```
import numpy as np

def f(t, y, k1, k2, k3):
    E, S, ES, P = y
    dE_dt = -k1*E*S + k2*ES+k3*ES
    dS_dt = -k1*E*S + k2*ES
    dES_dt = k1*E*S - k2*ES - k3*ES
    dP_dt = k3*ES
    return np.array([dE_dt, dS_dt, dES_dt, dP_dt])

def runge_kutta_4(t0, y0, t_max, h, k1, k2, k3):
    t = t0
    y = y0
    while t < t_max:
        k1 = h*f(t, y, k1, k2, k3)
        k2 = h*f(t + h/2, y + k1/2, k1, k2, k3)
        k3 = h*f(t + h/2, y + k2/2, k1, k2, k3)
        k4 = h*f(t + h, y + k3, k1, k2, k3)
        y = y + (k1 + 2*k2 + 2*k3 + k4)/6
        t = t + h
    return y

# Set the initial conditions
E0 = 1e-6
S0 = 10e-6
ES0 = 0
P0 = 0
y0 = np.array([E0, S0, ES0, P0])

# Set the time step and maximum time
h = 0.01   # min
t_max = 10   # min

# Set the rate constants
k1 = 100/1e-6
k2 = 600
k3 = 150

# Solve the equations using the Runge-Kutta method
y = runge_kutta_4(0, y0, t_max, h, k1, k2, k3)

# Print the final concentrations
print("E =", y[0], "µM")
print("S =", y[1], "µM")
```

```python
print("ES =", y[2], "µM")
print("P =", y[3], "µM")
```