SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 2

Name of Student : R.SHIVA SAI REDDY
Enrollment No. : 2303A51542
Batch No. : 22

Task 1: Cleaning Sensor Data

❖ Scenario:You are cleaning IoT sensor data where negative values are invalid.

❖ Task:Use Gemini in Colab to generate a function that filters out all negative numbers from a list.

❖ Expected Output:

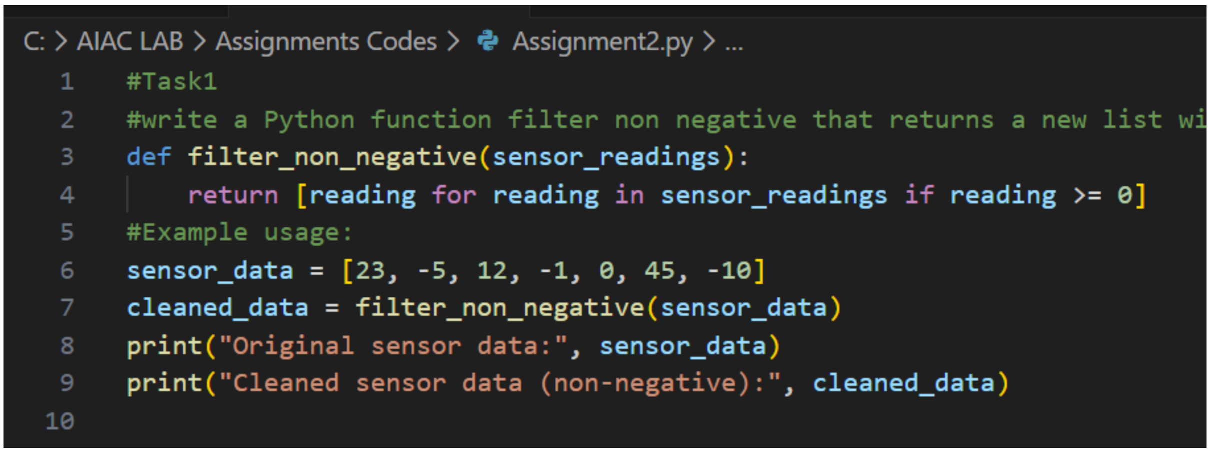
➢ Before/after list

➢ Screenshot of Colab execution

Prompt:-

write a Python function filter non negative that returns a new list with only values greater than or equal to 0, without modifying the original list, for cleaning invalid negative IoT sensor readings.

Code:-

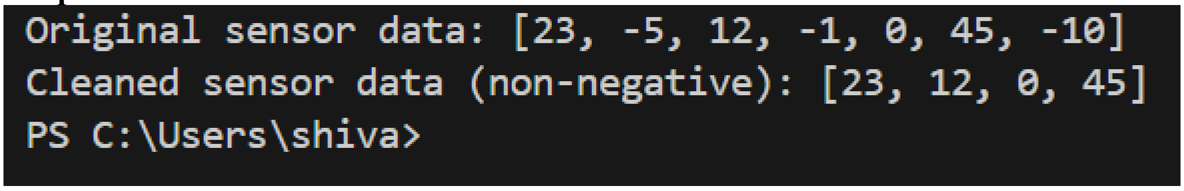


C: > AIAC LAB > Assignments Codes > Assignment2.py > ...

```python
#Task1
#write a Python function filter non negative that returns a new list wi
def filter_non_negative(sensor_readings):
    return [reading for reading in sensor_readings if reading >= 0]
#Example usage:
sensor_data = [23, -5, 12, -1, 0, 45, -10]
cleaned_data = filter_non_negative(sensor_data)
print("Original sensor data:", sensor_data)
print("Cleaned sensor data (non-negative):", cleaned_data)

```

Output:-



```
Original sensor data: [23, -5, 12, -1, 0, 45, -10]
Cleaned sensor data (non-negative): [23, 12, 0, 45]
PS C:\Users\shiva>
```

Justification:-

This function uses a list comprehension to create a new list that includes only the non-negative values from the original list. It does not modify the original list,ensuring that the raw sensor data remains intact for any further analysis or processing.

Task 2: String Character Analysis

❖ Scenario:

You are building a text-analysis feature.

❖ Task:

Use Gemini to generate a Python function that counts vowels, consonants, and digits in a string.

❖ Expected Output:

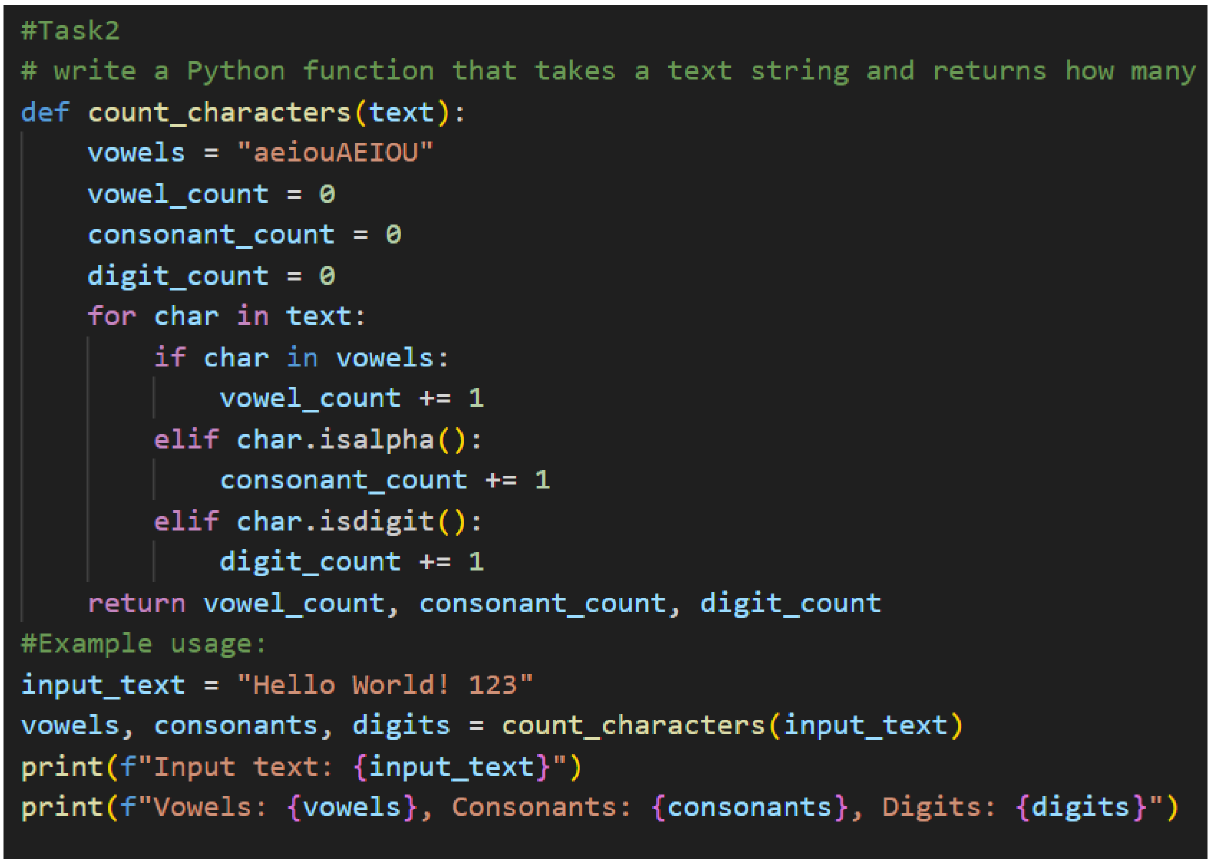
➢ Working function

➢ Sample inputs and outputs

Prompt:-

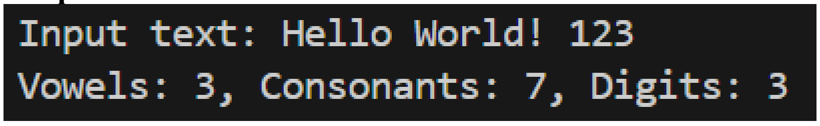
write a friendly Python function that takes a text string and returns how many vowels, consonants, and digits it contains, treating upper and lower case letters the same.

Code:-



```python
#Task2
# write a Python function that takes a text string and returns how many
def count_characters(text):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    digit_count = 0
    for char in text:
        if char in vowels:
            vowel_count += 1
        elif char.isalpha():
            consonant_count += 1
        elif char.isdigit():
            digit_count += 1
    return vowel_count, consonant_count, digit_count
#Example usage:
input_text = "Hello World! 123"
vowels, consonants, digits = count_characters(input_text)
print(f"Input text: {input_text}")
print(f"Vowels: {vowels}, Consonants: {consonants}, Digits: {digits}")
```

Output:-



```
Input text: Hello World! 123
Vowels: 3, Consonants: 7, Digits: 3
```

Justification:-This function iterates through each character in the input string, checking if it is a vowel, consonant, or digit, and increments the respective

counters accordingly. It treats upper and lower case letters the same by including both in the vowels string and using isalpha() for consonants.

Task 3: Palindrome Check – Tool Comparison

❖ Scenario:

You must decide which AI tool is clearer for string logic.

❖ Task:

Generate a palindrome-checking function using Gemini and Copilot, then compare the results.

❖ Expected Output:

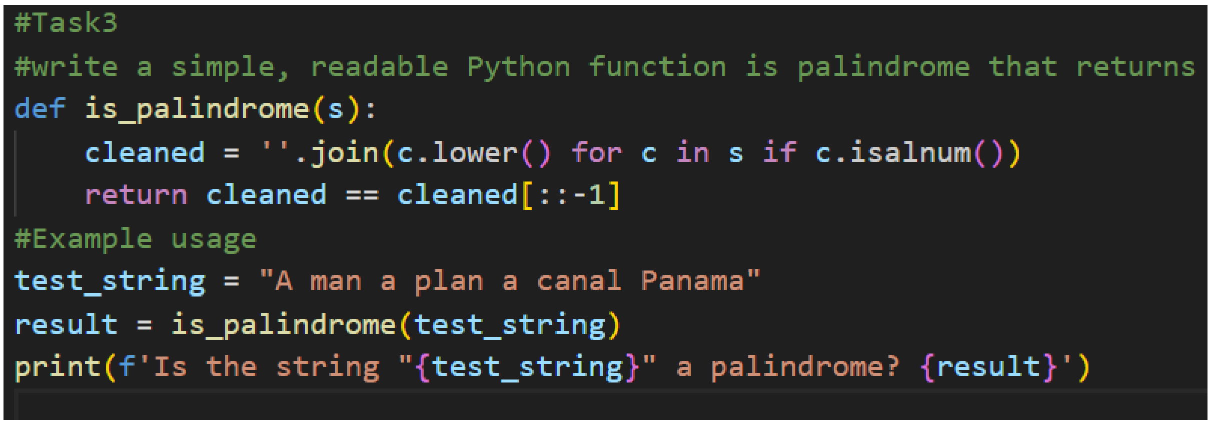
➢ Side-by-side code comparison

➢ Observations on clarity and structure

Prompt:-

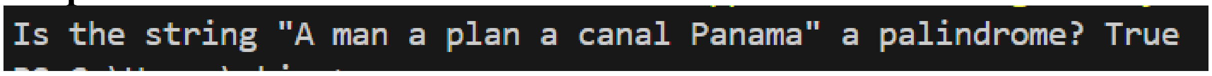
write a simple, readable Python function is palindrome that returns True if a string reads the same forwards and backwards ignoring case and spaces and False otherwise.

Code:-



```python
#Task3
#write a simple, readable Python function is palindrome that returns
def is_palindrome(s):
    cleaned = ''.join(c.lower() for c in s if c.isalnum())
    return cleaned == cleaned[::-1]
#Example usage
test_string = "A man a plan a canal Panama"
result = is_palindrome(test_string)
print(f'Is the string "{test_string}" a palindrome? {result}')
```

Output:-



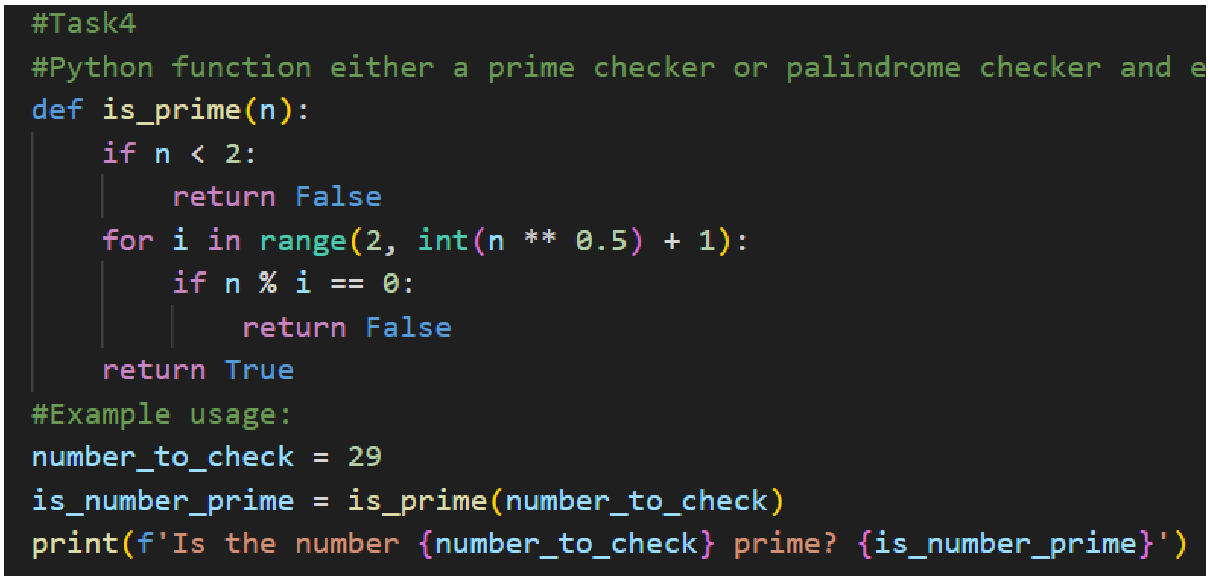
```
Is the string "A man a plan a canal Panama" a palindrome? True
```

Justification:-

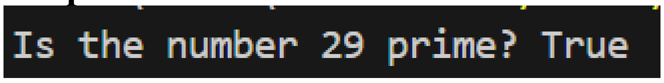
This function is simple and readable because it breaks down the problem into clear steps: cleaning the string by removing non-alphanumeric characters and converting to lowercase, then checking if the cleaned string is equal to its reverse. This makes it easy for beginners to understand how palindromes work.

Task 4: Code Explanation Using AI
❖ Scenario:
You are reviewing unfamiliar code written by another developer.
❖ Task:
Ask Gemini to explain a Python function (prime check OR palindrome check) line by line.
❖ Expected Output:
➢ Code snippet
➢ AI explanation
➢ Student comments on understanding

Prompt:-
Python function either a prime checker or palindrome checker and explain it line by line in simple, student friendly language, as if teaching a beginner.

Code:-


```python
#Task4
#Python function either a prime checker or palindrome checker and e
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True
#Example usage:
number_to_check = 29
is_number_prime = is_prime(number_to_check)
print(f'Is the number {number_to_check} prime? {is_number_prime}')
```

Output:-


```
Is the number 29 prime? True
```

Justification:-
The function checks if a number is prime by first handling numbers less than 2 (which are not prime). Then it iterates from 2 up to the square root of the

number, checking for divisibility. If any divisor is found, it returns False; otherwise, it returns True.