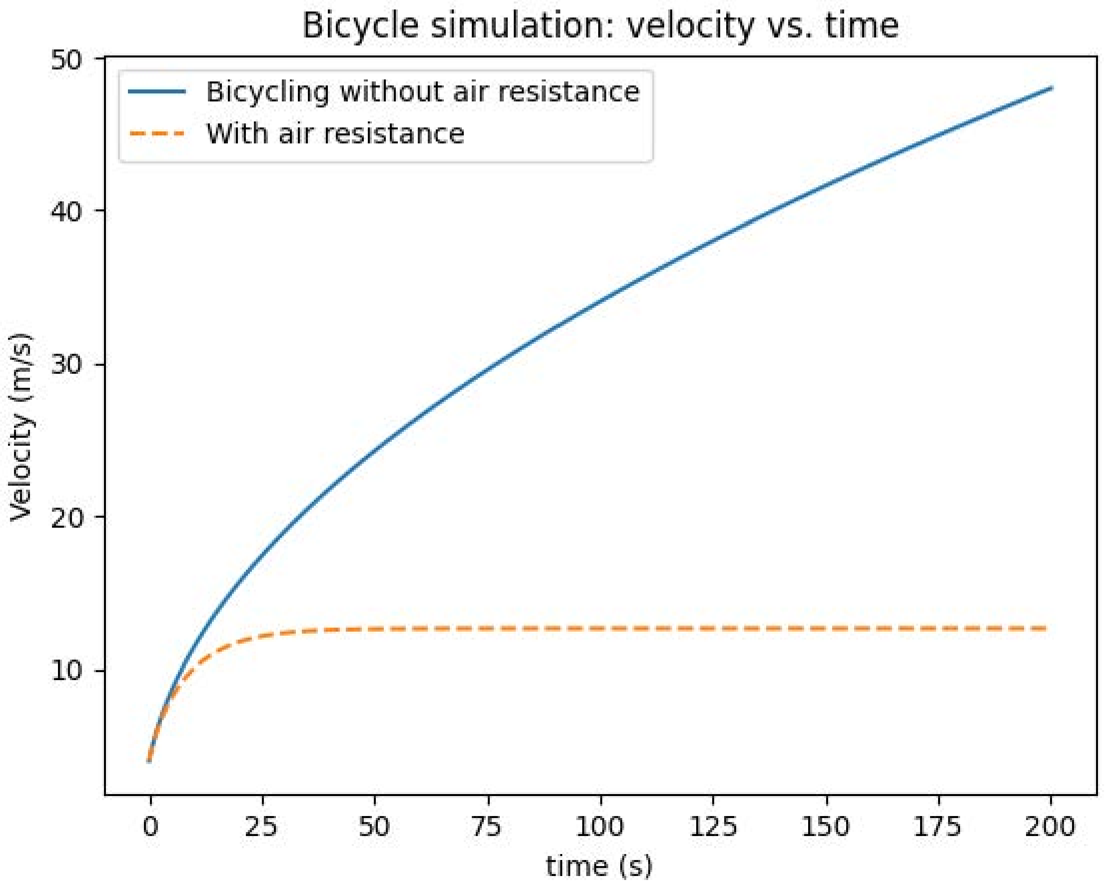
Bicycle simulation: velocity vs. time
Bicycling without air resistance
With air resistance
Velocity (m/s)
time (s)
10
20
30
40
50
0
25
50
75
100
125
150
175
200

```
import matplotlib.pyplot as plt

def main(p, m, step, dt, N, C=0, rhu=0, A=0):
    """
    p = power
    m = mass
    v = inital velocity
    dt = delta time
    N = number of iterations
    C = drag coefficient
    rhu = means Ï   , air density
    A = frontal area

    Implementation for equation 2.10 in Giordano's Computational Physics book, Page 22
    Calculates the next value of velocity and time in a series up to N iterations.
    Can be used to factor in air resistance, but by fault the drag force is zero.
    """
    for i in range(0, N):
        F_drag = ((C*rhu*A*step[i][0] ** 2)/(m))*dt
        v_i = (step[i][0] + (p/(m * step[i][0])) * dt) - F_drag
        t_i = (step[i][1] + dt)
        step[i+1] = [v_i, t_i]

    return step

def plot(x, y, title, labelx, labely, style='-', legendLabel=None):
    if(labelx is not None):
        plt.xlabel(labelx)
    if(labely is not None):
        plt.ylabel(labely)
    if(title is not None):
        plt.title(title)
    if(legendLabel is None):
        legendLabel = title
    line = plt.plot(x,y,style,label=legendLabel)
    plt.legend()
    return line

if __name__ == '__main__':
    steps = dict()
    #Power
    p = 400
    #mass
    m = 70
    #inital velocity
    v = 4
    #initial time
    t = 0
    #Increments of 0.1
    dt = 0.1
    #2000 iterations
    N = 2000
    #define inital array
```

```
steps[0] = [v, t]

main(p, m, steps, dt, N)

velocities = []
times = []
for key in steps.keys():
    velocities.append(steps[key][0])
    times.append(steps[key][1])

plot(times, velocities, 'Bicycling without air resistance', 'time (s)', 'Velocity (m/s)')

#C - drag coefficient
dragcoef=0.5
#Ï    - air density (kg/m^3)
rhu = 1.2
#A = frontal area (m^2)
frontalarea=0.33

main(p, m, steps, dt, N, C=dragcoef,rhu=rhu,A=frontalarea)

velocities = []
times = []
for key in steps.keys():
    velocities.append(steps[key][0])
    times.append(steps[key][1])

plot(times, velocities, 'Bicycle simulation: velocity vs. time', 'time (s)', 'Velocity (m/s)','--', 'With air resistance')
plt.show()
```