SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 2

Program	**: B. Tech (CSE)**
Specialization	**: CSE**
Course Title	**: AI Assisted Coding**
Course Code	**: 23CS201PC302**
Semester	**: 3 -2**
Academic Session	**: 2025-2026**
Name of Student	**: Eshwar**
Enrollment No.	**: 2403a51l26**
Batch No.	**: 51**
Date	**:20-01-2026**

Submission Instructions:

(All instructions should be followed strictly to avoid deduction of marks)

1. Use the same file to complete the assignment and don’t change the settings.
2. Minimum 10 screen shots of your account should be taken to showcase your work.
3. **File Format:**
   - Submit your assignment as a PDF document (pdf). Ensure the file is named according to the following convention:

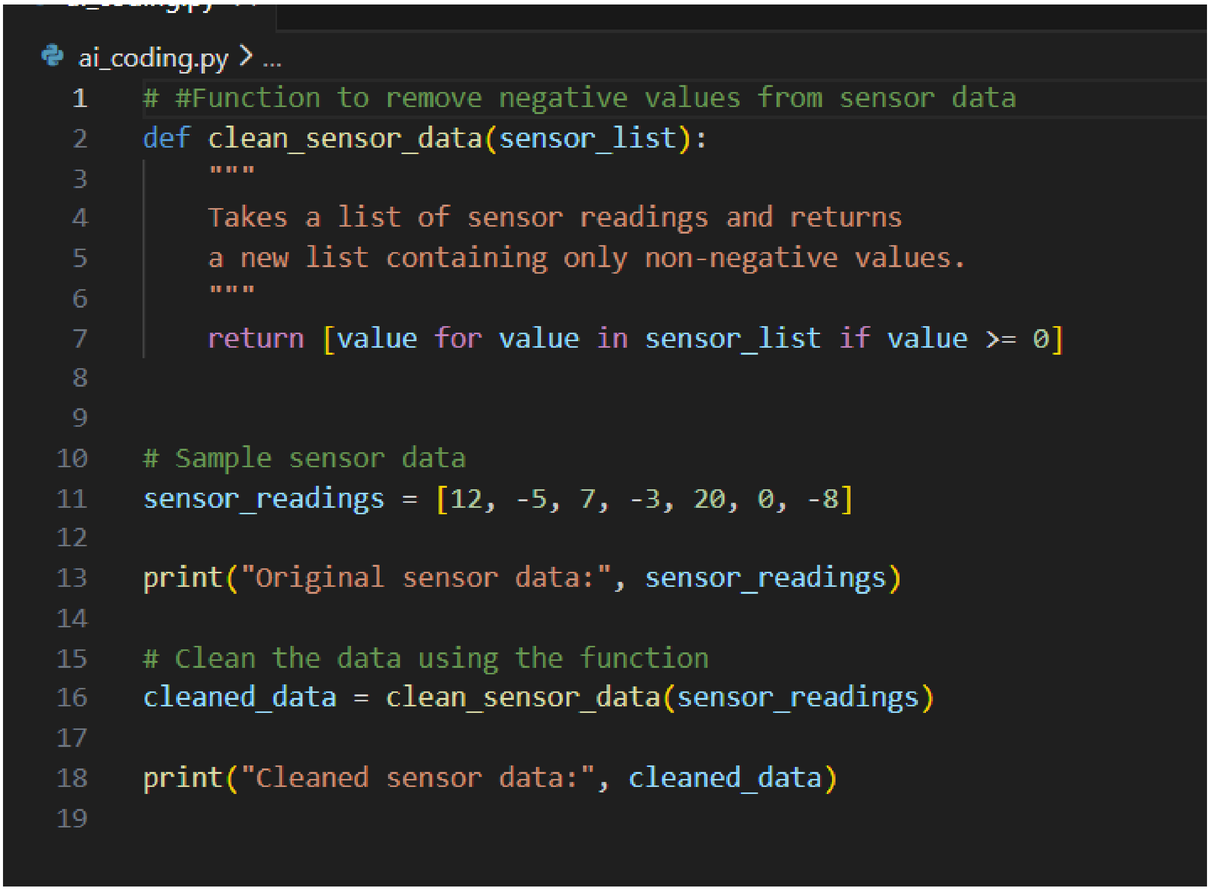
     BNo_StudentName_AI_Coding_A1.

     Sample: **B10_Rohit_22A523421_A1**
4. Fill all the entries mentioned on top section.
5. Mention your AWS Academy Virtual Lab Account details as shown in the next page.
6. **Don’t write on this page.**
7. All answers should be answered from next page only.

Lab assignment 2.2:

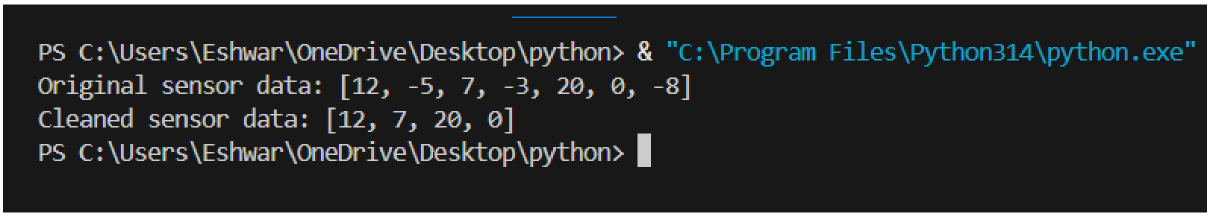
Prompt:

Cleaning Sensor Data:



```python
# #Function to remove negative values from sensor data
def clean_sensor_data(sensor_list):
    """
    Takes a list of sensor readings and returns
    a new list containing only non-negative values.
    """
    return [value for value in sensor_list if value >= 0]


# Sample sensor data
sensor_readings = [12, -5, 7, -3, 20, 0, -8]

print("Original sensor data:", sensor_readings)

# Clean the data using the function
cleaned_data = clean_sensor_data(sensor_readings)

print("Cleaned sensor data:", cleaned_data)
```

Output:



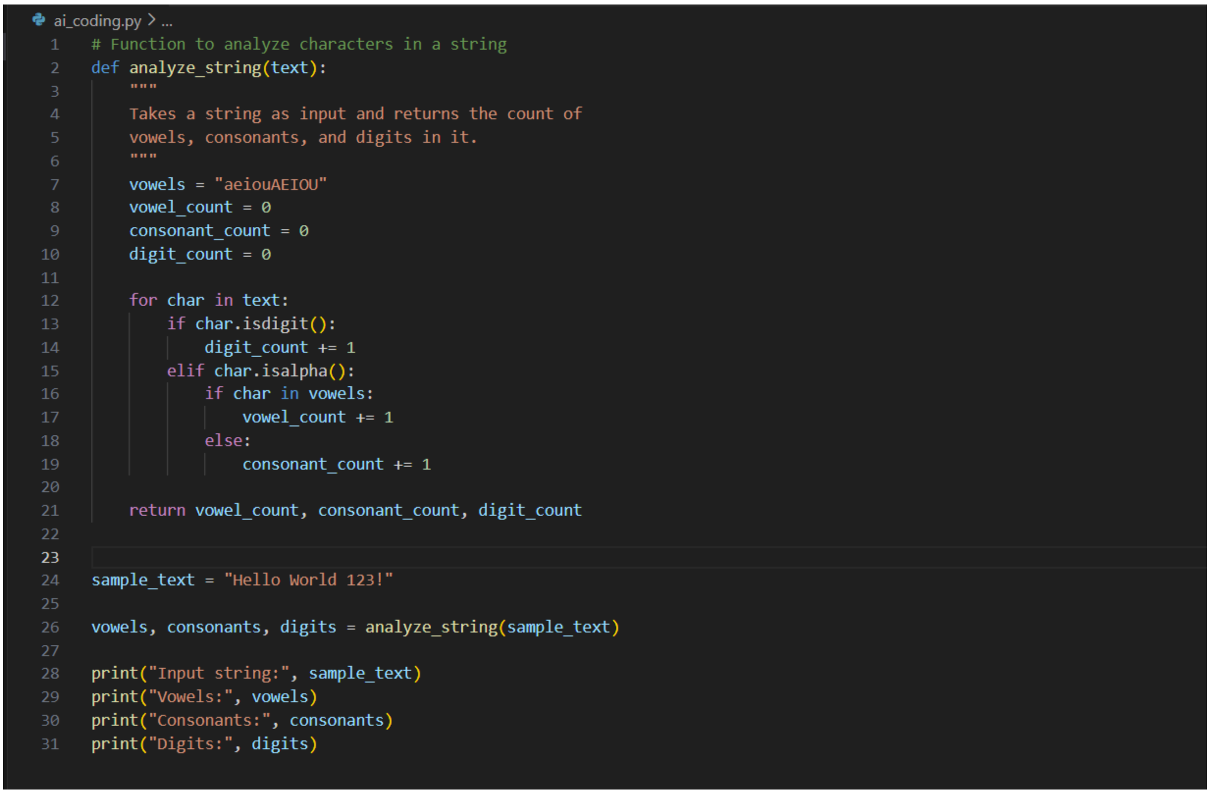
```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe"
Original sensor data: [12, -5, 7, -3, 20, 0, -8]
Cleaned sensor data: [12, 7, 20, 0]
PS C:\Users\Eshwar\OneDrive\Desktop\python> 
```

- **The function clean_sensor_data takes a list of sensor readings and removes negative values using list comprehension.**
- **It returns a new list containing only valid (non-negative) readings.**
- **This ensures data quality before further processing or analysis**

Task 2:

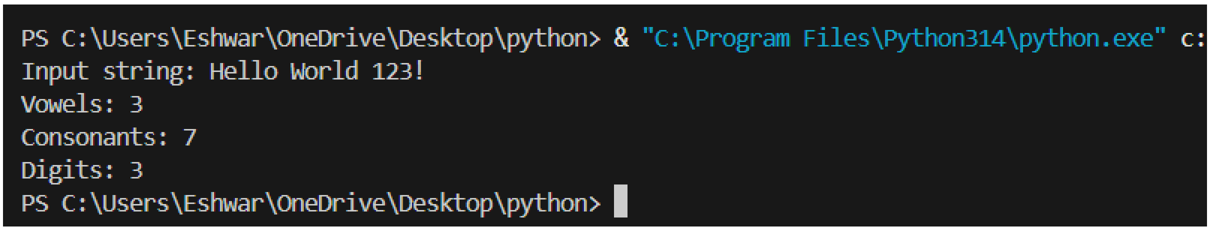
Prompt:

String Character Analysis



```python
# Function to analyze characters in a string
def analyze_string(text):
    """
    Takes a string as input and returns the count of
    vowels, consonants, and digits in it.
    """
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    digit_count = 0

    for char in text:
        if char.isdigit():
            digit_count += 1
        elif char.isalpha():
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count, digit_count


sample_text = "Hello World 123!"

vowels, consonants, digits = analyze_string(sample_text)

print("Input string:", sample_text)
print("Vowels:", vowels)
print("Consonants:", consonants)
print("Digits:", digits)
```

Output:



```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:
Input string: Hello World 123!
Vowels: 3
Consonants: 7
Digits: 3
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- **The function analyze_string iterates over each character in the input string.**

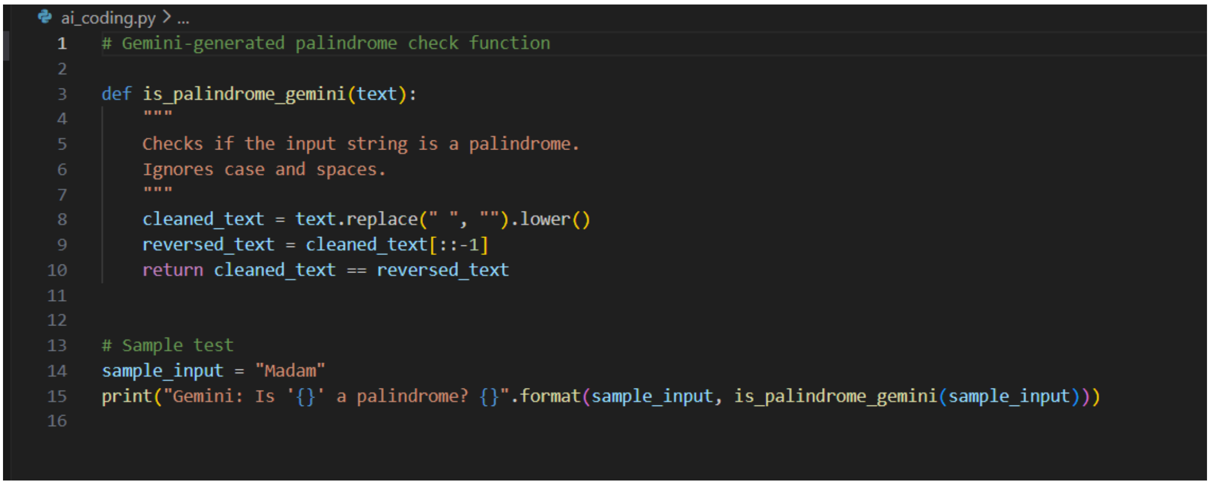
- **It increments counters for vowels, consonants, and digits based on character type**

- **Finally, it returns the counts, enabling text analysis for any given input.**

Task 3:

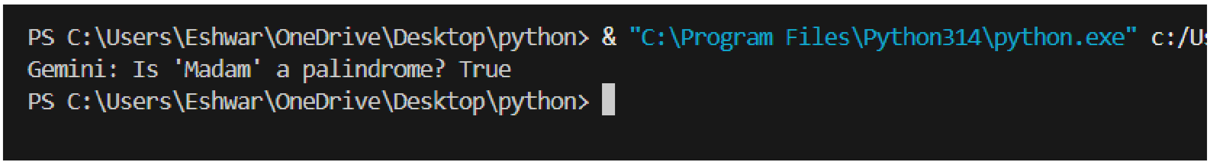
Prompt:

Palindrome Check – Tool Comparison:

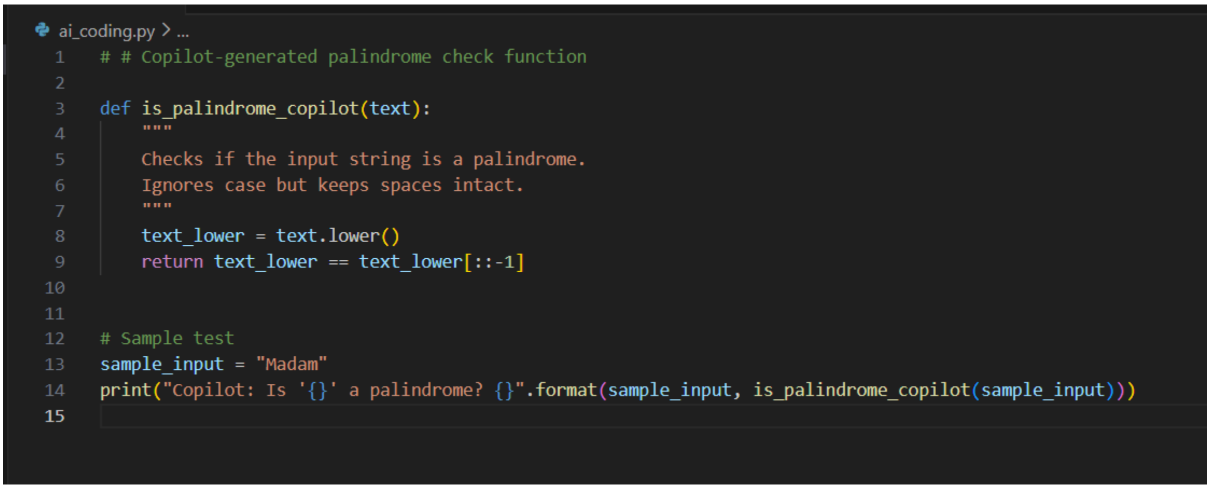


```python
# Gemini-generated palindrome check function

def is_palindrome_gemini(text):
    """
    Checks if the input string is a palindrome.
    Ignores case and spaces.
    """
    cleaned_text = text.replace(" ", "").lower()
    reversed_text = cleaned_text[::-1]
    return cleaned_text == reversed_text


# Sample test
sample_input = "Madam"
print("Gemini: Is '{}' a palindrome? {}".format(sample_input, is_palindrome_gemini(sample_input)))
```

Output:

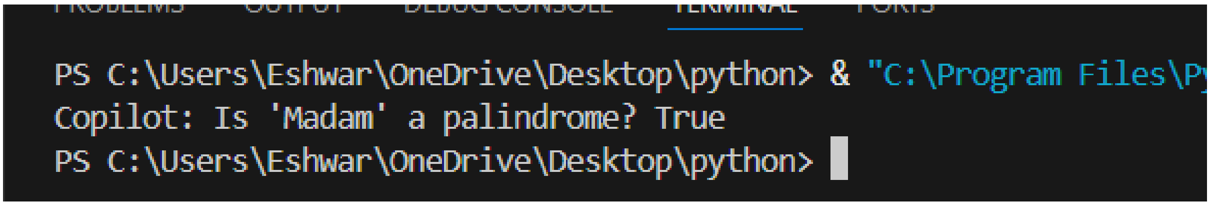


```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:/U
Gemini: Is 'Madam' a palindrome? True
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```



```python
# # Copilot-generated palindrome check function

def is_palindrome_copilot(text):
    """
    Checks if the input string is a palindrome.
    Ignores case but keeps spaces intact.
    """
    text_lower = text.lower()
    return text_lower == text_lower[::-1]


# Sample test
sample_input = "Madam"
print("Copilot: Is '{}' a palindrome? {}".format(sample_input, is_palindrome_copilot(sample_input)))
```

output:



```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\P
Copilot: Is 'Madam' a palindrome? True
PS C:\Users\Eshwar\OneDrive\Desktop\python>
```

- Both functions check whether a string reads the same forwards and backwards.

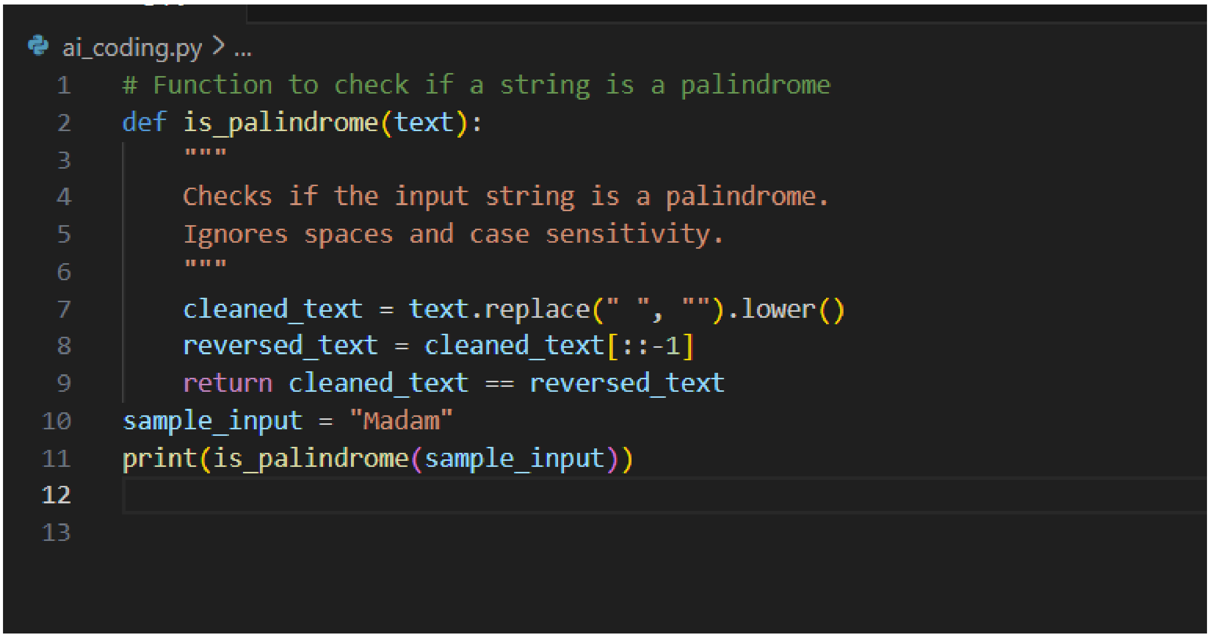
- Gemini separates cleaning and reversing for clarity, while Copilot uses a concise one-liner.

- Gemini is easier to understand, Copilot is shorter and better for quick deployment.

Task 4:

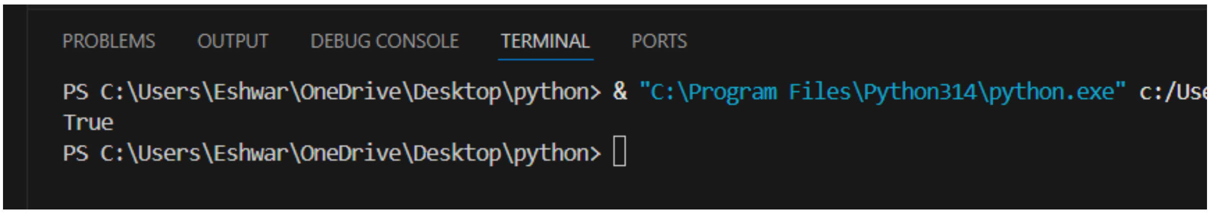
Prompt:

Code Explanation Using AI



```python
# Function to check if a string is a palindrome
def is_palindrome(text):
    """
    Checks if the input string is a palindrome.
    Ignores spaces and case sensitivity.
    """
    cleaned_text = text.replace(" ", "").lower()
    reversed_text = cleaned_text[::-1]
    return cleaned_text == reversed_text
sample_input = "Madam"
print(is_palindrome(sample_input))
```

Output:



```
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:/Use
True
PS C:\Users\Eshwar\OneDrive\Desktop\python> 
```

- **The function removes spaces and converts the string to lowercase.**

- **It then reverses the cleaned string and compares it with the original.**

- **It returns True if the string is a palindrome, otherwise False.**