**AI-ASSISTED-CODING**

**Assignment-8.1**

**2403A51252**

**Batch-11**

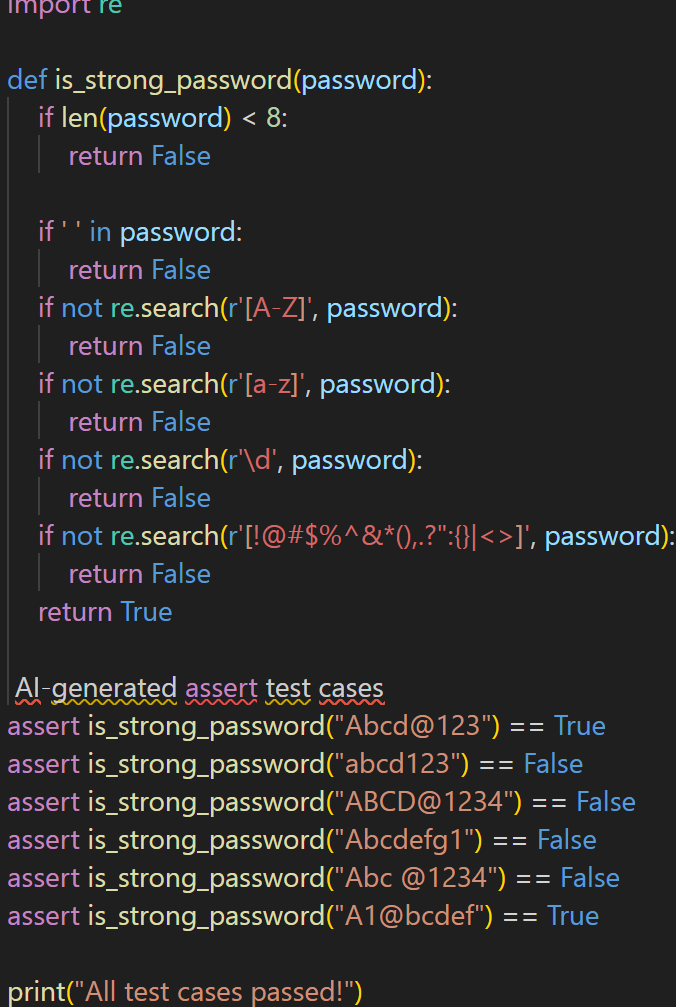
**Task-1:**

**Apply AI to generate at least 3 assert test cases for is\_strong\_password(password) and implement the validator**

**Prompt:**

Apply AI to generate at least 3 assert test cases for is\_strong\_password(password) and implement the validator function.

**Code:**



**output:**

All test cases passed!

**Task-2:**

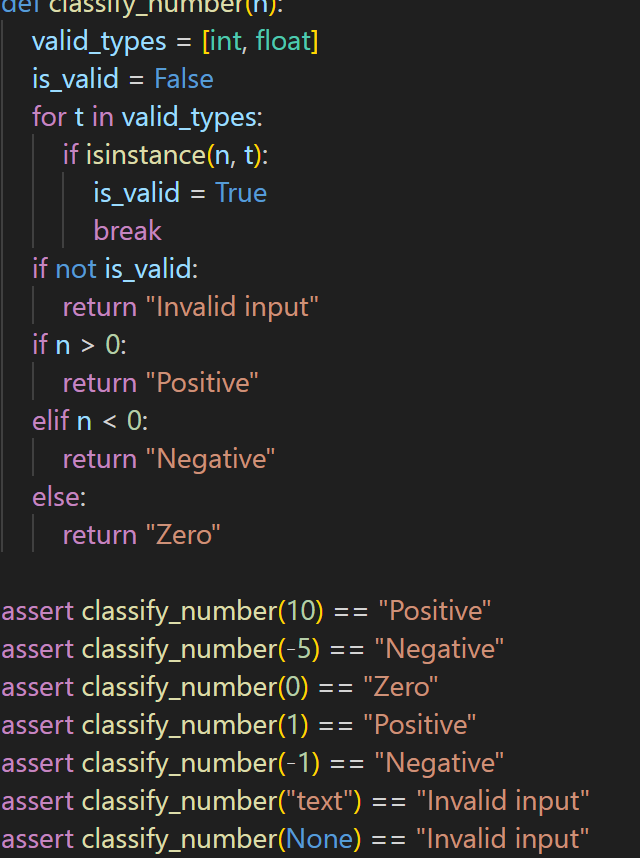
Use AI to generate at least 3 assert test cases for a  
classify\_number(n) function. Implement using loops.  
• Requirements:  
o Classify numbers as Positive, Negative, or Zero.  
o Handle invalid inputs like strings and None.

o Include boundary conditions (-1, 0, 1).  
Example Assert Test Cases:  
assert classify\_number(10) == "Positive"  
assert classify\_number(-5) == "Negative"  
assert classify\_number(0) == "Zero"

**Prompt:**

Use AI to generate at least 3 assert test cases for a classify\_number(n) function and implement it using loops

**code:**

****

**output:**

All test cases passed!

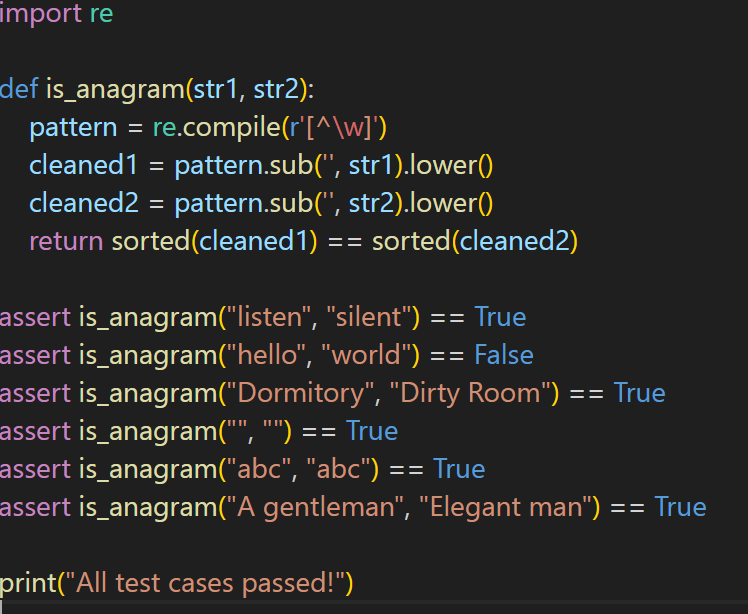
**Task-3:**

Use AI to generate at least 3 assert test cases for  
is\_anagram(str1, str2) and implement the function.  
• Requirements:  
o Ignore case, spaces, and punctuation.  
o Handle edge cases (empty strings, identical words).  
Example Assert Test Cases:  
assert is\_anagram("listen", "silent") == True  
assert is\_anagram("hello", "world") == False  
assert is\_anagram("Dormitory", "Dirty Room") == True

**Prompt:**

Use AI to generate at least 3 assert test cases for an is\_anagram(str1, str2) function and implement the function

**Code:**



**output:**

All test cases passed!

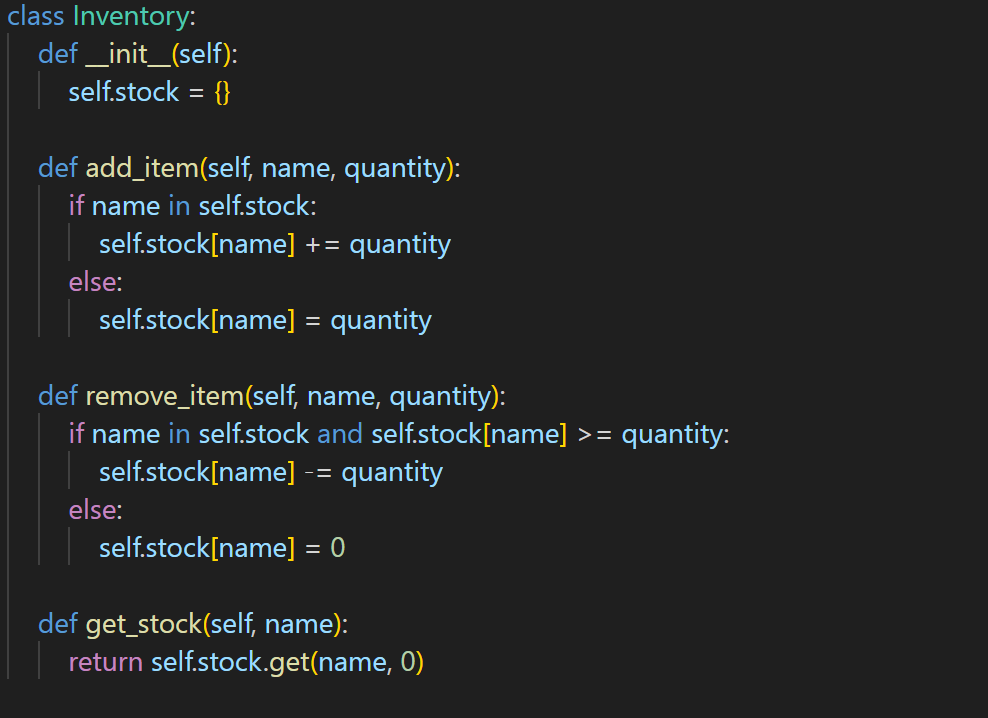
**Task-4:**

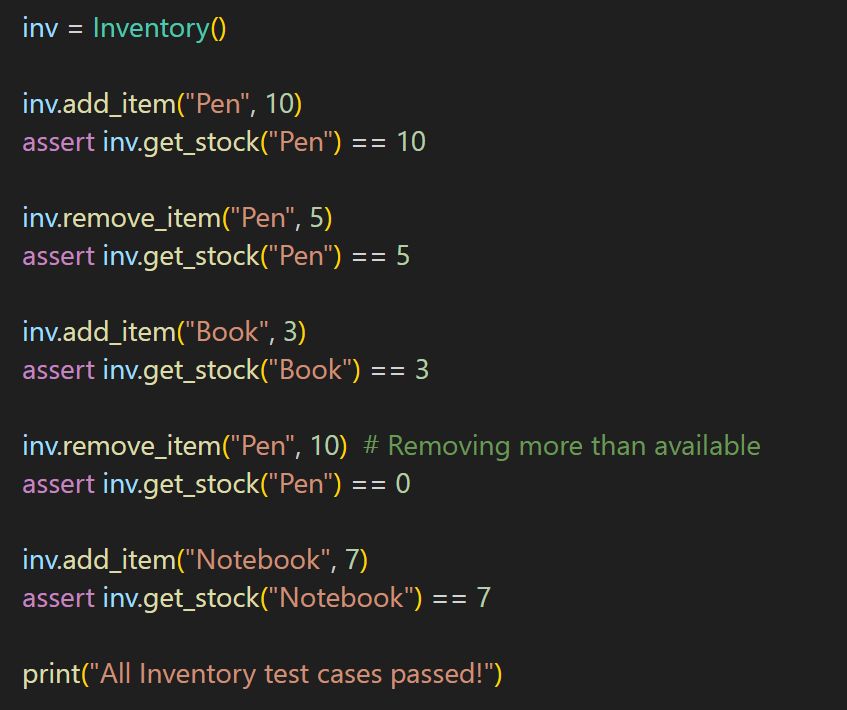
Ask AI to generate at least 3 assert-based tests for an  
Inventory class with stock management.  
• Methods:  
o add\_item(name, quantity)  
o remove\_item(name, quantity)  
o get\_stock(name)  
Example Assert Test Cases:  
inv = Inventory()  
inv.add\_item("Pen", 10)  
assert inv.get\_stock("Pen") == 10  
inv.remove\_item("Pen", 5)  
assert inv.get\_stock("Pen") == 5  
inv.add\_item("Book", 3)  
assert inv.get\_stock("Book") == 3

**Prompt:**

Ask AI to generate at least 3 assert-based tests for an **Inventory** class with stock management and implement the class

**Code:**





**output:**

All Inventory test cases passed!

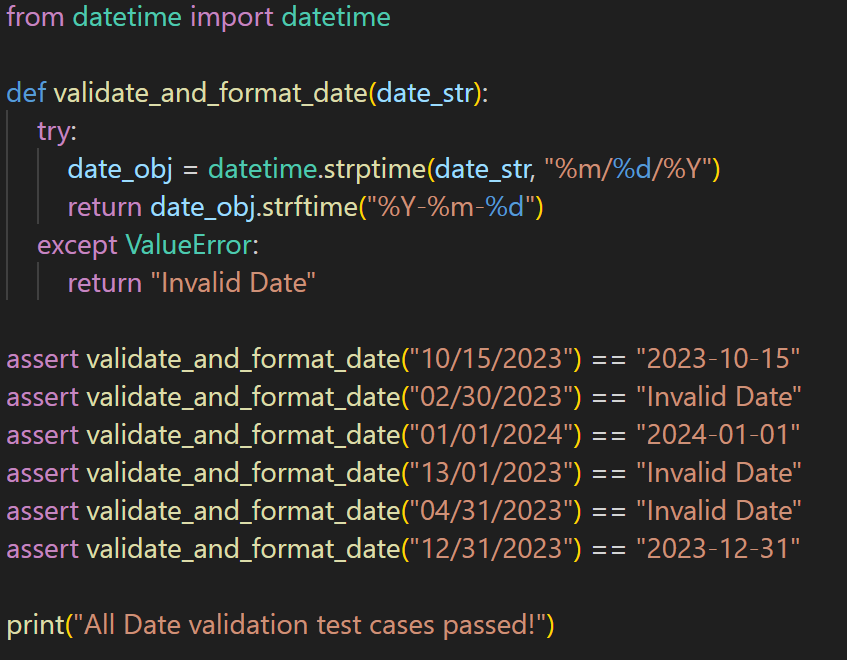
**Task-5:**

Use AI to generate at least 3 assert test cases for  
validate\_and\_format\_date(date\_str) to check and convert dates.  
• Requirements:  
o Validate "MM/DD/YYYY" format.  
o Handle invalid dates.  
o Convert valid dates to "YYYY-MM-DD".  
Example Assert Test Cases:  
assert validate\_and\_format\_date("10/15/2023") == "2023-10-15"  
assert validate\_and\_format\_date("02/30/2023") == "Invalid Date"  
assert validate\_and\_format\_date("01/01/2024") == "2024-01-01"

**Prompt:**

Use AI to generate at least 3 assert test cases for validate\_and\_format\_date(date\_str) to check and convert dates, and implement the function.

**Code:**



**output:**

All Date validation test cases passed!