

Varshini Girugula

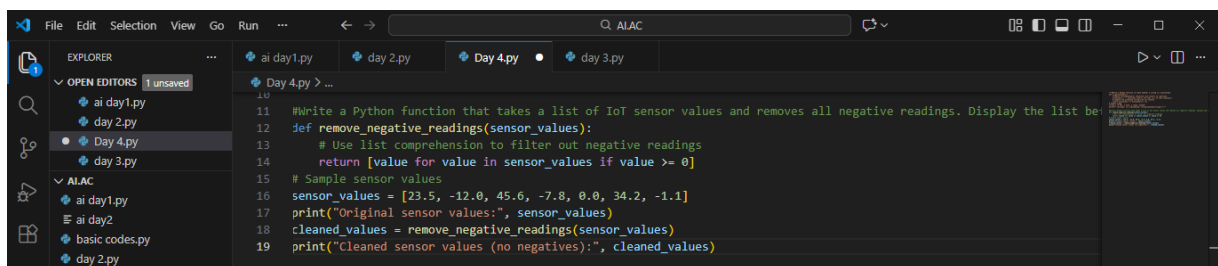
2403A51L14

B-51

## ASSIGNMENT -2.2

### Task 1: Cleaning Sensor Data

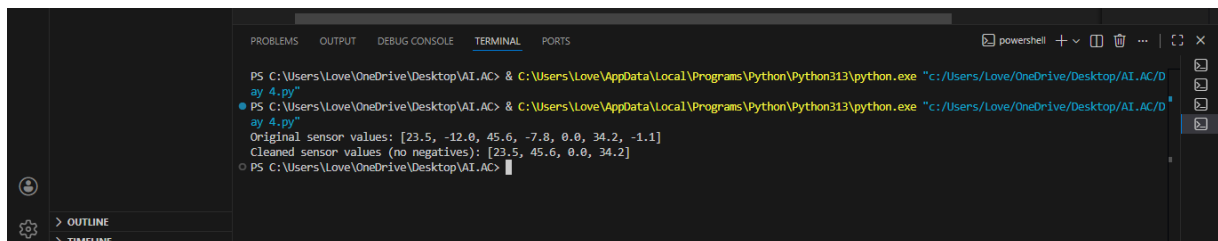
**PROMPT:** Write a Python function that takes a list of IoT sensor values and removes all negative readings. Display the list before and after cleaning.



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'ALAC' with files 'ai day1.py', 'day 2.py', 'Day 4.py', and 'day 3.py'. The code editor shows the following Python code:

```
10
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:



The screenshot shows a terminal window with the following output:

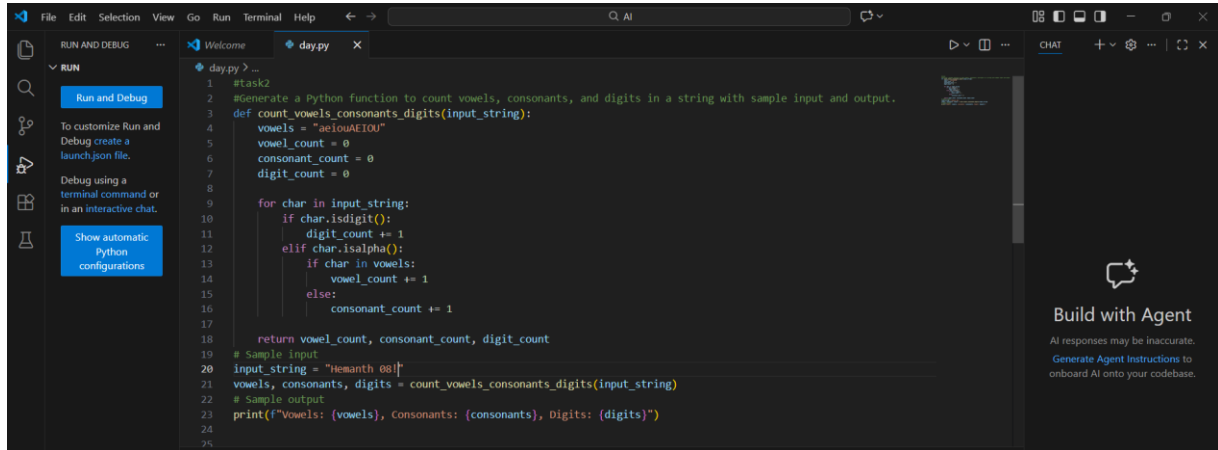
```
PS C:\Users\Love\OneDrive\Desktop\AI.AC> & C:\Users\Love\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/Love/OneDrive/Desktop/AI.AC/Day 4.py"
Original sensor values: [23.5, -12.0, 45.6, -7.8, 0.0, 34.2, -1.1]
Cleaned sensor values (no negatives): [23.5, 45.6, 0.0, 34.2]
```

### 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

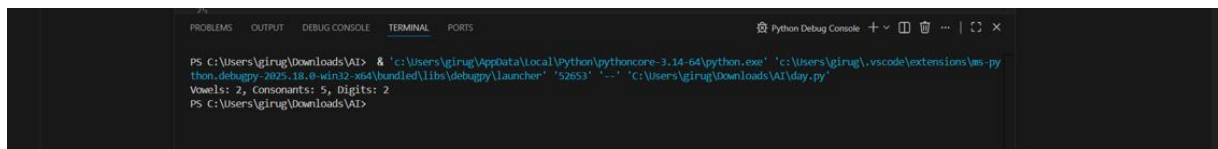
**PROMPT:** Generate a python function to count vowels, consonants and digits in a string with sample input and output.



The screenshot shows a VS Code editor with a Python file named `day.py`. The code defines a function `count_vowels_consonants_digits` that takes an `input_string` and returns a tuple of counts for vowels, consonants, and digits. The function uses `isalpha()` to check for vowels and `isdigit()` to check for digits. A sample input string "Hemanth 08j" is used to demonstrate the function's output.

```
1 #Task2
2 #Generate a Python function to count vowels, consonants, and digits in a string with sample input and output.
3 def count_vowels_consonants_digits(input_string):
4     vowels = "aeiouAEIOU"
5     vowel_count = 0
6     consonant_count = 0
7     digit_count = 0
8
9     for char in input_string:
10         if char.isdigit():
11             digit_count += 1
12         elif char.isalpha():
13             if char in vowels:
14                 vowel_count += 1
15             else:
16                 consonant_count += 1
17
18     return vowel_count, consonant_count, digit_count
19
20 # Sample input
21 input_string = "Hemanth 08j"
22 vowels, consonants, digits = count_vowels_consonants_digits(input_string)
23 # Sample output
24 print(f"Vowels: {vowels}, Consonants: {consonants}, Digits: {digits}")
25
```

## OUTPUT:



The screenshot shows a terminal window with the following output:

```
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.debugpy-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>
```

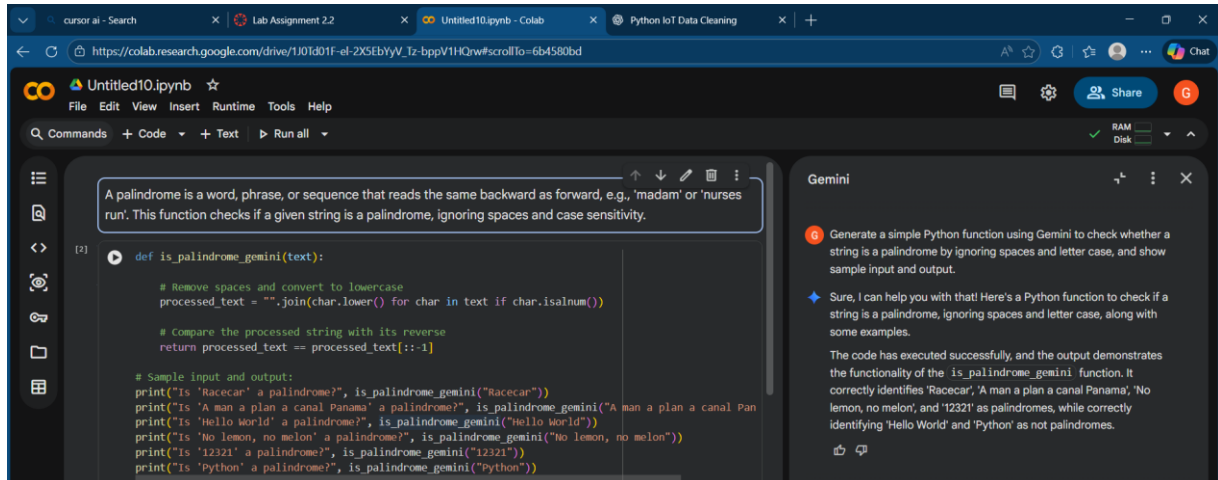
## EXPLANATION:

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 Google Colab notebook titled 'Untitled10.ipynb'. The code cell contains a Python function `is_palindrome_gemini(text)` that removes spaces and converts the string to lowercase, then compares it with its reverse. Below the function, sample inputs and outputs are printed. To the right, a Gemini chat window is open, showing a prompt to generate a simple Python function and a response that provides the function and explains its functionality.

```
[2] 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 Pan  
    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"))
```

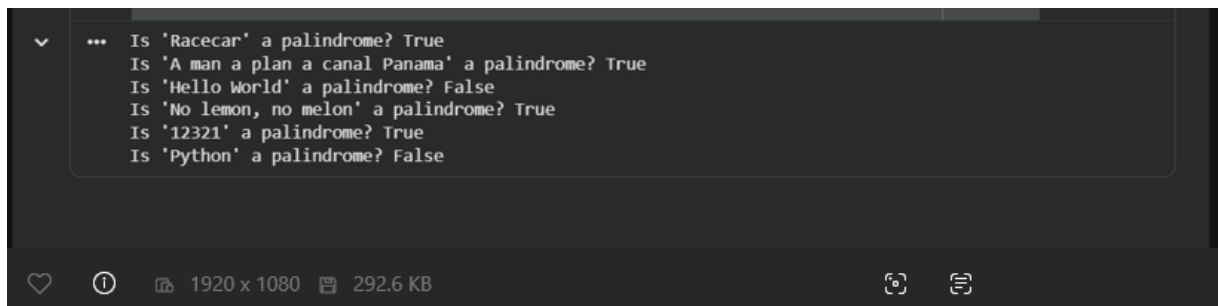
**Gemini**

Generate a simple Python function using Gemini to check whether a string is a palindrome by ignoring spaces and letter case, and show sample input and output.

Sure, I can help you with that! Here's a Python function to check if a string is a palindrome, ignoring spaces and letter case, along with some examples.

The code has executed successfully, and the output demonstrates the functionality of the `is_palindrome_gemini` function. It correctly identifies 'Racecar', 'A man a plan a canal Panama', 'No lemon, no melon', and '12321' as palindromes, while correctly identifying 'Hello World' and 'Python' as not palindromes.

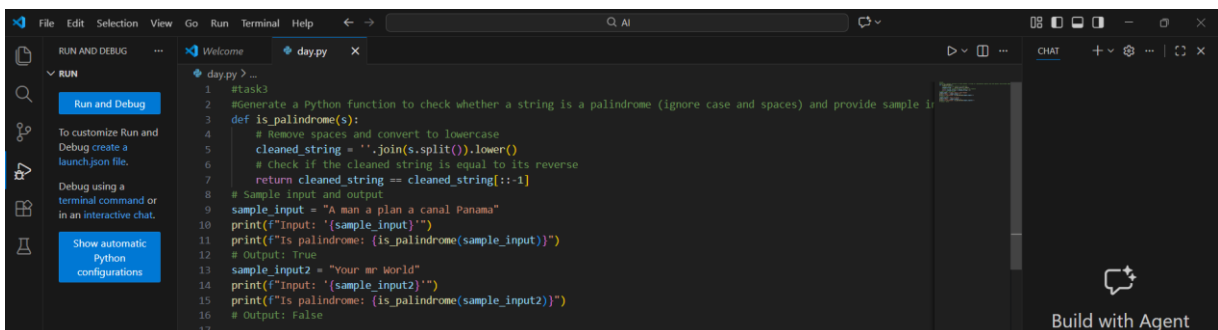
**OUTPUT:**



The screenshot shows the output of the Python function `is_palindrome_gemini`. It displays a series of questions and answers, confirming that 'Racecar', 'A man a plan a canal Panama', 'No lemon, no melon', and '12321' are palindromes, while 'Hello World' and 'Python' are not.

```
... 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
```

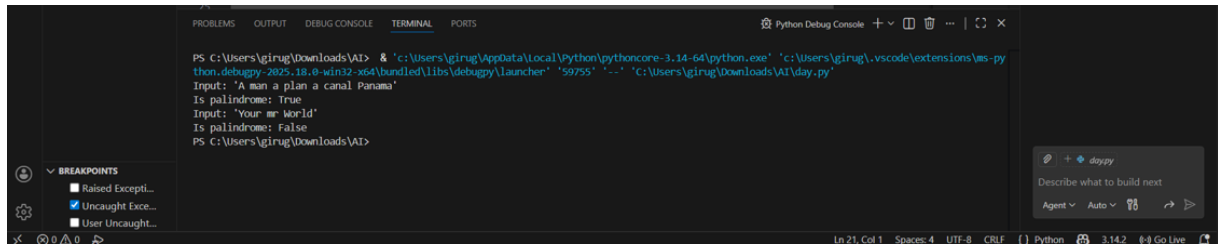
**Copilot Prompt:** Generate a python function to check whether a string is a palindrome (ignore case and spaces) and provide sample input and output.



The screenshot shows a Visual Studio Code editor with a file named `day.py`. The code cell contains a Python function `is_palindrome(s)` that removes spaces and converts the string to lowercase, then compares it with its reverse. Below the function, sample inputs and outputs are printed.

```
1 #task3  
2 #Generate a Python function to check whether a string is a palindrome (ignore case and spaces) and provide sample i  
3 def is_palindrome(s):  
4     # Remove spaces and convert to lowercase  
5     cleaned_string = "".join(s.split()).lower()  
6     # check if the cleaned string is equal to its reverse  
7     return cleaned_string == cleaned_string[::-1]  
8  
9 # Sample input and output  
10 sample_input = "A man a plan a canal Panama"  
11 print(f"Input: '{sample_input}'")  
12 print(f"Is palindrome: (is_palindrome(sample_input))")  
13 # Output: True  
14 sample_input2 = "Your mr World"  
15 print(f"Input: '{sample_input2}'")  
16 print(f"Is palindrome: (is_palindrome(sample_input2))")  
17 # Output: False
```

## OUTPUT:



```
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.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '59755' '--' 'C:\Users\girug\Downloads\AI\day.py'
Input: 'A man a plan a canal Panama'
Is palindrome: True
Input: 'Your mr World'
Is palindrome: False
PS C:\Users\girug\Downloads\AI>
```

## Comparison Table:

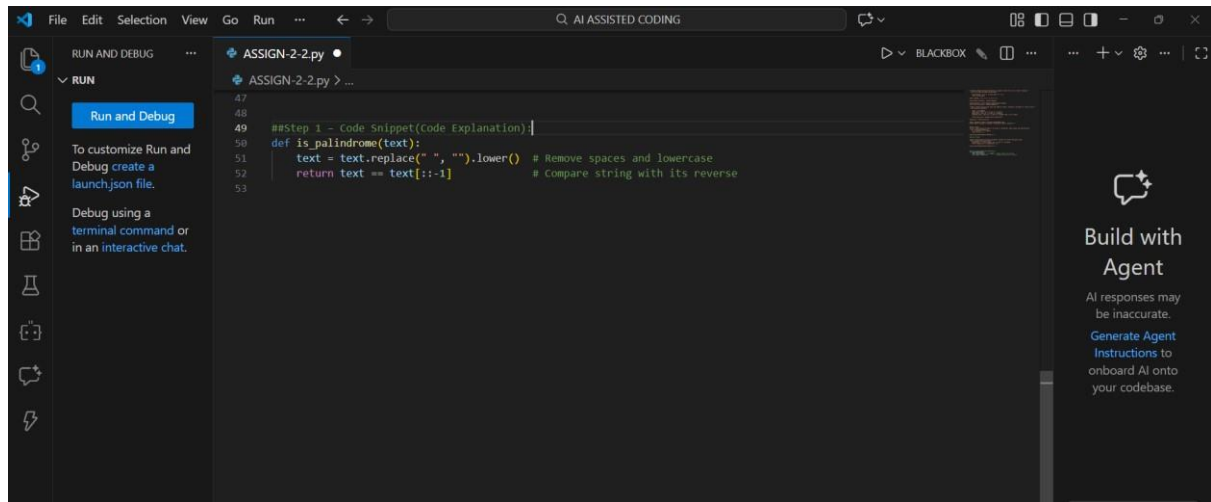
Feature	Gemini	Copilot
Clarity	Simple, minimal code	Slightly longer, more robust
Handling spaces/case	Ignores spaces, converts to lowercase	Ignores spaces and punctuation, lowercase
Readability	Very clear	Clear, slightly more detailed
Efficiency	Uses string slicing	Uses string comprehension

## 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:



### 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.