

Assignment-8.5

2303A51666

Sony Dodla

Batch-23

Task Description #1 (Username Validator – Apply AI in Authentication Context)

- Task: Use AI to generate at least 3 assert test cases for a function `is_valid_username(username)` and then implement the function using Test-Driven Development principles.

- Requirements:

- o Username length must be between 5 and 15 characters.
- o Must contain only alphabets and digits.
- o Must not start with a digit.
- o No spaces allowed.

Example Assert Test Cases:

```
assert is_valid_username("User123") == True  
assert is_valid_username("12User") == False  
assert is_valid_username("Us er") == False
```

Expected Output #1:

- Username validation logic successfully passing all AI-generated test cases.

The screenshot shows a code editor with a dark theme. The code in the editor is as follows:

```
❶ 2303A51666_LabAssignment-8.py > ...  
❷ 1 def is_valid_username(username):  
❸ 2     if len(username) < 5 or len(username) > 15:  
❹ 3         return False  
❺ 4     if not username[0].isalpha():  
❻ 5         return False  
❼ 6     for char in username:  
➋ 7         if not (char.isalnum() or char == '_'):  
⌋ 8             return False  
⌊ 9         return True  
⌋ 10 #test cases for the is_valid_username function  
⌋ 11 assert is_valid_username("User123") == True  
⌋ 12 assert is_valid_username("12User") == False  
⌋ 13 assert is_valid_username("Us er") == False  
⌋ 14 print("All test cases for is_valid_username passed!")
```

Below the code editor, there is a navigation bar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is selected. The terminal output shows:

```
| PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe"  
| op/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py"  
All test cases for is valid username passed!
```

Task Description #2 (Even–Odd & Type Classification – Apply AI for Robust Input Handling)

- Task: Use AI to generate at least 3 assert test cases for a function `classify_value(x)` and implement it using conditional logic and loops.

- Requirements:

- If input is an integer, classify as "Even" or "Odd".
- If input is 0, return "Zero".
- If input is non-numeric, return "Invalid Input".

Example Assert Test Cases:

```
assert classify_value(8) == "Even"  
assert classify_value(7) == "Odd"  
assert classify_value("abc") == "Invalid Input"
```

Expected Output #2:

- Function correctly classifying values and passing all test cases

```
17 def classify_value(x):  
18     if x < 0:  
19         return "Negative"  
20     elif x == 0:  
21         return "Zero"  
22     elif x%2 == 0:  
23         return "Even"  
24     else:  
25         return "Odd"  
26     # Test cases for the classify_value function  
27 assert classify_value(8) == "Even"  
28 assert classify_value(7) == "Odd"  
29 assert classify_value("abc") == "Invalid Input"  
  
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS  
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING & C:/Users/SONY REDDY/AppData/Local/Programs/Py  
File "c:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING\2303A51666_LabAssignment-8.py", line 29, in  
    assert classify_value("abc") == "Invalid Input"  
~~~~~  
File "c:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING\2303A51666_LabAssignment-8.py", line 18, in  
    if x < 0:  
~~~~~  
TypeError: '<' not supported between instances of 'str' and 'int'
```

Task Description #3 (Palindrome Checker – Apply AI for String Normalization)

- Task: Use AI to generate at least 3 assert test cases for a function `is_palindrome(text)` and implement the function.

- Requirements:

- Ignore case, spaces, and punctuation.
- Handle edge cases such as empty strings and single characters.

Example Assert Test Cases:

```
assert is_palindrome("Madam") == True  
assert is_palindrome("A man a plan a canal Panama") ==
```

True

```
assert is_palindrome("Python") == False
```

Expected Output #3:

- Function correctly identifying palindromes and passing all AI-generated tests

The screenshot shows a Jupyter Notebook cell with the following code:

```
31 def is_palindrome(text):
32     cleaned_text = ''.join(char.lower() for char in text if char.isalnum())
33     return cleaned_text == cleaned_text[::-1]
34 # Test cases for the is_palindrome function
35 assert is_palindrome("Madam") == True
36 assert is_palindrome("A man a plan a canal Panama") == True
37 assert is_palindrome("Python") == False
38 print("All test cases for is_palindrome passed!")
```

Below the code, there is a navigation bar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected), and PORTS.

The terminal output shows:

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py"
All test cases for is_palindrome passed!
```

Task Description #4 (BankAccount Class – Apply AI for Object-Oriented Test-Driven Development)

- Task: Ask AI to generate at least 3 assert-based test cases for a BankAccount class and then implement the class.

- Methods:

- o deposit(amount)

- o withdraw(amount)

- o get_balance()

Example Assert Test Cases:

```
acc = BankAccount(1000)
```

```
acc.deposit(500)
```

```
assert acc.get_balance() == 1500
```

```
acc.withdraw(300)
```

```
assert acc.get_balance() == 1200
```

Expected Output #4:

- Fully functional class that passes all AI-generated assertions.

```

2303A51666_LabAssignment-8.py > ...
41 class BankAccount:
42     def __init__(self, account_number, balance=0):
43         self.account_number = account_number
44         self.balance = balance
45
46     def deposit(self, amount):
47         if amount > 0:
48             self.balance += amount
49             return True
50         return False
51
52     def withdraw(self, amount):
53         if 0 < amount <= self.balance:
54             self.balance -= amount
55             return True
56         return False
57     def get_balance(self):
58         return self.balance
59
# Test cases for the BankAccount class
60 acc = BankAccount(1000)
61 acc.deposit(500)
62 assert acc.get_balance() == 1500
63 acc.withdraw(300)
64 assert acc.get_balance() == 1200
65 print("All test cases for BankAccount passed!")

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py"
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py"
Traceback (most recent call last):
  File "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py", line 62
    assert acc.get_balance() == 1500
                           ^
AssertionError

```

Task Description #5 (Email ID Validation – Apply AI for Data Validation)

- Task: Use AI to generate at least 3 assert test cases for a function validate_email(email) and implement the function.

- Requirements:

- Must contain @ and .
- Must not start or end with special characters.
- Should handle invalid formats gracefully.

Example Assert Test Cases:

```

assert validate_email("user@example.com") == True
assert validate_email("userexample.com") == False
assert validate_email("@gmail.com") == False

```

Expected Output #5:

- Email validation function passing all AI-generated test cases and handling edge cases correctly.

```
68 def validate_email(email):
69     if '@' not in email or '.' not in email:
70         return False
71     at_index = email.index('@')
72     dot_index = email.rindex('.')
73     if at_index < 1 or dot_index < at_index + 2 or dot_index >= len(email) - 1:
74         return False
75     return True
76 # Test cases for the validate_email function
77 assert validate_email("user@example.com") == True
78 assert validate_email("user@example.com") == False
79 assert validate_email("@gmail.com") == False
80 print("All test cases for validate_email passed!")

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
```

PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python op/AI ASSISTANT CODING/2303A51666_LabAssignment-8.py"
All test cases for validate_email passed!