

# AI ASSISTANT CODING ASSIGNMENT - 2

**NAME: Kushal Mandal**

**HT.NO: 2303A51621**

**BATCH: 22**

---

**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.**

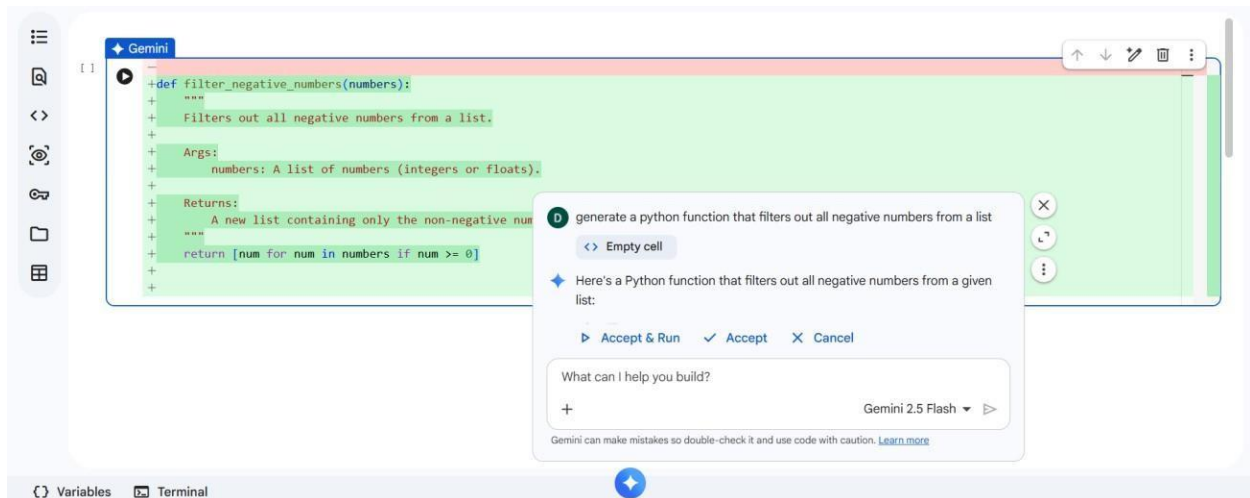
❖ **Expected Output:**

➤ **Before/after list**

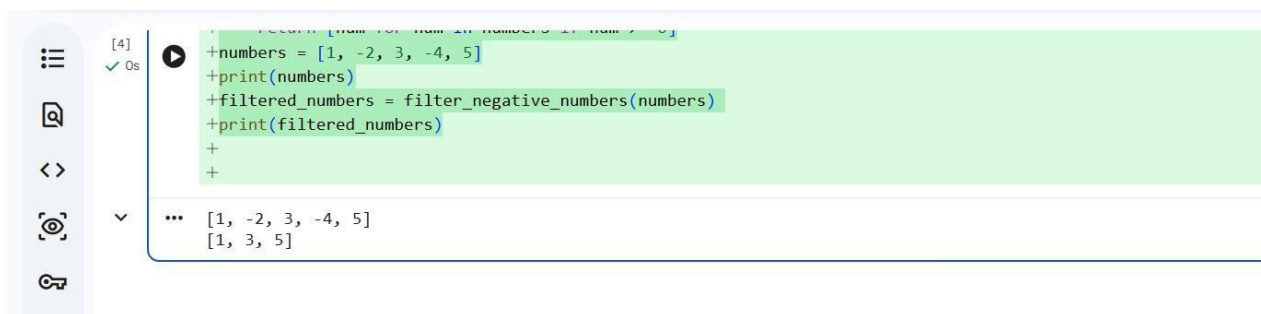
➤ **Screenshot of Colab execution**

---

## 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.

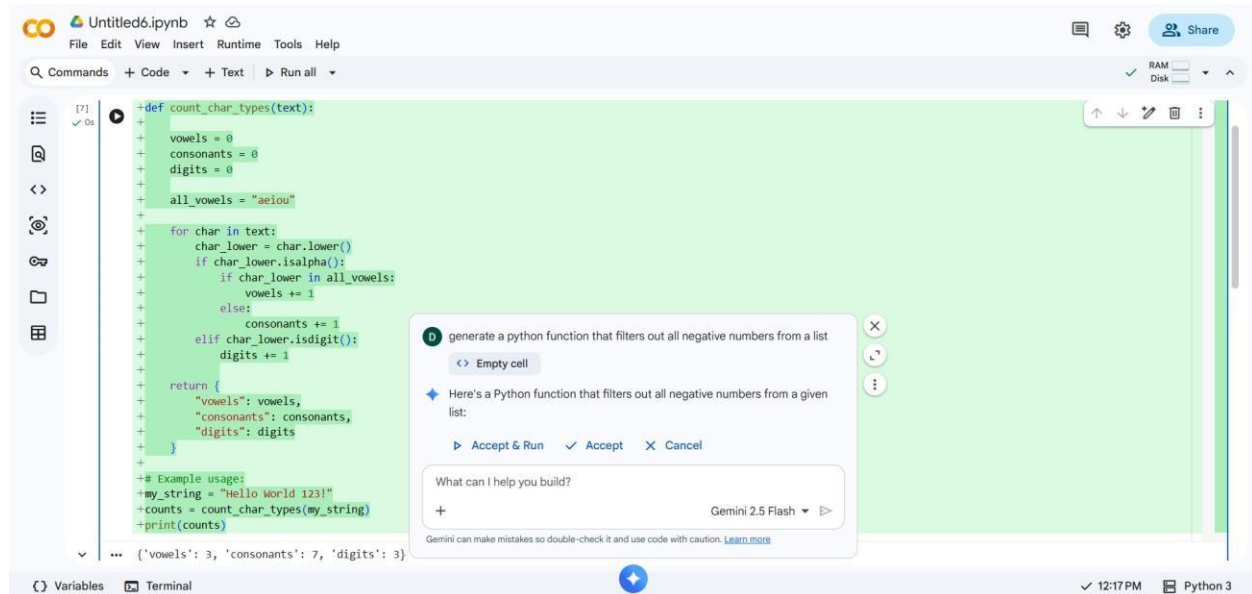
## ❖ Expected Output:

### ➤ Working function

### ➤ Sample inputs and outputs

---

## CODE :

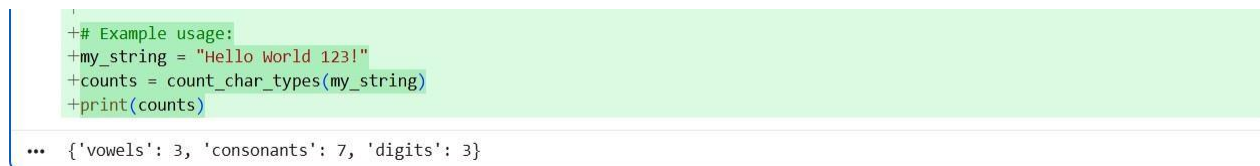


The screenshot shows a Jupyter Notebook titled 'Untitled6.ipynb'. The code in the cell defines a function `count_char_types(text)` that counts vowels, consonants, and digits in a string. The function initializes counters to 0 and a string of vowels 'aeiou'. It iterates through each character in the text, converting it to lowercase. If it's a vowel, it increments the vowel counter. If it's a digit, it increments the digit counter. Otherwise, it increments the consonant counter. The function returns a dictionary with the counts. Below the function definition, there is an example usage: `my_string = "Hello world 123!"`, `counts = count_char_types(my_string)`, and `print(counts)`. The output of the cell is `{'vowels': 3, 'consonants': 7, 'digits': 3}`. A Gemini AI chat window is overlaid on the right side of the notebook, showing a prompt to generate a Python function that filters out negative numbers from a list, and a response providing a function `filter_negative_numbers`.

```
[?] Untitled6.ipynb ☆ ☁
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text ▶ Run all
RAM Disk
[?] 0s
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
    }
    ...
    ... {'vowels': 3, 'consonants': 7, 'digits': 3}

What can I help you build?
+ Gemini 2.5 Flash ▶
Generate
Gemini can make mistakes so double-check it and use code with caution. Learn more
```

## OUTPUT:



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

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

## Task 3: Palindrome Check – Tool Comparison

### ❖ Scenario:

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

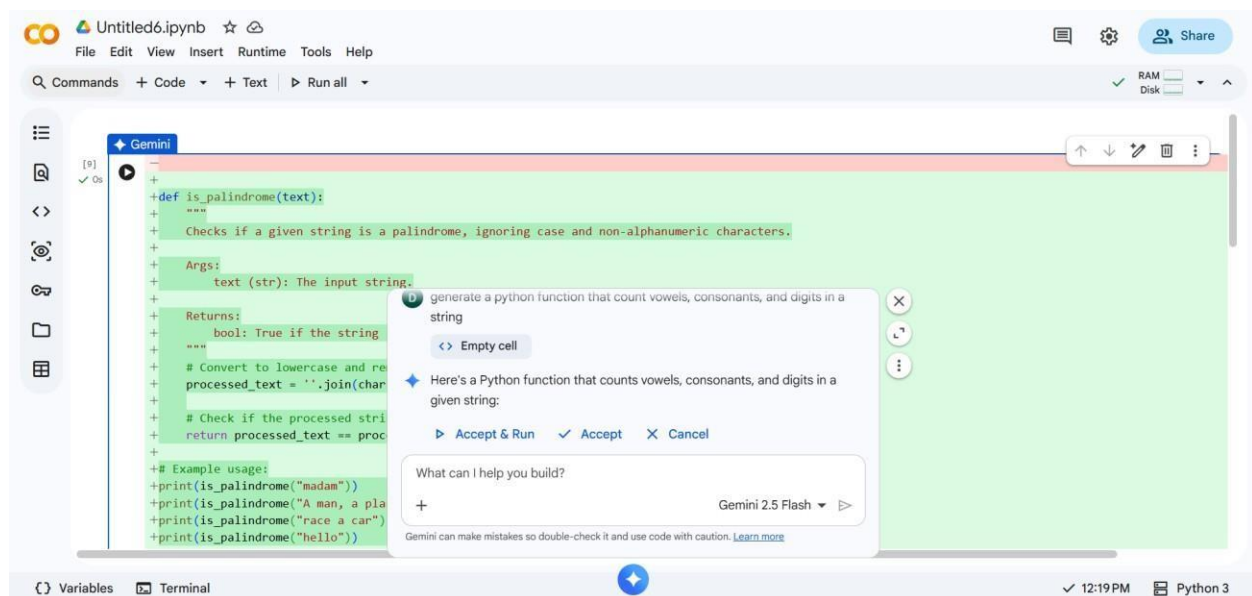
## ❖ Task:

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

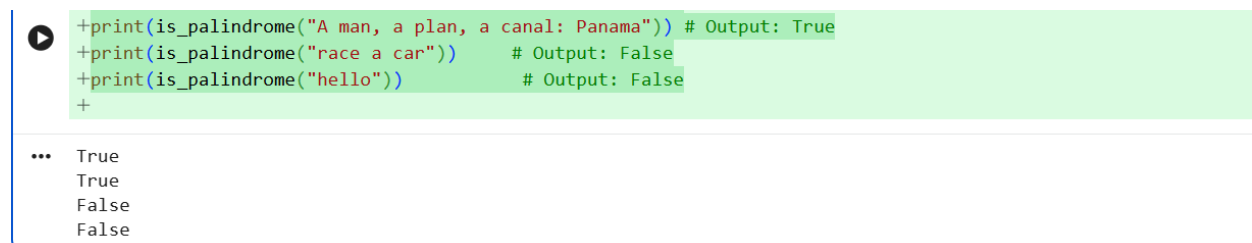
## ❖ Expected Output:

- Side-by-side code comparison
- Observations on clarity and structure

## CODE:



## OUTPUT:



## 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:

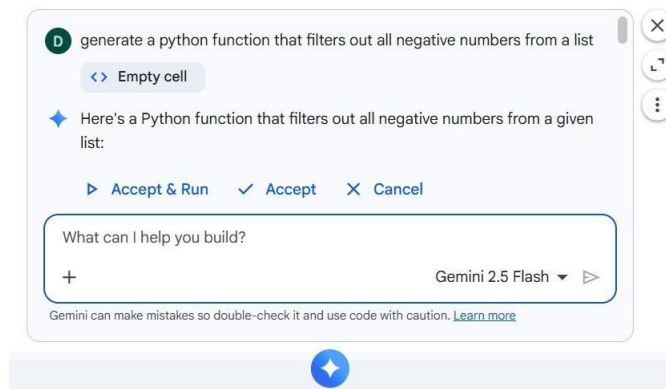
➤ Code snippet

➤ AI explanation

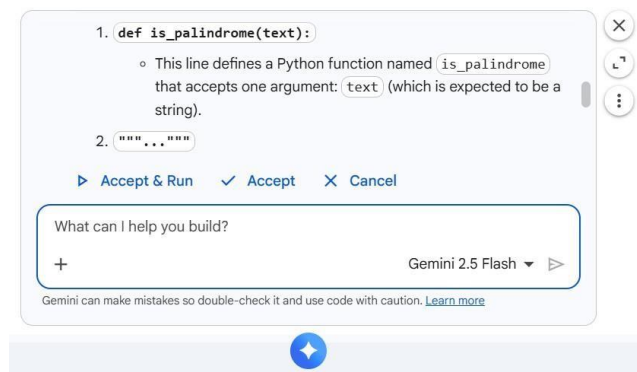
➤ Student comments on understanding

---

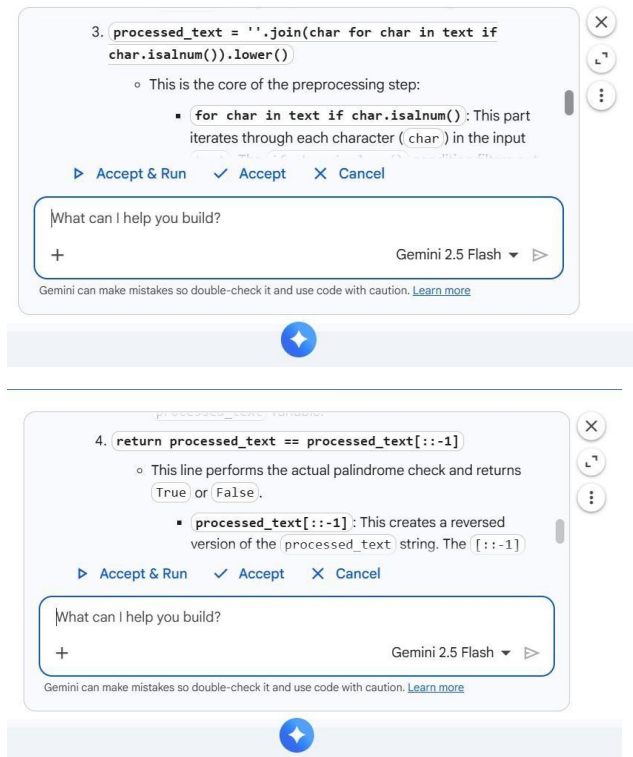
OUTPUT:



The screenshot shows a Gemini chat window. The prompt is "generate a python function that filters out all negative numbers from a list". The response is "Here's a Python function that filters out all negative numbers from a given list:". Below the response are three buttons: "Accept & Run", "Accept", and "Cancel". At the bottom of the chat window is a text input field with the placeholder "What can I help you build?" and a "Gemini 2.5 Flash" dropdown menu. A small disclaimer at the bottom reads "Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)".



The screenshot shows a Gemini chat window. The prompt is "1. def is\_palindrome(text):". The response is "This line defines a Python function named is\_palindrome that accepts one argument: text (which is expected to be a string).". Below the response are three buttons: "Accept & Run", "Accept", and "Cancel". At the bottom of the chat window is a text input field with the placeholder "What can I help you build?" and a "Gemini 2.5 Flash" dropdown menu. A small disclaimer at the bottom reads "Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)".



### 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.