8.1

(1) $d[E]/dt = -k_1[E][S] + k_2[ES] + k_3[ES]$

(2) $d[S]/dt = -k_1[E][S] + k_2[ES]$

(3) $d[ES]/dt = k_1[E][S] – k_2[ES] – k_3[ES]$

(4) $d[P]/dt = k_3[ES]$

8.2

To use the Runge-Kutta method, the number of variables need to be reduced. The concentration of free enzyme can be expressed as (total enzyme – ES complex). [E] in equations 2 and 3 can be substituted with ($[E_0]$ – [ES]) which is [1 – ES]. Also, $k_1 = \frac{100/\mu M}{60s} = \frac{5}{3}\mu M/s$, $k_2 = \frac{600}{min} = \frac{10}{s}$, $k_3 = \frac{150}{min} = \frac{2.5}{s}$.

Therefore, equation 2 can be expressed as $d[S]/dt = -(5/3)(1 – [ES])[S] + 10[ES] \rightarrow f(t, x, y)$

Equation 3 can be expressed as $d[ES]/dt = (5/3)(1 – [ES])[S] - 12.5[ES] \rightarrow g(t, x, y)$

These are 2 simultaneous equations, with dependent variables [S] and [ES], and independent variable t. According to 4th order Runge Kutta method, $[S] = [S]_0 + k$ while $[ES] = [ES]_0 + l$. k and l represents $\frac{1}{6}(k_1+2k_2+2k_3+k_4)$ and $\frac{1}{6}(l_1+2l_2+2l_3+l_4)$ respectively, and the values of k_{1-4} and l_{1-4} can be derived by:

$k_1 = h*f(t, x, y)$

$l_1 = h*g(t, x, y)$

$k_2 = h*f(t + 0.5*h, x + 0.5*k_1, y + 0.5*l_1)$

$l_2 = h*g(t + 0.5*h, x + 0.5*k_1, y + 0.5*l_1)$

$k_3 = h*f(t + 0.5*h, x + 0.5*k_2, y + 0.5*l_2)$

$l_3 = h*g(t + 0.5*h, x + 0.5*k_2, y + 0.5*l_2)$

$k_4 = h*f(t + h, x + k_3, y + l_3)$

$l_4 = h*g(t + h, x + k_3, y + l_3)$

By letting step size be 0.02 and maximum time be 30s, the value of [S] and [ES] every 0.02 seconds up till 30s can be calculated.

As mentioned earlier, $[E] = [E_0] – [ES]$, so the value of [E] at every time point can be derived from 1 – [ES] at corresponding time points.

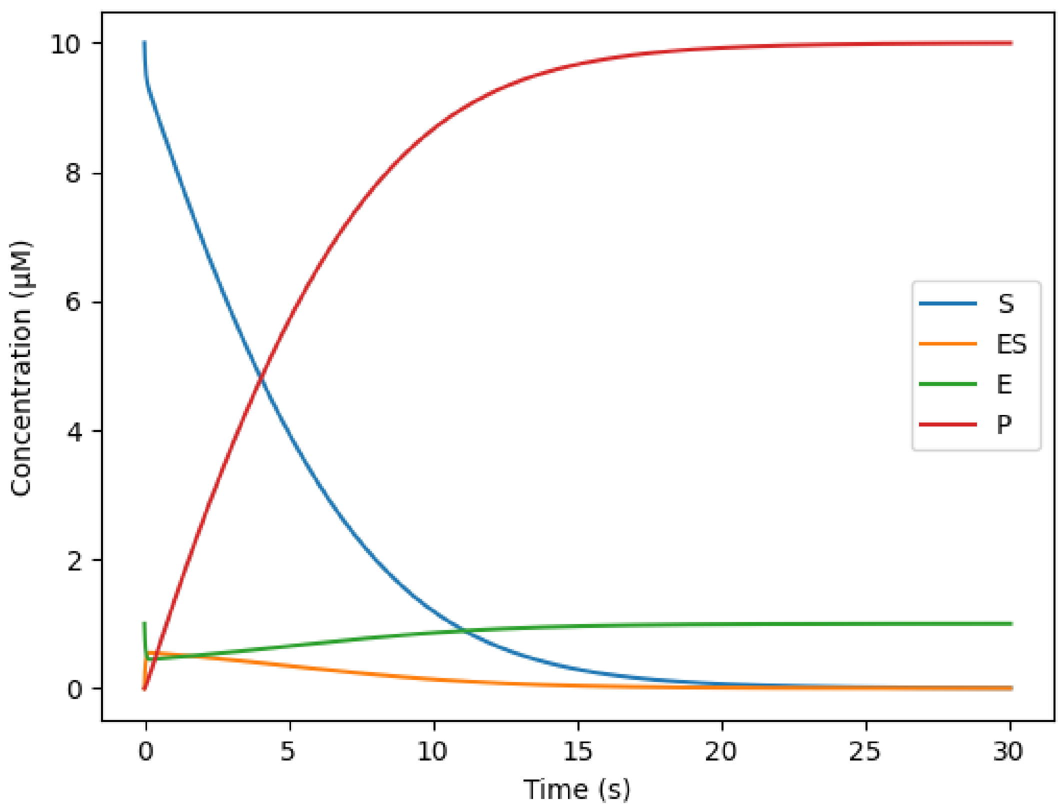
Considering that $[S] = [S_0] – [P] –[ES]$, [P] can be expressed as $[S_0] – [S] –[ES]$. Thus, value of [P] at every time point can be derived from 10 – [S] – [ES] at corresponding time points.

Values of [E], [S], [ES] and [P] every 0.02s can be stored in arrays, and these concentration values can be plotted against an array of time values.

The code and graph output will be:

```
import numpy as np
import matplotlib.pyplot as plt

def dxdt(t, x, y):
    return -(5/3)*(1 - y)*(x) + 10*y

def dydt(t, x, y):
    return (5/3)*(1 - y)*(x) - 12.5*y

x0 = 10
y0 = 0
t0 = 0
h = 0.02
tmax = 30

def runge(t, x, y, h):
    k1 = h*dxdt(t, x, y)
    l1 = h*dydt(t, x, y)
    k2 = h*dxdt(t + 0.5*h, x + 0.5*k1, y + 0.5*l1)
    l2 = h*dydt(t + 0.5*h, x + 0.5*k1, y + 0.5*l1)
    k3 = h*dxdt(t + 0.5*h, x + 0.5*k2, y + 0.5*l2)
    l3 = h*dydt(t + 0.5*h, x + 0.5*k2, y + 0.5*l2)
    k4 = h*dxdt(t + h, x + k3, y + l3)
    l4 = h*dydt(t + h, x + k3, y + l3)
    x += (1/6)*(k1 + 2*k2 + 2*k3 + k4)
    y += (1/6)*(l1 + 2*l2 + 2*l3 + l4)
    t += h
    return t, x, y

x_list = [x0]
y_list = [y0]
t_list = [t0]

while t_list[-1] < tmax:
    t, x, y = runge(t_list[-1], x_list[-1], y_list[-1], h)
    x_list.append(x)
    y_list.append(y)
    t_list.append(t)

prod_list = [0]
for i in range(1,len(y_list)):
    prod_list.append(10 - x_list[i] - y_list[i])

x_arr = np.array(x_list)
y_arr = np.array(y_list)
t_arr = np.array(t_list)
enz_arr = 1 - y_arr
prod_arr = np.array(prod_list)

plt.plot(t_arr,x_arr)
plt.plot(t_arr,y_arr)
plt.plot(t_arr,enz_arr)
plt.plot(t_arr,prod_arr)
plt.xlabel('Time (s)')
plt.ylabel('Concentration (μM)')
```

```
plt.legend(['S','ES','E','P'])
plt.show()

velo = 150*y_arr
plateau = np.max(velo)
for i in range(len(velo)-1):
    if velo[i] != plateau:
        velo[i] = plateau
    elif velo[i] == plateau:
        break

plt.plot(x_arr,velo)
plt.ylabel('Rate of change of P (min)')
plt.xlabel('Substrate concentration (μM)')
plt.show()
```



8.3

Since velocity is rate of change of P, v = d[p]/dt = 150[ES]. Previously, arrays containing values for [ES] and [S] every 0.02s were created. This graph will plot 150[ES] against [S] of corresponding time points. However, the graph shows a downwards trend immediately after reaching V_m, as this corresponds to the rapid increase in [ES] at high initial substrate concentration at early time points in part 8.2. The correct trend should show a saturation of velocity at V_m. Thus, in the array for 150[ES], all elements with positions before the maximum value (and also with smaller value) are assigned the value of V_m.

From the plot, value of V_m is **82.6**.