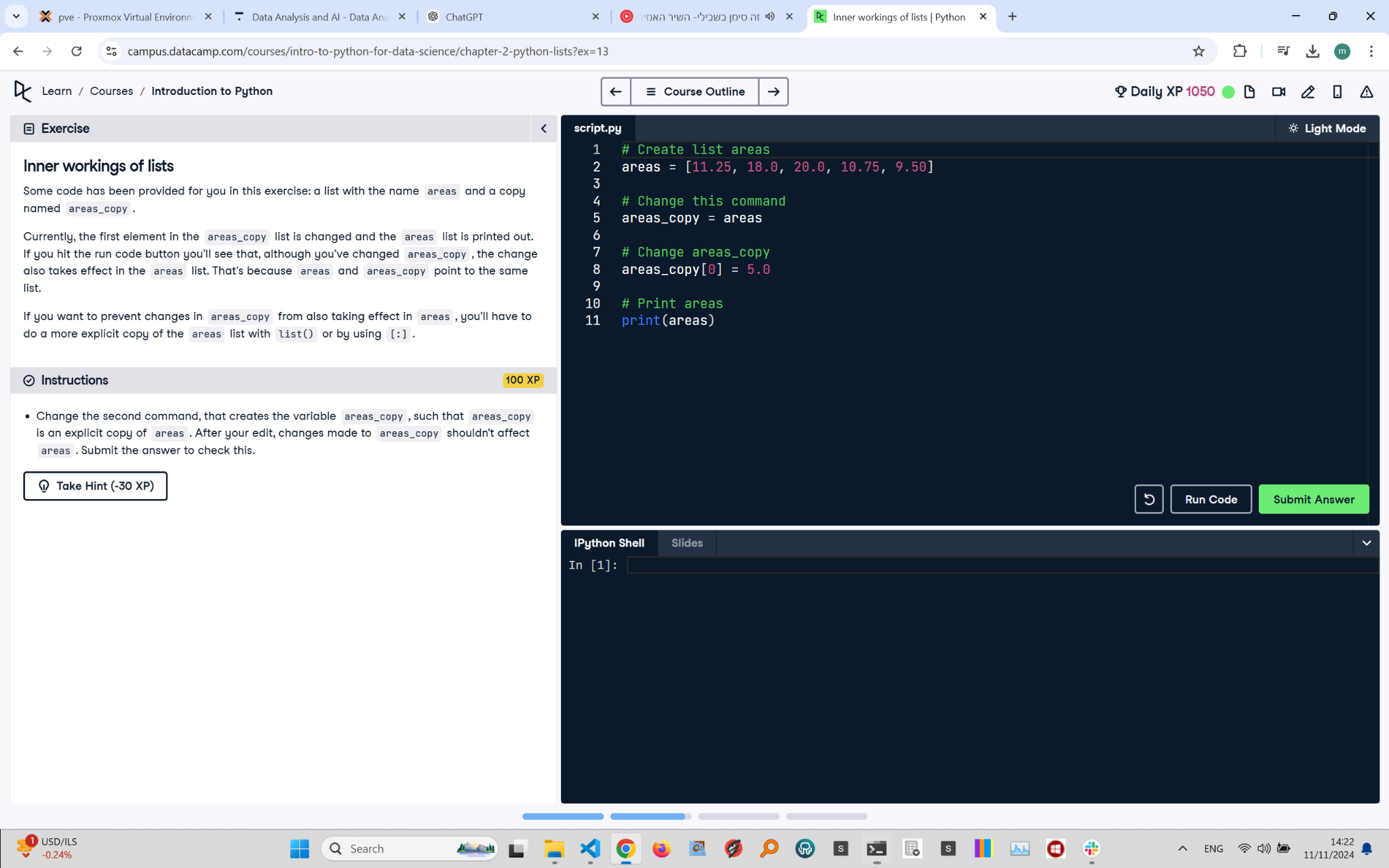
# Inner Workings of Lists - Python Exercise

Below is the image provided along with the recreated question, terminal output, and answer:



## Recreated Question and Terminal

Inner Workings of Lists  
Some code has been provided for you in this exercise: a list with the name areas and a copy named areas\_copy.

Currently, the first element in the areas\_copy list is changed and the areas list is printed out. If you hit the run code button you'll see that, although you've changed areas\_copy, the change also takes effect in the areas list. That's because areas and areas\_copy point to the same list.  
  
If you want to prevent changes in areas\_copy from also taking effect in areas, you'll have to do a more explicit copy of the areas list with list() or by using [:].

Instructions:  
- Change the second command that creates the variable areas\_copy, such that areas\_copy is an explicit copy of areas. After your edit, changes made to areas\_copy shouldn't affect areas. Submit the answer to check this.

## Corrected Answer

# Create list areas  
areas = [11.25, 18.0, 20.0, 10.75, 9.50]  
  
# Make an explicit copy of areas  
areas\_copy = areas[:]  
  
# Change areas\_copy  
areas\_copy[0] = 5.0  
  
# Print areas  
print(areas)

## Explanation of the Corrected Answer

The code uses slicing to create an explicit copy of the areas list. areas\_copy = areas[:] ensures that areas\_copy is a separate list. Changes to areas\_copy, like modifying the first element, do not affect the original areas list. The original list is then printed to confirm.