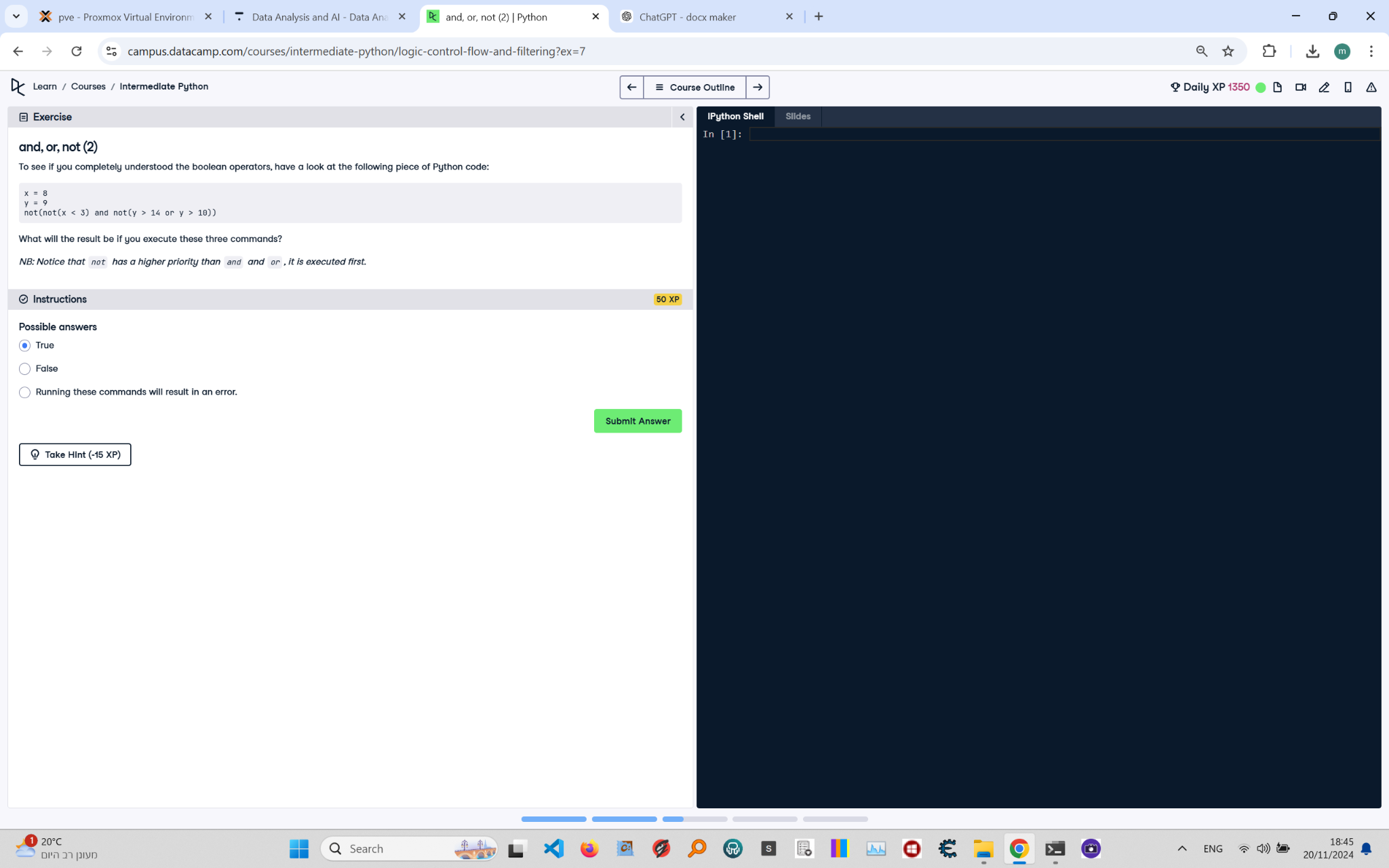
# Understanding Boolean Operators: and, or, not (2)



\*\*Question:\*\*

To see if you completely understood the boolean operators, have a look at the following piece of Python code:  
```python  
x = 8  
y = 9  
not(not(x < 3) and not(y > 14 or y > 10))  
```  
What will the result be if you execute these three commands?  
NB: Notice that `not` has a higher priority than `and` or `or`. It is executed first.

\*\*Answer:\*\*

The answer is \*\*False\*\*.

\*\*Explanation:\*\*

1. \*\*Break down the expression step by step:\*\*  
 - `x = 8` and `y = 9` are assigned as variables.  
 - Inside the innermost parentheses: `x < 3` evaluates to `False`. Applying `not`, it becomes `True`.  
 - For `y > 14 or y > 10`, both conditions are `False`. Applying `not`, it becomes `True`.  
 - Now the expression is simplified to: `not(True and True)`. Since `True and True` evaluates to `True`, we apply the `not` operator, which makes it `False`.  
 - Finally, the outer `not` negates this result, making the overall result \*\*False\*\*.