

AI-ASSISTED CODING ASSIGNMENT 2

SHRUTHI ADDAGUDI | 2303A52133 | BATCH 37



The screenshot shows a code editor with a Python function `remove_negative_values` that filters out negative integers from a list. The function uses a list comprehension. Below the function, an example usage is provided with a list `[1, -2, 3, 0, -5, 6, -1]`. The output shows the original list and the cleaned list `[1, 3, 0, 6]`. To the right, the Gemini chat window shows a prompt to write a function that removes negative values, followed by a detailed response explaining the logic and showing the execution results.

```
def remove_negative_values(integer_list):  
    """  
    Removes all negative values from a list of integers.  
    Args:  
        integer_list (list): A list of integers.  
    Returns:  
        list: A new list containing only the non-negative integers.  
    """  
    return [num for num in integer_list if num >= 0]  
  
# Example usage:  
my_list = [1, -2, 3, 0, -5, 6, -1]  
print("Original list:", my_list)  
  
cleaned_list = remove_negative_values(my_list)  
  
print("List after removing negative values:", cleaned_list)
```

Original list: [1, -2, 3, 0, -5, 6, -1]
List after removing negative values: [1, 3, 0, 6]

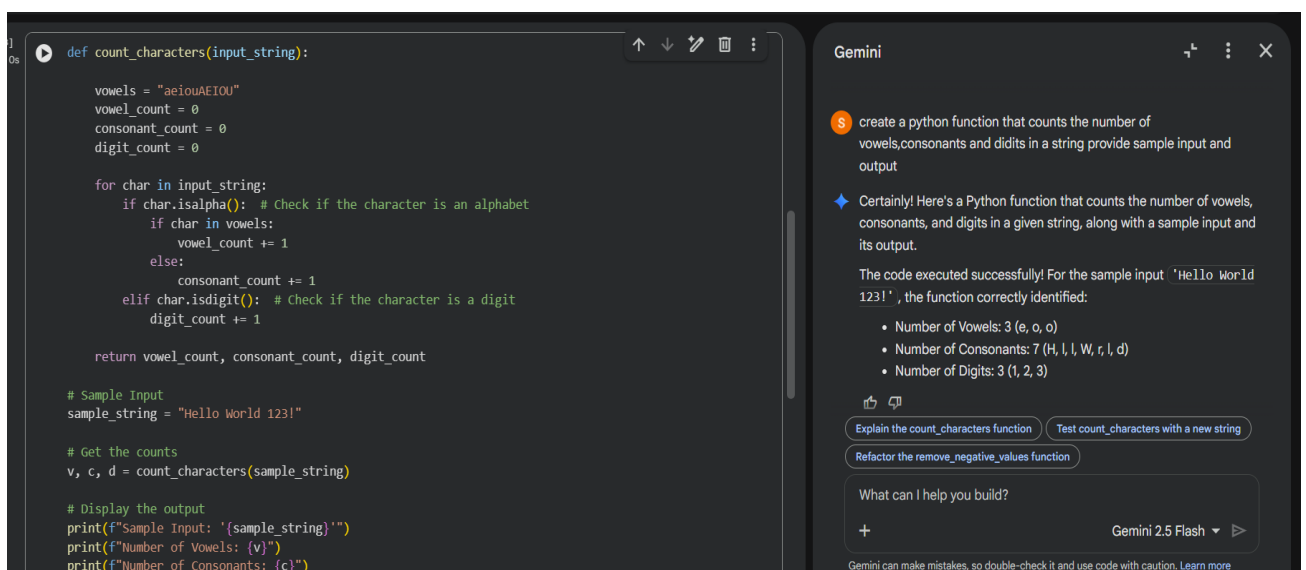
Task 1

Prompt Used : Write a Python function that removes all negative values from a list of integers.

Output : Before : [1, -2, 3, 0, -5, 6, -1]

After : [1, 3, 0, 6]

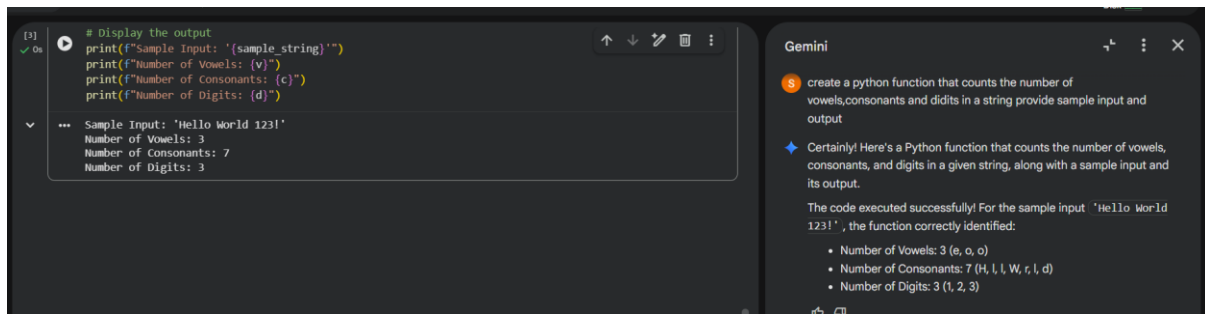
Understanding: Gemini generated a clean and efficient function using lists. The logic was easy to understand and it required no need for manual correction.



The screenshot shows a code editor with a Python function `count_characters` that counts the number of vowels, consonants, and digits in a string. The function uses a loop to iterate through each character and check its type. Below the function, a sample input string "Hello World 123!" is provided, and the output shows the counts for vowels (3), consonants (7), and digits (3). To the right, the Gemini chat window shows a prompt to create a function that counts these characters, followed by a detailed response explaining the logic and showing the execution results.

```
def count_characters(input_string):  
    vowels = "aeiouAEIOU"  
    vowel_count = 0  
    consonant_count = 0  
    digit_count = 0  
  
    for char in input_string:  
        if char.isalpha(): # Check if the character is an alphabet  
            if char in vowels:  
                vowel_count += 1  
            else:  
                consonant_count += 1  
        elif char.isdigit(): # Check if the character is a digit  
            digit_count += 1  
  
    return vowel_count, consonant_count, digit_count  
  
# Sample Input  
sample_string = "Hello World 123!"  
  
# Get the counts  
v, c, d = count_characters(sample_string)  
  
# Display the output  
print(f"Sample Input: '{sample_string}'")  
print(f"Number of Vowels: {v}")  
print(f"Number of Consonants: {c}")
```

Sample Input: 'Hello World 123!'
Number of Vowels: 3
Number of Consonants: 7
Number of Digits: 3



Task 2

Prompt Used : Create a Python function that counts the number of vowels, consonants, and digits in a string.

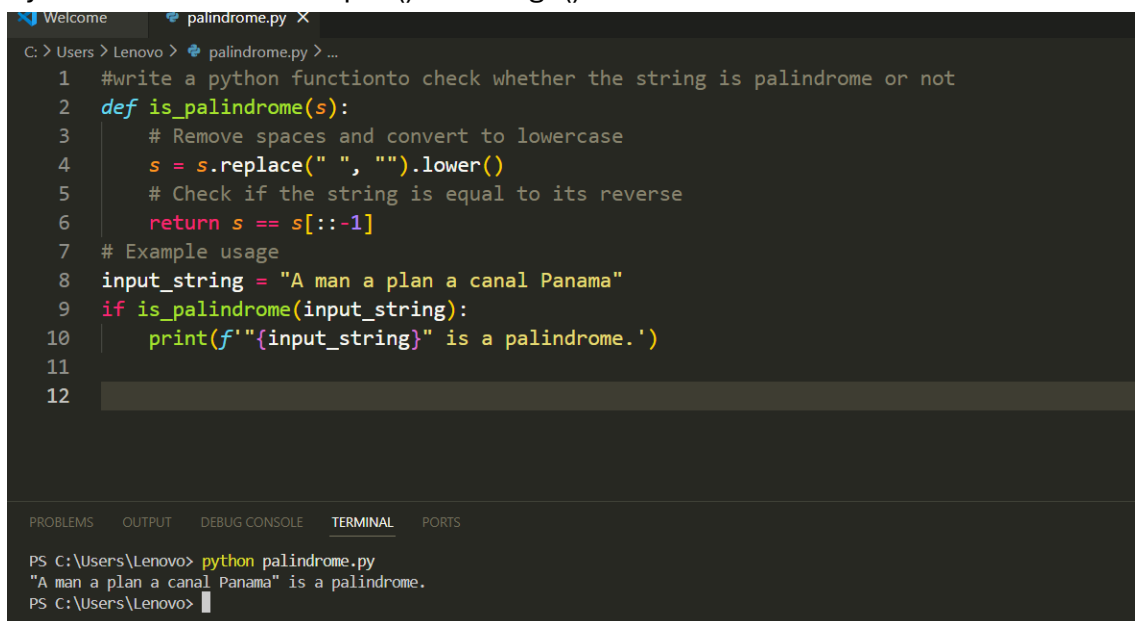
Sample Input: 'Hello World 123!'

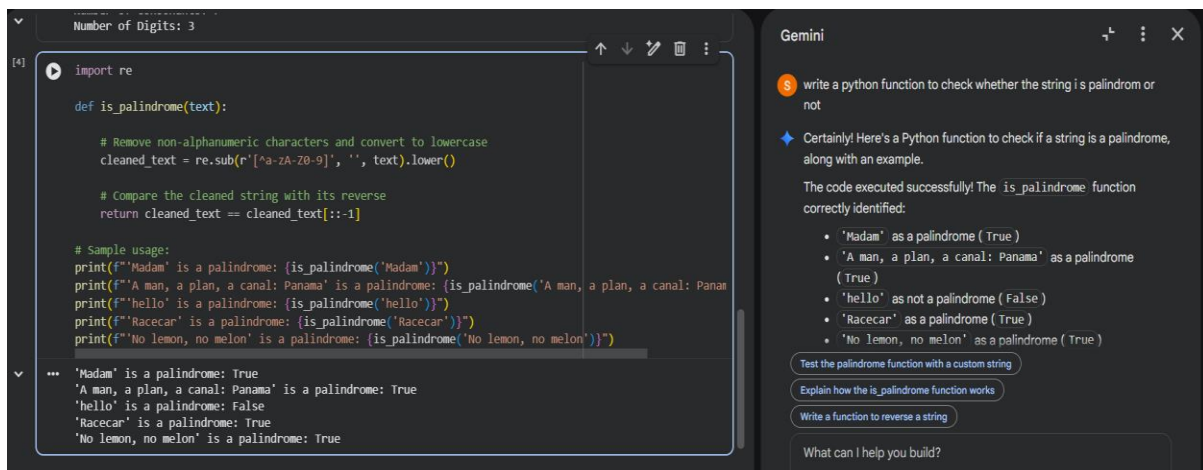
Number of Vowels: 3

Number of Consonants: 7

Number of Digits: 3

Student Comment: Gemini clearly handled character classification using built-in Python methods like `isalpha()` and `isdigit()` metho

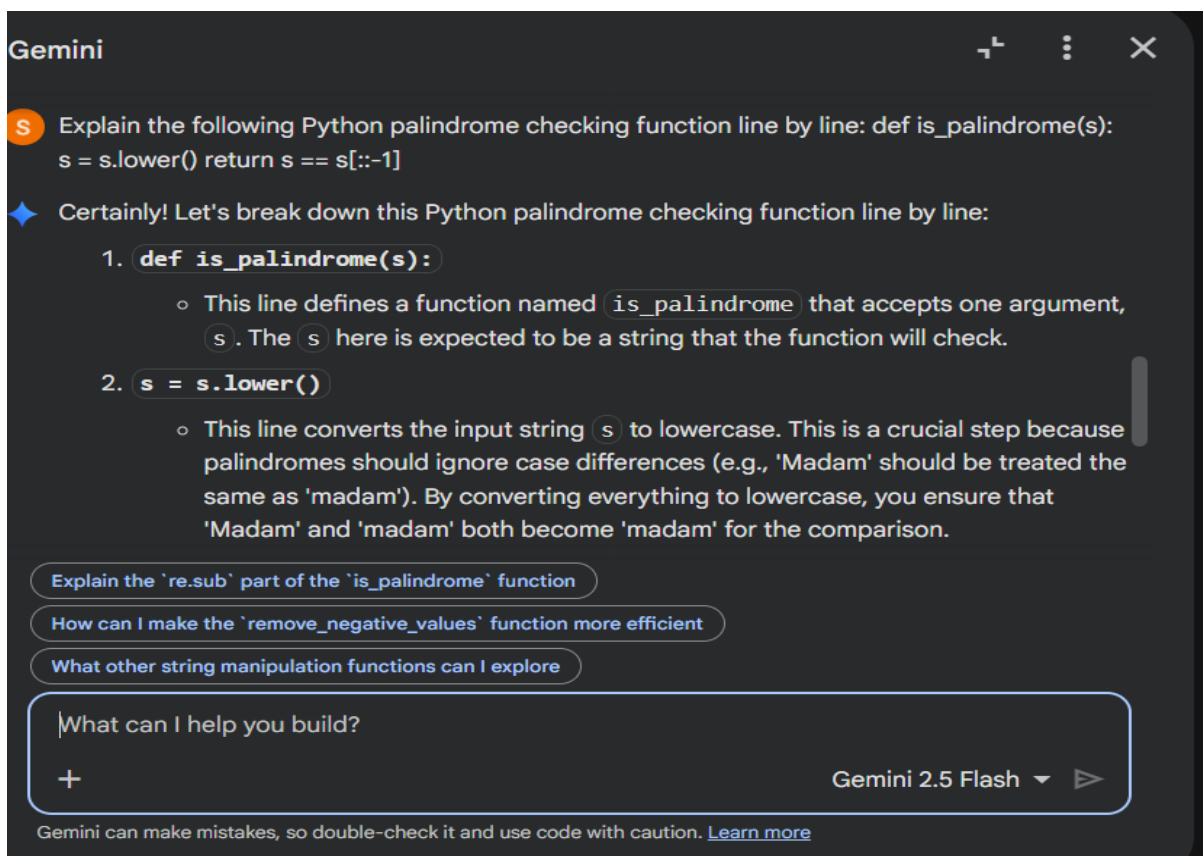


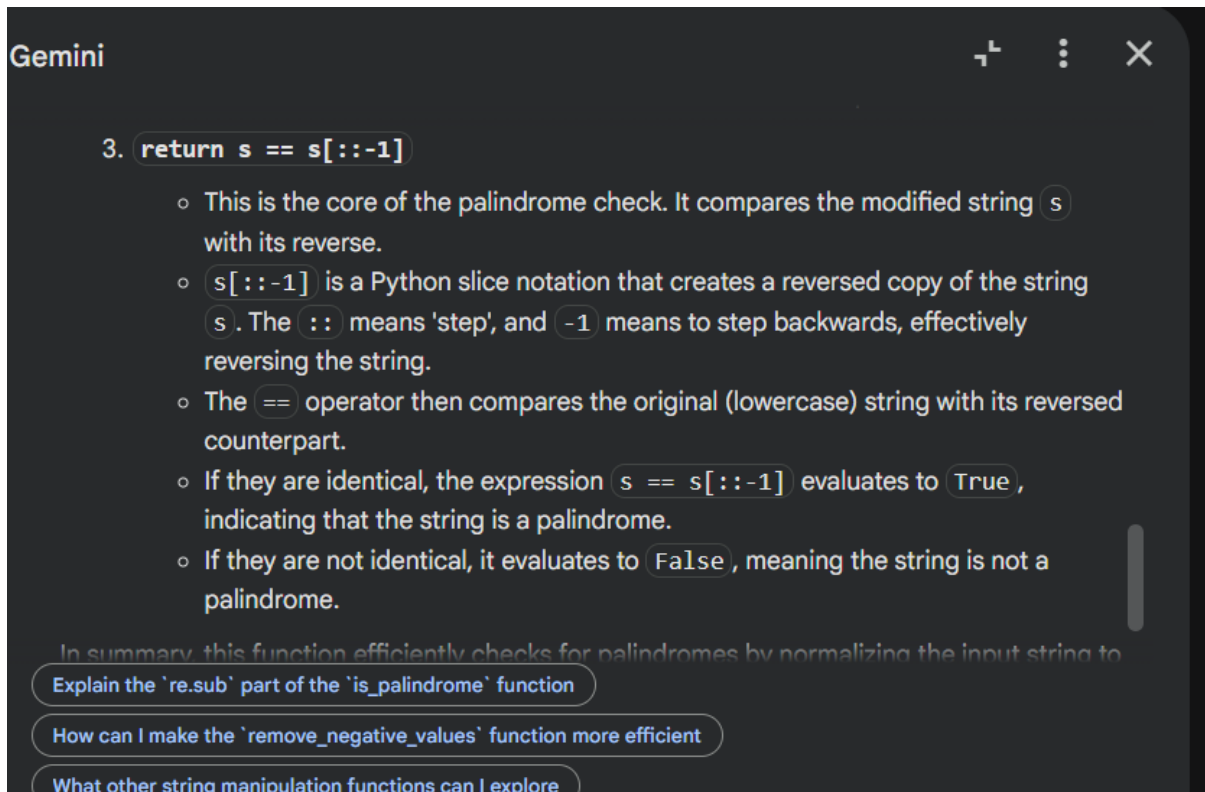


TASK 3:

Palindrome Check – Tool Comparison Prompt Used: Write a Python function to check whether a string is a palindrome.

Tool	Code quality	readability
Gemini	Clear and beginner friendly	Explains logic
copilot	Short and optimised	Assumes prior python knowledge





TASK 4

Code Explanation Using Gemini Prompt Used: Explain the following Python palindrome checking function line by line

`is_palindrome(s): s = s.lower() return s == s[::-1]` The explanation helped me understand slicing operations and how reversing of a string works. Gemini explanation was clear and it was easy to follow

Conclusion

In this lab, I explored AI-assisted coding using Google Gemini in Google Colab. Gemini proved to be effective in generating beginner-friendly code accompanied by clear and detailed explanations. GitHub Copilot was useful in producing more optimized and concise solutions, while Cursor AI assisted in refining and refactoring the code for better structure and readability. Overall, this lab enhanced my understanding of how different AI tools support programmers at various skill levels and stages of the development process.

