

# AI ASSISTANT CODING ASSIGNMENT -

## 2

**NAME: visikamalla vishal**

**HT.NO: 2303A51442**

**BATCH: 21**

---

**LAB 2:**

**Exploring Additional AI Coding Tools beyond Copilot – Gemini (Colab)  
and Cursor AI**

**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.**

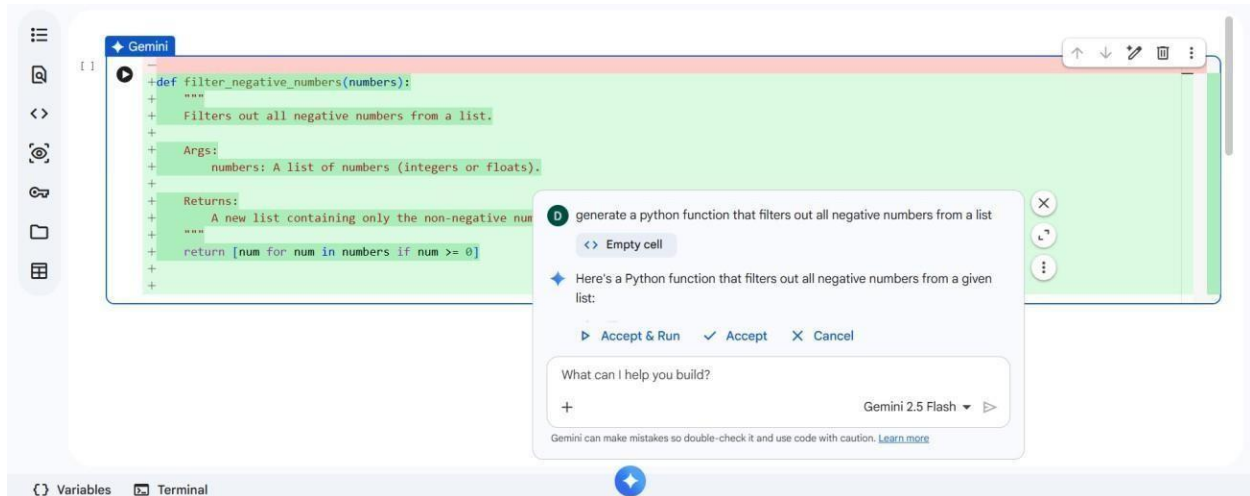
❖ **Expected Output:**

➤ **Before/after list**

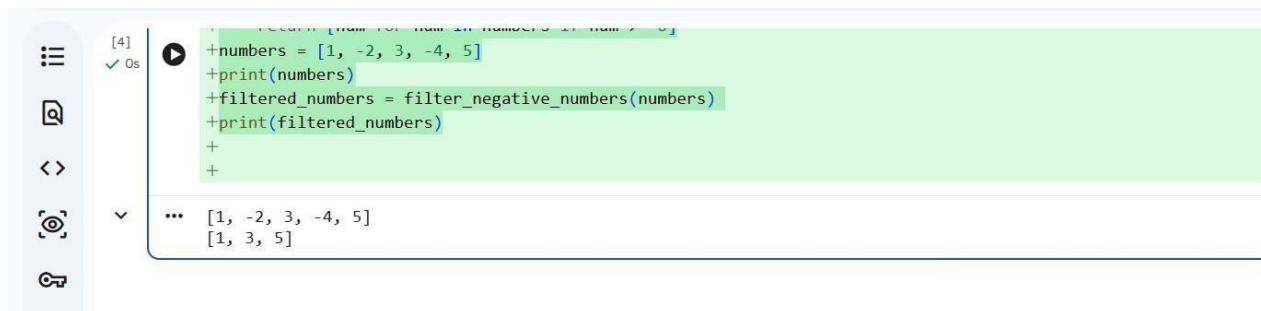
➤ **Screenshot of Colab execution**

---

**CODE :**



## OUTPUT:



## Task 2: String Character Analysis

### ❖ Scenario:

You are building a text-analysis feature.

### ❖ Task:

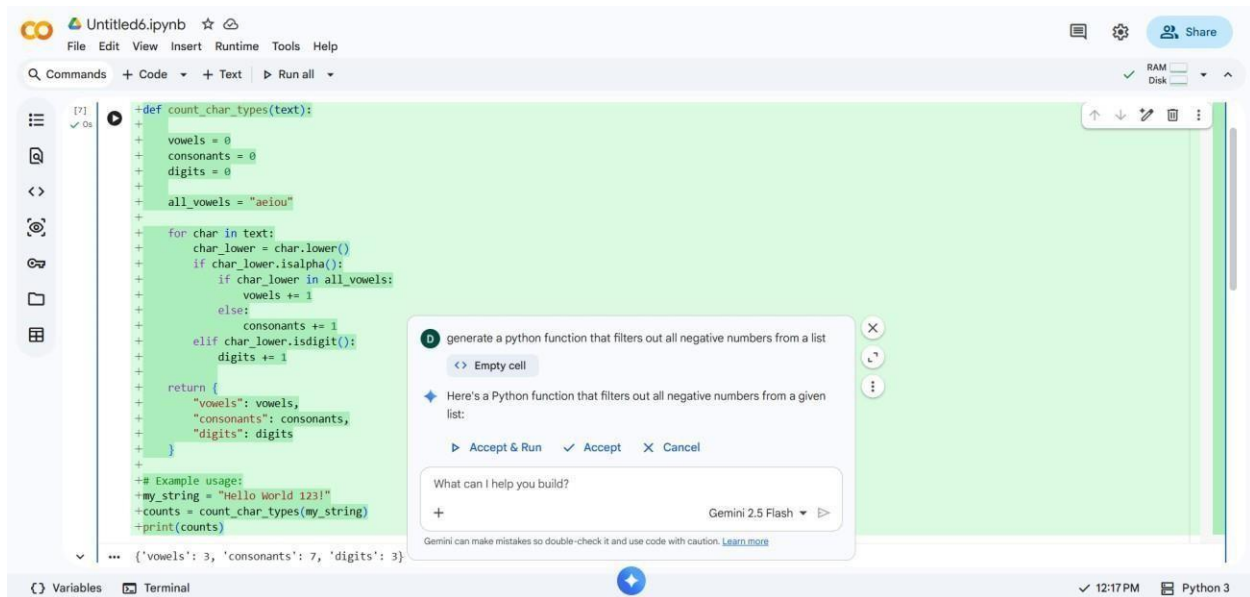
Use Gemini to generate a Python function that counts vowels, consonants, and digits in a string.

## ❖ Expected Output:

### ➤ Working function

### ➤ Sample inputs and outputs

## CODE :



The screenshot shows a Jupyter Notebook titled 'Untitled6.ipynb'. The code cell contains a function `count_char_types` that counts vowels, consonants, and digits in a given string. The function uses a loop to iterate through each character, checking if it is a vowel, consonant, or digit. The output of the function is a dictionary: `{'vowels': 3, 'consonants': 7, 'digits': 3}`. A Gemini AI chat window is overlaid on the code, showing a prompt to generate a Python function that filters out negative numbers from a list. The chat window also displays a response from Gemini 2.5 Flash, suggesting a function to filter out negative numbers from a list.

```
[7] ✓ On
def count_char_types(text):
    vowels = 0
    consonants = 0
    digits = 0
    all_vowels = "aeiou"
    for char in text:
        char_lower = char.lower()
        if char_lower.isalpha():
            if char_lower in all_vowels:
                vowels += 1
            else:
                consonants += 1
        elif char_lower.isdigit():
            digits += 1
    return {
        "vowels": vowels,
        "consonants": consonants,
        "digits": digits
    }

+ # Example usage:
+ my_string = "Hello World 123!"
+ counts = count_char_types(my_string)
+ print(counts)

... {'vowels': 3, 'consonants': 7, 'digits': 3}
```

generate a python function that filters out all negative numbers from a list

Empty cell

Here's a Python function that filters out all negative numbers from a given list:

Accept & Run Accept Cancel

What can I help you build?

Gemini 2.5 Flash

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

## OUTPUT:



The screenshot shows the output of the Python function `count_char_types`. The output is a dictionary: `{'vowels': 3, 'consonants': 7, 'digits': 3}`.

```
+ # Example usage:
+ my_string = "Hello World 123!"
+ counts = count_char_types(my_string)
+ print(counts)

... {'vowels': 3, 'consonants': 7, 'digits': 3}
```

## Task 3: Palindrome Check – Tool Comparison

### ❖ Scenario:

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

### ❖ Task:

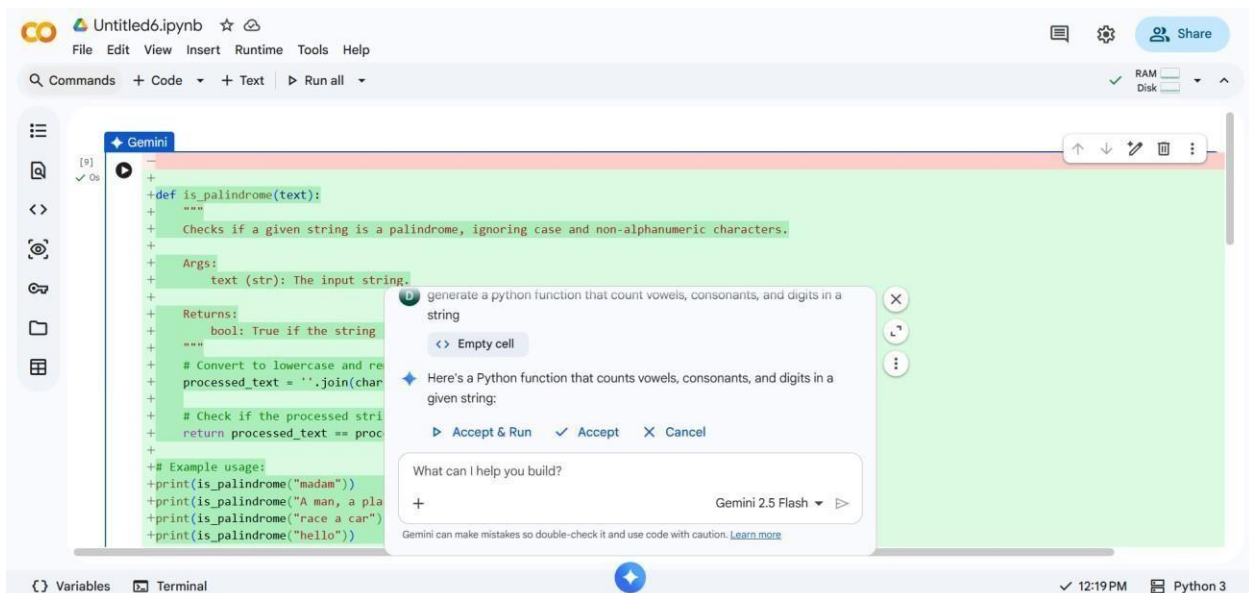
Generate a palindrome-checking function using Gemini and Copilot, then compare the results.

### ❖ Expected Output:

➤ Side-by-side code comparison

➤ Observations on clarity and structure

## CODE:



The screenshot shows a Jupyter Notebook interface with a file named 'Untitled6.ipynb'. The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with icons for commands, code, text, and running all cells. A sidebar on the left contains icons for file explorer, search, and other notebook functions. The main area displays a code cell with a green background, containing a Python function `is_palindrome` that checks if a string is a palindrome, ignoring case and non-alphanumeric characters. The function includes docstrings for its purpose, arguments, and return values, as well as example usage code. A Gemini AI chat window is overlaid on the code cell, showing a prompt to 'generate a python function that count vowels, consonants, and digits in a string' and a response providing a Python function for counting vowels, consonants, and digits. The chat window also includes buttons for 'Accept & Run', 'Accept', and 'Cancel', and a text input field for further interaction.

```
[9] ✓ Os
+ def is_palindrome(text):
+     """
+     Checks if a given string is a palindrome, ignoring case and non-alphanumeric characters.
+
+     Args:
+         text (str): The input string.
+
+     Returns:
+         bool: True if the string is a palindrome, False otherwise.
+     """
+     # Convert to lowercase and remove non-alphanumeric characters
+     processed_text = ''.join(char.lower() for char in text if char.isalnum())
+     # Check if the processed string is a palindrome
+     return processed_text == processed_text[::-1]
+
+ # Example usage:
+ print(is_palindrome("madam"))
+ print(is_palindrome("A man, a plan, a canal, Panama"))
+ print(is_palindrome("race a car"))
+ print(is_palindrome("hello"))
```

generate a python function that count vowels, consonants, and digits in a string

<> Empty cell

Here's a Python function that counts vowels, consonants, and digits in a given string:

Accept & Run Accept Cancel

What can I help you build?

Gemini 2.5 Flash

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

## OUTPUT:

```
+print(is_palindrome("A man, a plan, a canal: Panama")) # Output: True
+print(is_palindrome("race a car")) # Output: False
+print(is_palindrome("hello")) # Output: False
+
```

```
... True
... True
... False
... False
```

## 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.

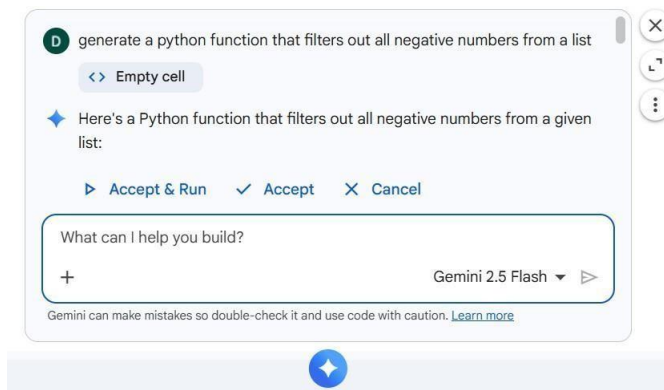
### ❖ Expected Output:

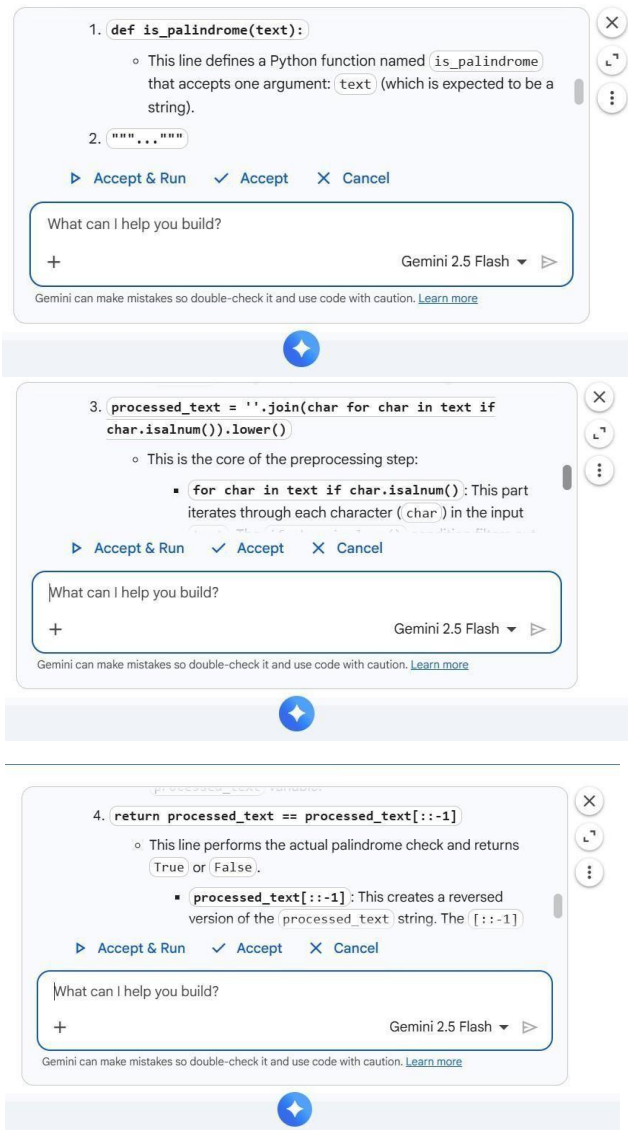
➤ Code snippet

➤ AI explanation

➤ Student comments on understanding

## OUTPUT:





### My own experience using both Gemini and GitHub Copilot:

While using Gemini in Google Colab, I found the explanations to be very clear and helpful in understanding the logic behind the code. Gemini was especially useful for learning and analyzing Python programs step by step. GitHub Copilot, was faster in generating code directly inside the editor and helped me complete tasks quickly. Copilot felt more suitable for continuous coding, while Gemini was better for conceptual clarity. Overall, using both tools together improved my coding efficiency and understanding.