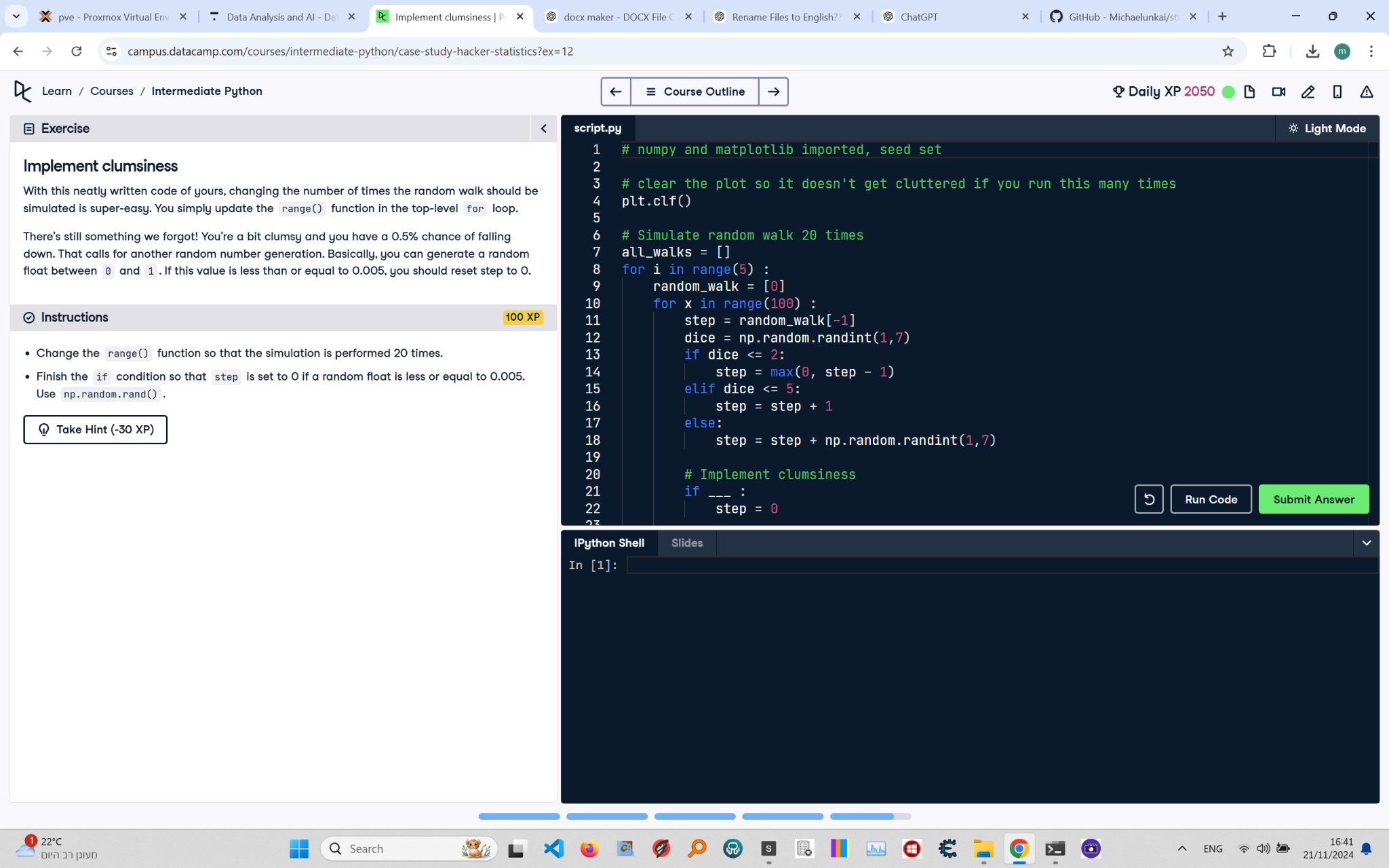
# Implement Clumsiness



Below is the exercise on 'Implement Clumsiness' 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)  
  
# Clear the plot to avoid cluttering  
plt.clf()  
  
# Simulate random walk 20 times  
all\_walks = []  
for i in range(20): # Loop runs 20 times  
 random\_walk = [0]  
 for x in range(100): # Loop runs 100 times for each walk  
 # Set step: last element in random\_walk  
 step = random\_walk[-1]  
   
 # Roll the dice  
 dice = np.random.randint(1, 7)  
   
 # Determine next step using max to prevent step from going below 0  
 if dice <= 2:  
 step = max(0, step - 1) # Move down but ensure step doesn't go below 0  
 elif dice <= 5:  
 step = step + 1 # Move up  
 else:  
 step = step + np.random.randint(1, 7) # Move up by a random value  
   
 # Implement clumsiness  
 if np.random.rand() <= 0.005: # If random float <= 0.005, reset step to 0  
 step = 0  
   
 # Append next\_step to random\_walk  
 random\_walk.append(step)  
   
 # Append random\_walk to all\_walks  
 all\_walks.append(random\_walk)  
  
# Convert all\_walks to NumPy array: np\_aw  
np\_aw = np.array(all\_walks)  
  
# Transpose np\_aw: np\_aw\_t  
np\_aw\_t = np.transpose(np\_aw)  
  
# Plot np\_aw\_t and show  
plt.plot(np\_aw\_t)  
plt.show()

Explanation:

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

2. Clear any previous plots using plt.clf() to avoid cluttering when running multiple times.

3. Modify the outer loop to simulate 20 random walks by changing the range to range(20).

4. During the random walk simulation, introduce 'clumsiness' by checking if a random float generated by np.random.rand() is less than or equal to 0.005.

5. If the clumsiness condition is met, reset the current step to 0.

6. Append the updated step to random\_walk, and after completing the walk, append random\_walk to all\_walks.

7. Convert all\_walks to a NumPy array np\_aw, transpose it to np\_aw\_t, and plot the transposed array to visualize the walks.