

1.

$$\begin{aligned}\frac{dc(S)}{dt} &= k_2 * c(ES) - k_1 * c(E) * c(S) \\ \frac{dc(E)}{dt} &= (k_2 + k_3) * c(ES) - k_1 * c(E) * c(S) \\ \frac{dc(ES)}{dt} &= k_1 * c(E) * c(S) - (k_2 + k_3) * c(ES) \\ \frac{dc(P)}{dt} &= k_3 * c(ES)\end{aligned}$$

2. fourth-order Runge-Kutta method by Python:

```
1. import numpy as np
2. import matplotlib.pyplot as plt
3.
4. # define each rate of change
5. def dx(x, y, z, w):
6.     return 600*z - 100*x*y
7.
8. def dy(x, y, z, w):
9.     return 750*z - 100*x*y
10.
11. def dz(x, y, z, w):
12.     return 100*x*y - 750*z
13.
14. def dw(x, y, z, w):
15.     return 150*z
16.
17. # fourth-order Runge-Kutta
18. def runge_kutta_4(t0, x0, y0, z0, w0, h, n):
19.     t = t0
20.     x = x0
21.     y = y0
22.     z = z0
23.     w = w0
24.     for i in range(n):
25.         k1x = h * dx(x, y, z, w)
26.         k1y = h * dy(x, y, z, w)
27.         k1z = h * dz(x, y, z, w)
28.         k1w = h * dw(x, y, z, w)
29.         k2x = h * dx(x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
30.         k2y = h * dy(x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
31.         k2z = h * dz(x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
32.         k2w = h * dw(x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
```

```

33.         k3x = h * dx( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
34.         k3y = h * dy( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
35.         k3z = h * dz( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
36.         k3w = h * dw( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
37.         k4x = h * dx( x + k3x, y + k3y, z + k3z, w + k3w)
38.         k4y = h * dy( x + k3x, y + k3y, z + k3z, w + k3w)
39.         k4z = h * dw(x + k3x, y + k3y, z + k3z, w + k3w)
40.         k4w = h * dz( x + k3x, y + k3y, z + k3z, w + k3w)
41.         x += (k1x + 2*k2x + 2*k3x + k4x)/6
42.         y += (k1y + 2*k2y + 2*k3y + k4y)/6
43.         z += (k1z + 2*k2z + 2*k3z + k4z)/6
44.         w += (k1w + 2*k2w + 2*k3w + k4w)/6
45.         t += h
46.
47.     return t, x, y, z, w

```

3. use the same method to get the plot:

```

1.     t = 0
2.     x = 10
3.     y = 1
4.     z = 0
5.     w = 0
6.     h=0.001
7.     fig, ax = plt.subplots()
8.     for i in range(90):
9.         ax.plot(x, 150*z, 'ro')
10.        k1x = h * dx(x, y, z, w)
11.        k1y = h * dy(x, y, z, w)
12.        k1z = h * dz(x, y, z, w)
13.        k1w = h * dw(x, y, z, w)
14.        k2x = h * dx(x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
15.        k2y = h * dy( x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
16.        k2z = h * dz( x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
17.        k2w = h * dw( x + k1x/2, y + k1y/2, z + k1z/2, w + k1w/2)
18.        k3x = h * dx( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
19.        k3y = h * dy( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
20.        k3z = h * dz( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
21.        k3w = h * dw( x + k2x/2, y + k2y/2, z + k2z/2, w + k2w/2)
22.        k4x = h * dx( x + k3x, y + k3y, z + k3z, w + k3w)
23.        k4y = h * dy( x + k3x, y + k3y, z + k3z, w + k3w)
24.        k4z = h * dw(x + k3x, y + k3y, z + k3z, w + k3w)

```

```

25.     k4w = h * dz( x + k3x, y + k3y, z + k3z, w + k3w)
26.     x += (k1x + 2*k2x + 2*k3x + k4x)/6
27.     y += (k1y + 2*k2y + 2*k3y + k4y)/6
28.     z += (k1z + 2*k2z + 2*k3z + k4z)/6
29.     w += (k1w + 2*k2w + 2*k3w + k4w)/6
30.     t += h
31.     plt.show()

```

As seen in figure, V_m is approximately 120

