

Varshini Girugula

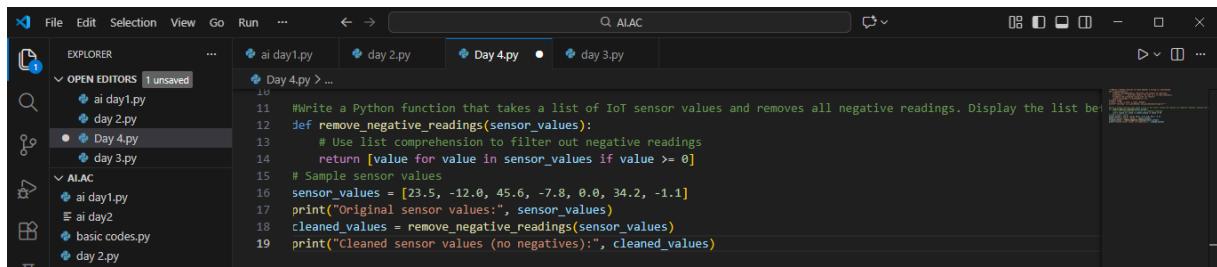
2403A51L14

B-51

ASSIGNMENT -2.2

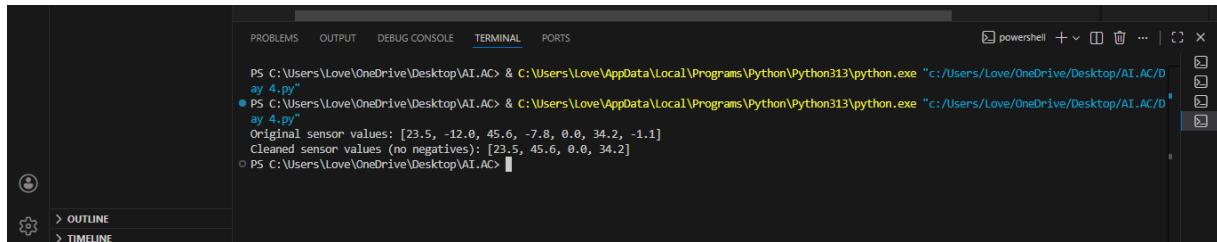
Task 1: Cleaning Sensor Data

PROMPT: Write a Python function that takes a list of IoT sensor values and removes all negative readings. Display the list before and after cleaning.



```
File Edit Selection View Go Run ... 🔍 AIAC
EXPLORER OPEN EDITORS 1 unsaved
ai day1.py day 2.py Day 4.py ● day 3.py
Day 4.py > ...
11  #Write a Python function that takes a list of IoT sensor values and removes all negative readings. Display the list before and after cleaning.
12  def remove_negative_readings(sensor_values):
13      # Use list comprehension to filter out negative readings
14      return [value for value in sensor_values if value >= 0]
15  # Sample sensor values
16  sensor_values = [23.5, -12.0, 45.6, -7.8, 0.0, 34.2, -1.1]
17  print("Original sensor values:", sensor_values)
18  cleaned_values = remove_negative_readings(sensor_values)
19  print("Cleaned sensor values (no negatives):", cleaned_values)
```

OUTPUT:



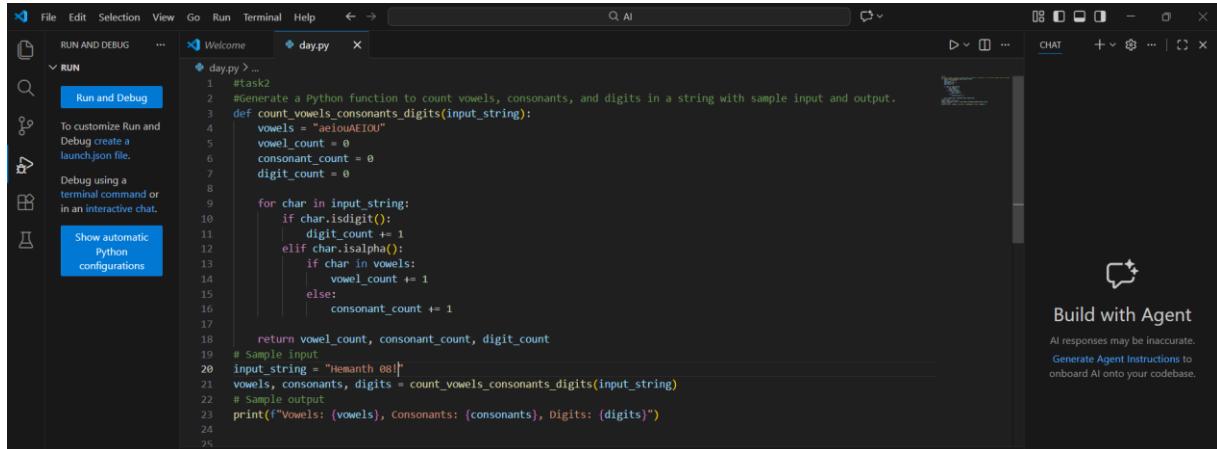
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell + ×
PS C:\Users\Love\OneDrive\Desktop\AI.AC> & C:\Users\Love\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/Love/OneDrive/Desktop/AI.AC/Day 4.py"
PS C:\Users\Love\OneDrive\Desktop\AI.AC> & C:\Users\Love\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/Love/OneDrive/Desktop/AI.AC\Day 4.py"
Original sensor values: [23.5, -12.0, 45.6, -7.8, 0.0, 34.2, -1.1]
Cleaned sensor values (no negatives): [23.5, 45.6, 0.0, 34.2]
```

EXPLANATION:

This function removes invalid negative sensor values using list comprehension. Only values greater than or equal to zero are retained, ensuring clean IoT sensor data.

Task 2: String Character Analysis

PROMPT: Generate a python function to count vowels, consonants and digits in a string with sample input and output.



The screenshot shows the Visual Studio Code interface. On the left, the 'RUN AND DEBUG' sidebar is open, showing options like 'Run and Debug', 'To customize Run and Debug create a launch.json file.', and 'Show automatic Python configurations'. The main code editor window displays a Python script named 'day.py' with the following content:

```
#task2
#Generate a Python function to count vowels, consonants, and digits in a string with sample input and output.
def count_vowels_consonants_digits(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    digit_count = 0

    for char in input_string:
        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 input
input_string = "Hemanth 89"
vowels, consonants, digits = count_vowels_consonants_digits(input_string)
# Sample output
print(f"Vowels: {vowels}, Consonants: {consonants}, Digits: {digits}")
```

To the right of the code editor is an AI sidebar titled 'Build with Agent'. It includes a message bubble icon and the text 'Build with Agent'.

OUTPUT:



The screenshot shows the 'Python Debug Console' tab. The output window displays the following command and its result:

```
PS C:\Users\girug\Downloads\AI> & 'c:\Users\girug\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\girug\vscode\extensions\ms-python.debug-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '52653' '--' 'C:\Users\girug\Downloads\AI\day.py'
Vowels: 2, Consonants: 5, Digits: 2
PS C:\Users\girug\Downloads\AI>
```

EXPLANATION:

The function iterates through each character and classifies it as a vowel, consonant, or digit.

Python string methods like `isalpha()` and `isdigit()` improve accuracy and readability.

Task 3: Palindrome Check – Tool Comparison

Gemini Prompt: Write a Python function to check if a string is a palindrome.
Ignore spaces and capitalization.

The screenshot shows a Jupyter Notebook interface in Google Colab. A code cell contains a Python function named `is_palindrome_gemini`. The function removes spaces and converts the text to lowercase, then compares it with its reverse. Below the function, several print statements show sample inputs and their outputs, demonstrating the function's ability to correctly identify palindromes like 'Racecar', 'A man a plan a canal Panama', and '12321' while correctly identifying non-palindromes like 'Hello World' and 'Python'.

```
def is_palindrome_gemini(text):
    # Remove spaces and convert to lowercase
    processed_text = ''.join([char.lower() for char in text if char.isalnum()])

    # Compare the processed string with its reverse
    return processed_text == processed_text[::-1]

# Sample input and output:
print("Is 'Racecar' a palindrome?", is_palindrome_gemini("Racecar"))
print("Is 'A man a plan a canal Panama' a palindrome?", is_palindrome_gemini("A man a plan a canal Panama"))
print("Is 'Hello World' a palindrome?", is_palindrome_gemini("Hello world"))
print("Is 'No lemon, no melon' a palindrome?", is_palindrome_gemini("No lemon, no melon"))
print("Is '12321' a palindrome?", is_palindrome_gemini("12321"))
print("Is 'Python' a palindrome?", is_palindrome_gemini("Python"))
```

OUTPUT:

The screenshot shows the output of the code execution. The results are displayed in a code cell, showing the function's response to each sample input. The outputs are:

- Is 'Racecar' a palindrome? True
- Is 'A man a plan a canal Panama' a palindrome? True
- Is 'Hello World' a palindrome? False
- Is 'No lemon, no melon' a palindrome? True
- Is '12321' a palindrome? True
- Is 'Python' a palindrome? False

Copilot Prompt: Generate a python function to check whether a string is a palindrome (ignore case and spaces) and provide sample input and output.

The screenshot shows the Microsoft Visual Studio Code interface. A Python file named `day.py` is open. The code is a generated function `is_palindrome` that removes spaces, converts to lowercase, and checks if the string is equal to its reverse. It includes sample inputs and outputs for testing.

```
#Generate a Python function to check whether a string is a palindrome (ignore case and spaces) and provide sample input and output
def is_palindrome(s):
    # Remove spaces and convert to lowercase
    cleaned_string = ''.join(s.split()).lower()
    # check if the cleaned string is equal to its reverse
    return cleaned_string == cleaned_string[::-1]

# Sample input and output
sample_input = "A man a plan a canal Panama"
print(f"Input: '{sample_input}'")
print(f"Is palindrome: {is_palindrome(sample_input)}")
# Output: True
sample_input2 = "Your Mr World"
print(f"Input: '{sample_input2}'")
print(f"Is palindrome: {is_palindrome(sample_input2)}")
# Output: False
```

OUTPUT:

The screenshot shows a terminal window titled "Python Debug Console" with the following output:

```
PS C:\Users\girug\Downloads\AI> & ':::users\girug\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\girug\vscode\extensions\ms-python_debugger-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '59755' '--' 'c:\Users\girug\Downloads\AI\day.py'
Input: 'A man a plan a canal Panama'
Is palindrome: True
Input: 'Your mm World'
Is palindrome: False
PS C:\Users\girug\Downloads\AI>
```

The terminal also displays a "BREAKPOINTS" sidebar with three items: "Raised Except...", "Uncaught Except..." (which is checked), and "User Uncaught...".

Comparison Table:

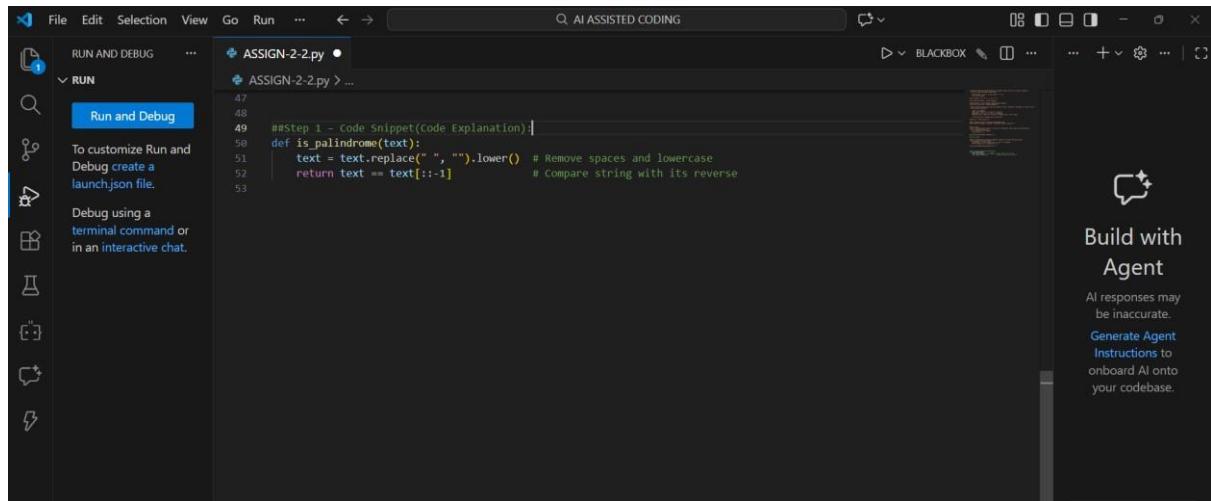
Feature	Gemini	Copilot
Clarity	Simple, minimal code	Slightly longer, more robust
Handling spaces/case	Ignores spaces, converts to lowercase	Ignores spaces and punctuation, lowercase
Readability	Very clear	Clear, slightly more detailed
Efficiency	Uses string slicing	Uses string comprehension

EXPLANATION:

Gemini provides concise and easy-to-read logic, making it beginnerfriendly. Copilot generates more robust code that handles punctuation and special characters.

Task 4: Code Explanation Using AI Step 1 –

Code Snippet:



The screenshot shows a code editor interface with a dark theme. The file being edited is `ASSIGN-2-2.py`. The code contains the following Python function:

```
47
48
49 ##Step 1 - Code Snippet(Code Explanation):
50 def is_palindrome(text):
51     text = text.replace(" ", "").lower() # Remove spaces and lowercase
52     return text == text[::-1] # Compare string with its reverse
53
```

The code is annotated with AI-generated comments. The first line of the function is preceded by `##Step 1 - Code Snippet(Code Explanation):`. The line `text = text.replace(" ", "").lower()` is annotated with `# Remove spaces and lowercase`. The line `return text == text[::-1]` is annotated with `# Compare string with its reverse`.

Step 2 – AI Explanation:

1. `text.replace(" ", "").lower()` → Removes spaces and converts letters to lowercase.
2. `text == text[::-1]` → Checks if the string is equal to its reverse.

EXPLANATION:

The function normalizes the string to avoid case and space mismatches. It then compares the string with its reverse to verify palindrome logic.