**AI ASSISTED CODING **

*Lab 8.2: Test-Driven Development with AI – Generating and Working with Test Cases*

**Roll no**: 2503A51L06

**Name**:D.Nagamrutha

**Batch**: 25BTCAICSB19

**TASK#1:**

**Prompt:** Generate test cases for a function is prime(n) and then implement the  
function.

**Code Generated:**

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.**Output:**

**Observation:**

The code defines an efficient is\_prime function with edge case handling and tests it against various cases. Results are printed with expected outputs, and validation confirms correctness if at least one prime is identified**.**

**TASK#2:**

**Prompt:** Generate test cases for celsius\_to\_fahrenheit(c) and fahrenheit\_to\_celsius(f).

**Code Generated:**

A screen shot of a computer program

AI-generated content may be incorrect.

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Observation:**

This code implements Celsius-to-Fahrenheit and Fahrenheit-to-Celsius conversions with type checks to ensure only numeric inputs are processed. It uses well-defined test cases, including standard values, negatives, and decimals, then compares expected and actual results with outputs rounded for accuracy, confirming the correctness of the conversion logic.

**TASK#3:**

**Prompt:** Write test cases for a function count\_words(text) that returns the number of  
words in a sentence.

**Code Generated:**

A computer screen shot of a program code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.**Output:**

**Observation:**

This code defines a count\_words function that uses regex to accurately count words in a string while handling punctuation, extra spaces, empty strings, and numbers. The test cases cover normal sentences, multiple spaces, punctuation, single words, and empty inputs, with outputs showing both expected and actual results to verify the function’s correctness.

**TASK#4:**

**Prompt**: Generate test cases for a BankAccount class with:

Methods:  
 deposit(amount)  
 withdraw(amount)  
 check\_balance()

A screen shot of a computer program

AI-generated content may be incorrect.**Code Generated:**

A screenshot of a computer error

AI-generated content may be incorrect.**Output:**

**Observation:**

This code defines a BankAccount class with methods to deposit, withdraw, and check balance, including error handling for negative amounts and overdrafts. The test cases verify valid deposits and withdrawals, attempt invalid operations, and print expected versus actual results, demonstrating that the class correctly manages the account balance and enforces constraints.

**TASK#5:**

**Prompt**: Generate test cases for is\_number\_palindrome(num), which checks if an integer reads the same backward.

Examples:  
121→True  
123→False  
0, negative numbers → handled gracefully.

**Code Generated:**

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.**Output:**

**Observation:**

This code defines an is\_number\_palindrome function that checks if an integer reads the same backward, handling negative numbers and non-integer inputs. The test cases include positive palindromes, non-palindromes, zero, and negative numbers, with outputs printed alongside expected results to verify correctness.