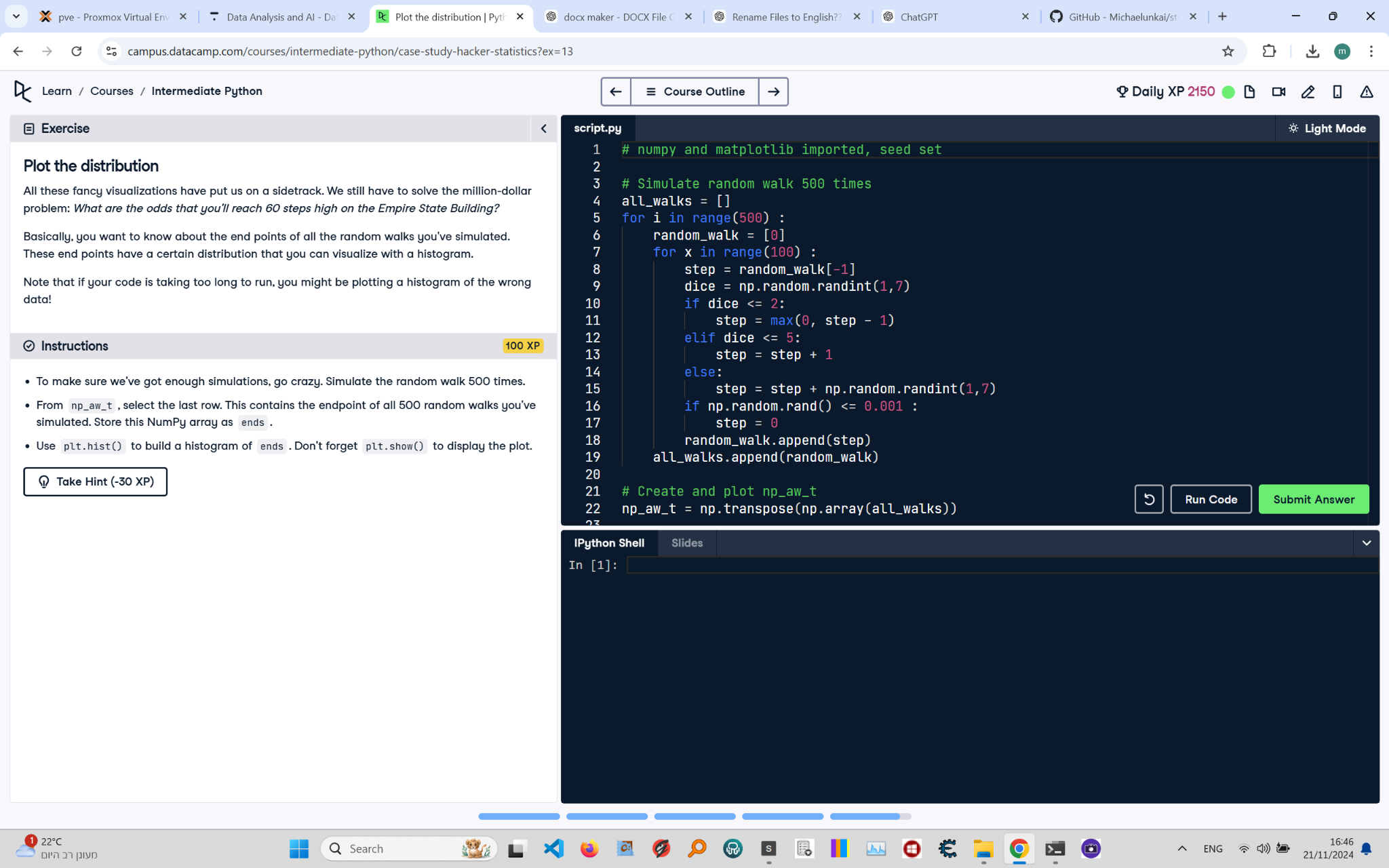
# Plot the Distribution - Corrected Solution



Below is the corrected solution for the 'Plot the Distribution' exercise from the Python course. The image includes the instructions, code, and task details.

Corrected Solution:

# numpy and matplotlib imported, seed set  
import numpy as np  
import matplotlib.pyplot as plt  
np.random.seed(123)  
  
# Simulate random walk 500 times  
all\_walks = []  
for i in range(500): # Loop runs 500 times  
 random\_walk = [0]  
 for x in range(100): # Loop runs 100 times for each walk  
 step = random\_walk[-1]  
 dice = np.random.randint(1, 7)  
 if dice <= 2:  
 step = max(0, step - 1)  
 elif dice <= 5:  
 step = step + 1  
 else:  
 step = step + np.random.randint(1, 7)  
 if np.random.rand() <= 0.001: # Introduce clumsiness  
 step = 0  
 random\_walk.append(step)  
 all\_walks.append(random\_walk)  
  
# Create and plot np\_aw\_t  
np\_aw\_t = np.transpose(np.array(all\_walks))  
  
# Select last row from np\_aw\_t: ends  
ends = np\_aw\_t[-1]  
  
# Plot histogram of ends and display the plot  
plt.hist(ends)  
plt.show()

Explanation:

1. Import numpy as np and matplotlib.pyplot as plt, and set the random seed using np.random.seed(123).

2. Simulate 500 random walks using an outer loop. Each walk consists of 100 steps simulated by an inner loop.

3. For each step, determine the next step based on the dice roll, ensuring steps don't go below 0.

4. Introduce a 'clumsiness' factor by resetting the step to 0 if np.random.rand() is less than or equal to 0.001.

5. Append the final step of each walk to all\_walks.

6. Convert all\_walks to a NumPy array, transpose it, and store the result in np\_aw\_t.

7. Extract the last row from np\_aw\_t to analyze the end points of the walks and store it in ends.

8. Plot a histogram of the values in ends using plt.hist(ends) and display it with plt.show().