3, k1, k2)
        kS4 = h*dSdt(E+kE3, S+kS3, ES+kES3, k1, k2)
        kES4 = h*dESdt(E+kE3, S+kS3, ES+kES3, k1, k2, k3)
        kP4 = h*dPdt(ES+kES3, k3)

        E += (kE1 + 2*kE2 + 2*kE3 + kE4)/6
        S += (kS1 + 2*kS2 + 2*kS3 + kS4)/6
        ES += (kES1 + 2*kES2 + 2*kES3 + kES4)/6
        P += (kP1 + 2*kP2 + 2*kP3 + kP4)/6
    return E, S, ES, P

# Define initial concentrations and rate constants
E = 1
S = 10
ES = 0
P = 0
k1 = 100
k2 = 600
k3 = 150

# Define time step and number of iterations
h = 0.0005
n = 1000
```

```
# Call the Runge-Kutta function
E, S, ES, P = runge_kutta(E, S, ES, P, k1, k2, k3, h, n)

# Print the final concentrations
print("E: ", E)
print("S: ", S)
print("ES: ", ES)
print("P: ", P)
```

Output of the code:

E: 0.9995416746853101	S: 0.0033564141454787803
ES: 0.00045832531469139684	P: 9.996185260539821

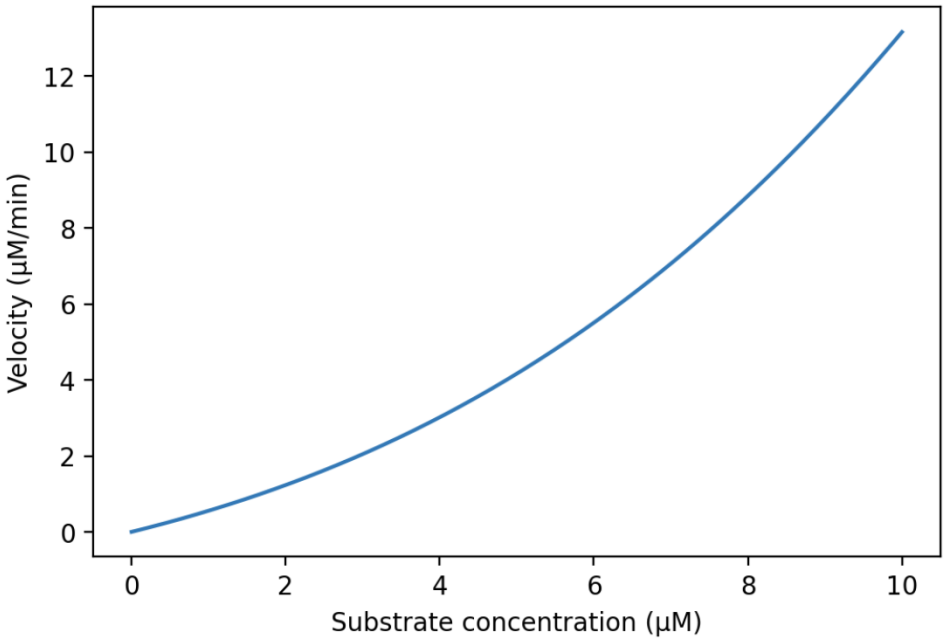
8.3:

Code:

```
import matplotlib.pyplot as plt

# Define initial concentrations and rate constants
E = 1
#S = 10
ES = 0
P = 0
k1 = 100
k2 = 600
k3 = 150

# Define time step and number of iterations
h = 0.0002
n = 1000

# Define an array of substrate concentrations to iterate over
S_range = np.linspace(0, 10, 200)

# Create empty arrays to store the velocities
velocities = []

# Iterate over substrate concentrations
for S in S_range:
    E, S, ES, P = runge_kutta(E, S, ES, P, k1, k2, k3, h, n)
    velocities.append(dPdt(ES, k3))

# Plot the velocity as a function of substrate concentration
plt.figure(dpi=100)
```

```
plt.plot(S_range, velocities)
plt.xlabel('Substrate concentration (µM)')
plt.ylabel('Velocity (µM/min)')
plt.show()

print("velocities", velocities[-1])
```

Plot:



Output of the code: 13.155167407464443