AI Assisted Coding
Assignment – 8.5
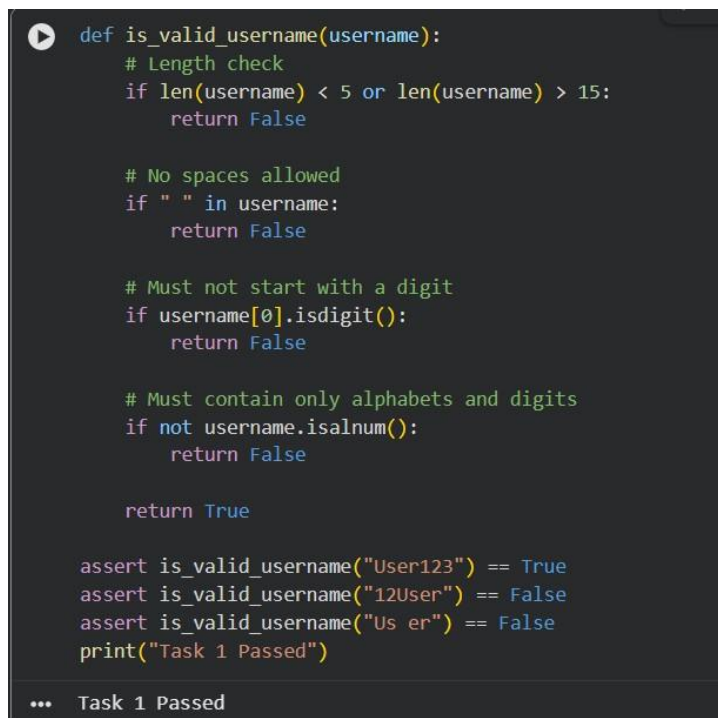
Name: S.Ravikiran

HT No: 2303A52261

Batch: 36

Lab Objectives

- To introduce students to Test-Driven Development (TDD) using AI tools.

- To generate test cases before writing Python implementations.

- To reinforce the importance of validation, testing, and error handling.

- To encourage clean and reliable code based on AI-generated test expectations.

Task 1: Username Validator (Authentication Context)

AI-Generated Assert Test Cases

```python
def is_valid_username(username):
    # Length check
    if len(username) < 5 or len(username) > 15:
        return False

    # No spaces allowed
    if " " in username:
        return False

    # Must not start with a digit
    if username[0].isdigit():
        return False

    # Must contain only alphabets and digits
    if not username.isalnum():
        return False

    return True

assert is_valid_username("User123") == True
assert is_valid_username("12User") == False
assert is_valid_username("Us er") == False
print("Task 1 Passed")
```

```
... Task 1 Passed
```

Task 2: Even–Odd & Type Classification

AI-Generated Assert Test Cases

```python
def classify_value(x):
    if not isinstance(x, int):
        return "Invalid Input"

    if x == 0:
        return "Zero"

    if x % 2 == 0:
        return "Even"
    else:
        return "Odd"


assert classify_value(8) == "Even"
assert classify_value(7) == "Odd"
assert classify_value(0) == "Zero"
assert classify_value("abc") == "Invalid Input"
assert classify_value(2.5) == "Invalid Input"

assert classify_value(8) == "Even"
assert classify_value(7) == "Odd"
assert classify_value("abc") == "Invalid Input"
print("Task 2 Passed")
```

```
Task 2 Passed
```

Task 3: Palindrome Checker (String Normalization)

AI-Generated Assert Test Cases

```python
import string

def is_palindrome(text):
    # Normalize text: remove spaces and punctuation, lower case
    cleaned = ""
    for ch in text.lower():
        if ch.isalnum():
            cleaned += ch

    return cleaned == cleaned[::-1]

assert is_palindrome("Madam") == True
assert is_palindrome("Python") == False
assert is_palindrome("") == True
print("Task 3 Passed")
```

```
...  Task 3 Passed
```

Task 4: BankAccount Class (Object-Oriented TDD)

```python
class BankAccount:
    def __init__(self, balance=0):
        self.balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount

    def withdraw(self, amount):
        if amount > 0 and amount <= self.balance:
            self.balance -= amount

    def get_balance(self):
        return self.balance

acc = BankAccount(1000)
acc.deposit(500)
assert acc.get_balance() == 1500

acc.withdraw(300)
assert acc.get_balance() == 1200

acc.withdraw(5000)
assert acc.get_balance() == 1200
print("Task 4 Passed")
```

```
...  Task 4 Passed
```

Task 5: Email ID Validation (Data Validation)

```python
def validate_email(email):
    if "@" not in email or "." not in email:
        return False

    # Must not start or end with special characters
    if email[0] in "@." or email[-1] in "@.":
        return False

    # Must contain exactly one @
    if email.count("@") != 1:
        return False

    return True

assert validate_email("user@example.com") == True
assert validate_email("userexample.com") == False
assert validate_email("@example.com") == False
print("Task 5 Passed")
```

... Task 5 Passed

Final Conclusion

This lab successfully demonstrated Test-Driven Development using AI-generated assert test cases. Writing tests before implementation improved correctness, reliability, and error handling. AI-assisted test generation was useful, but human verification was necessary to ensure complete coverage and logical accuracy.