

Assignment-8.5

Ht. No: 2303A51923

Name: V.Sravani

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.

```
task-01.py > ...
1  def is_valid_username(username):
2      if len(username) < 5 or len(username) > 15:
3          return False
4      if not username[0].isalpha():
5          return False
6      for char in username:
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!")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\SRAVANI\Documents\AI Assist Coding> & C:/Users/SRAVANI/
Users/SRAVANI/Documents/AI Assist Coding/task-01.py"
All test cases for is_valid_username passed!
PS C:\Users\SRAVANI\Documents\AI Assist Coding>
```

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:

- o If input is an integer, classify as "Even" or "Odd".
- o If input is 0, return "Zero".
- o 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.

```
1  def classify_value(x):
2      if x < 0:
3          return "Negative"
4      elif x == 0:
5          return "Zero"
6      elif x%2 == 0:
7          return "Even"
8      else:
9          return "Odd"
10 # Test cases for the classify_value function
11 assert classify_value(8) == "Even"
12 assert classify_value(7) == "Odd"
13 assert classify_value("abc") == "Invalid Input"
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\SRAVANI\Documents\AI Assist Coding> & C:/Users/SRAVANI/AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation\Python\Python39\python.exe C:/Users/SRAVANI/Documents/AI Assist Coding/task-01.py
Traceback (most recent call last):
  File "c:\Users\SRAVANI\Documents\AI Assist Coding\task-01.py", line 13, in <module>
    assert classify_value("abc") == "Invalid Input"
           ~~~~~^~~~~~
  File "c:\Users\SRAVANI\Documents\AI Assist Coding\task-01.py", line 2, in classify_value
    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:

- o Ignore case, spaces, and punctuation.
- o 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 code editor with a Python function `is_palindrome` and its test cases. The function cleans the input text by removing non-alphanumeric characters and converting it to lowercase, then checks if it is a palindrome. The test cases use `assert` to verify the function's behavior for "Madam", "A man a plan a canal Panama", and "Python". Below the code, the terminal output shows the command to run the script and the message "All test cases for is_palindrome passed!".

```
1 def is_palindrome(text):  
2     cleaned_text = ''.join(char.lower() for char in text if char.isalnum())  
3     return cleaned_text == cleaned_text[::-1]  
4 # Test cases for the is_palindrome function  
5 assert is_palindrome("Madam") == True  
6 assert is_palindrome("A man a plan a canal Panama") == True  
7 assert is_palindrome("Python") == False  
8 print("All test cases for is_palindrome passed!")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\SRAVANI\Documents\AI Assist Coding> & C:/Users/SRAVANI/AppData/Local/Python  
Users/SRAVANI/Documents/AI Assist Coding/task-01.py  
All test cases for is_palindrome passed!  
PS C:\Users\SRAVANI\Documents\AI Assist Coding>
```

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

```
1 class BankAccount:
2     def __init__(self, account_number, balance=0):
3         self.account_number = account_number
4         self.balance = balance
5
6     def deposit(self, amount):
7         if amount > 0:
8             self.balance += amount
9             return True
10        return False
11
12    def withdraw(self, amount):
13        if 0 < amount <= self.balance:
14            self.balance -= amount
15            return True
16        return False
17    def get_balance(self):
18        return self.balance
19
20 # Test cases for the BankAccount class
21 acc = BankAccount(1000)
22 acc.deposit(500)
23 assert acc.get_balance() == 1500
24 acc.withdraw(300)
25 assert acc.get_balance() == 1200
26 print("All test cases for BankAccount passed!")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Traceback (most recent call last):
File "c:\Users\SRAVANI\Documents\AI Assist Coding\task-01.py", line 22, in <module>
assert acc.get_balance() == 1500
~~~~~  
**AssertionError**  
PS C:\Users\SRAVANI\Documents\AI Assist Coding>

## 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:

- o Must contain `@` and `.`

- o Must not start or end with special characters.

- o 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.

```
1 def validate_email(email):
2     if '@' not in email or '.' not in email:
3         return False
4     at_index = email.index('@')
5     dot_index = email.rindex('.')
6     if at_index < 1 or dot_index < at_index + 2 or dot_index >= len(email) - 1:
7         return False
8     return True
9 # Test cases for the validate_email function
10 assert validate_email("user@example.com") == True
11 assert validate_email("userexample.com") == False
12 assert validate_email("@gmail.com") == False
13 print("All test cases for validate_email passed!")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
PS C:\Users\SRAVANI\Documents\AI Assist Coding> & C:/Users/SRAVANI/AppData/Local/Python/pythoncon
Users/SRAVANI/Documents/AI Assist Coding/task-01.py"
All test cases for validate_email passed!
PS C:\Users\SRAVANI\Documents\AI Assist Coding>
```