Course Title: AI Assisted Coding

Course Code: 23CS002PC304

Faculty Name: Dr. R. Prashant Kumar

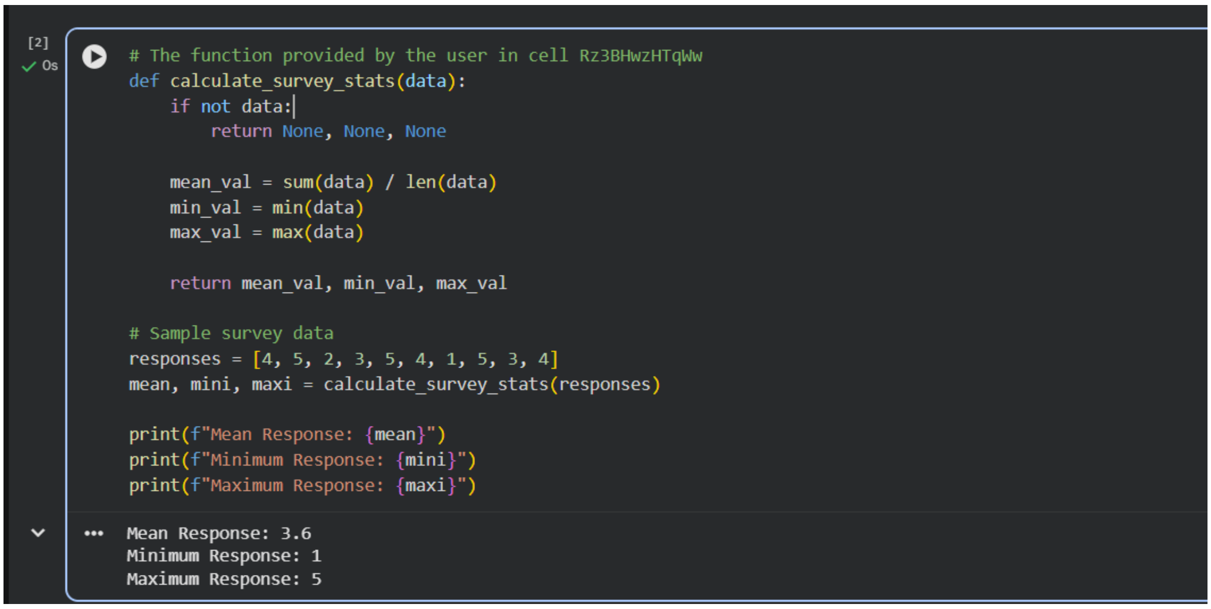
Name: D. Varshitha

HT no: 2303A52268- Batch(36)

Question: Lab 2: Exploring Additional AI Coding Tools beyond Copilot – Gemini (Colab) and Cursor AI

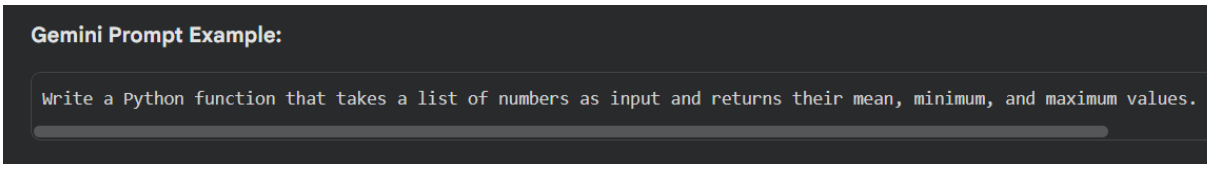
Task 1: Statistical Summary for Survey Data

Python Function: The Python function you have in cell Rz3BHwzHTqWw (calculate_survey_stats) is an excellent example of what Gemini would likely generate to address this prompt. The code is structured correctly, calculates the required statistics, and handles the edge case of an empty list.



```
# The function provided by the user in cell Rz3BHwzHTqWw
def calculate_survey_stats(data):
    if not data:
        return None, None, None

    mean_val = sum(data) / len(data)
    min_val = min(data)
    max_val = max(data)

    return mean_val, min_val, max_val

# Sample survey data
responses = [4, 5, 2, 3, 5, 4, 1, 5, 3, 4]
mean, mini, maxi = calculate_survey_stats(responses)

print(f"Mean Response: {mean}")
print(f"Minimum Response: {mini}")
print(f"Maximum Response: {maxi}")
```

```
Mean Response: 3.6
Minimum Response: 1
Maximum Response: 5
```

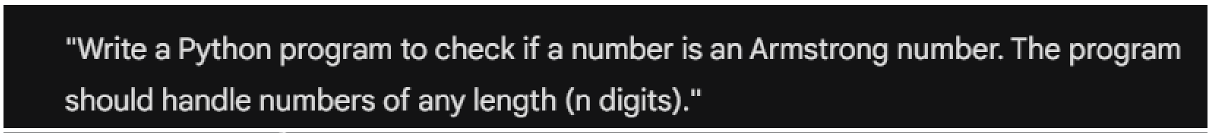


Gemini Prompt Example:

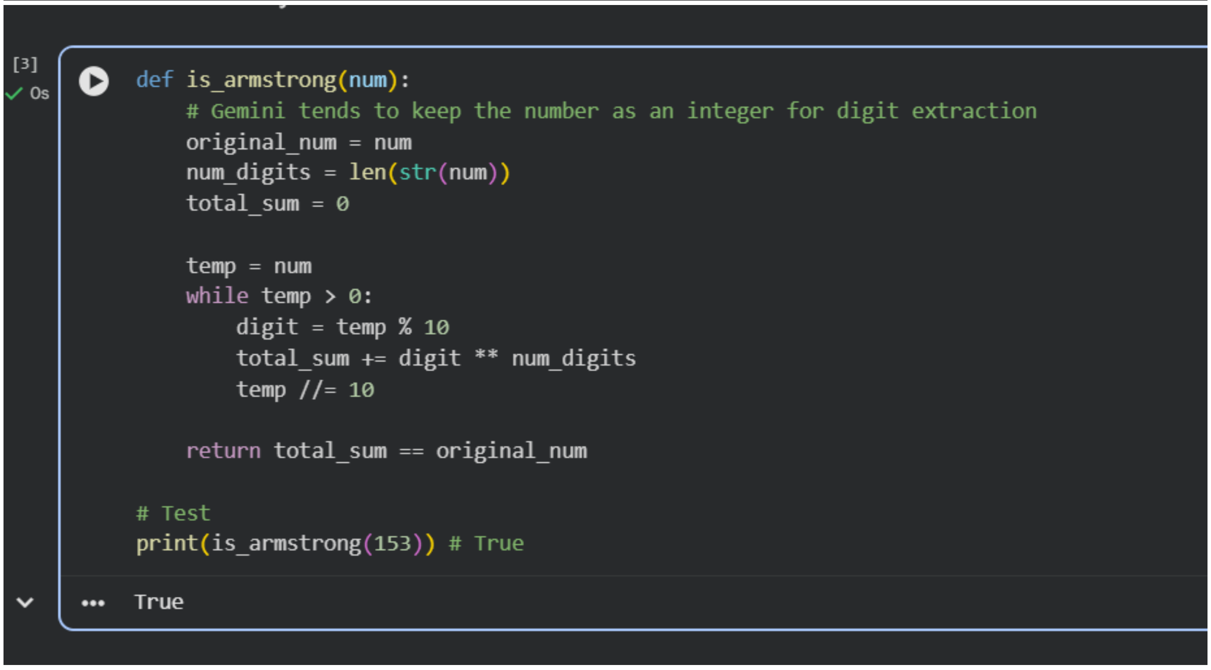
```
Write a Python function that takes a list of numbers as input and returns their mean, minimum, and maximum values.
```

Task 2: Armstrong Number – AI Comparison

Feature	**Google Gemini (Colab)**	**GitHub Copilot**
Logic Style	**Educational/Mathematical:** Often uses a while loop with % 10 and // 10 to extract digits.	**Pythonic/Concise:** Often uses string conversion str(n) and list comprehensions for speed.
Code Clarity	Very High. Includes descriptive variable names and full docstrings.	High. Focuses on being "ready-to-use" with minimal boilerplate.
Explanation	Provides a step-by-step tutorial on *why* the math works.	Provides a brief summary of *what* the code does.
Best For	Students and analysts who want to understand the algorithm.	Developers who need a quick, efficient snippet to paste into a project.



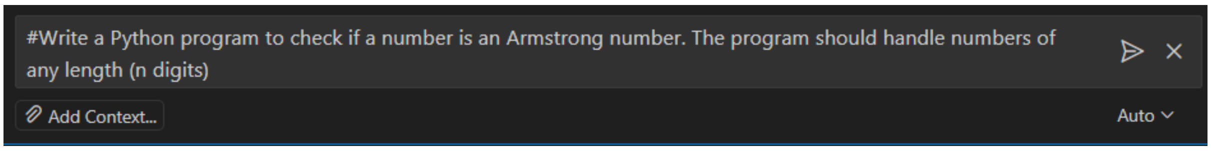
"Write a Python program to check if a number is an Armstrong number. The program should handle numbers of any length (n digits)."



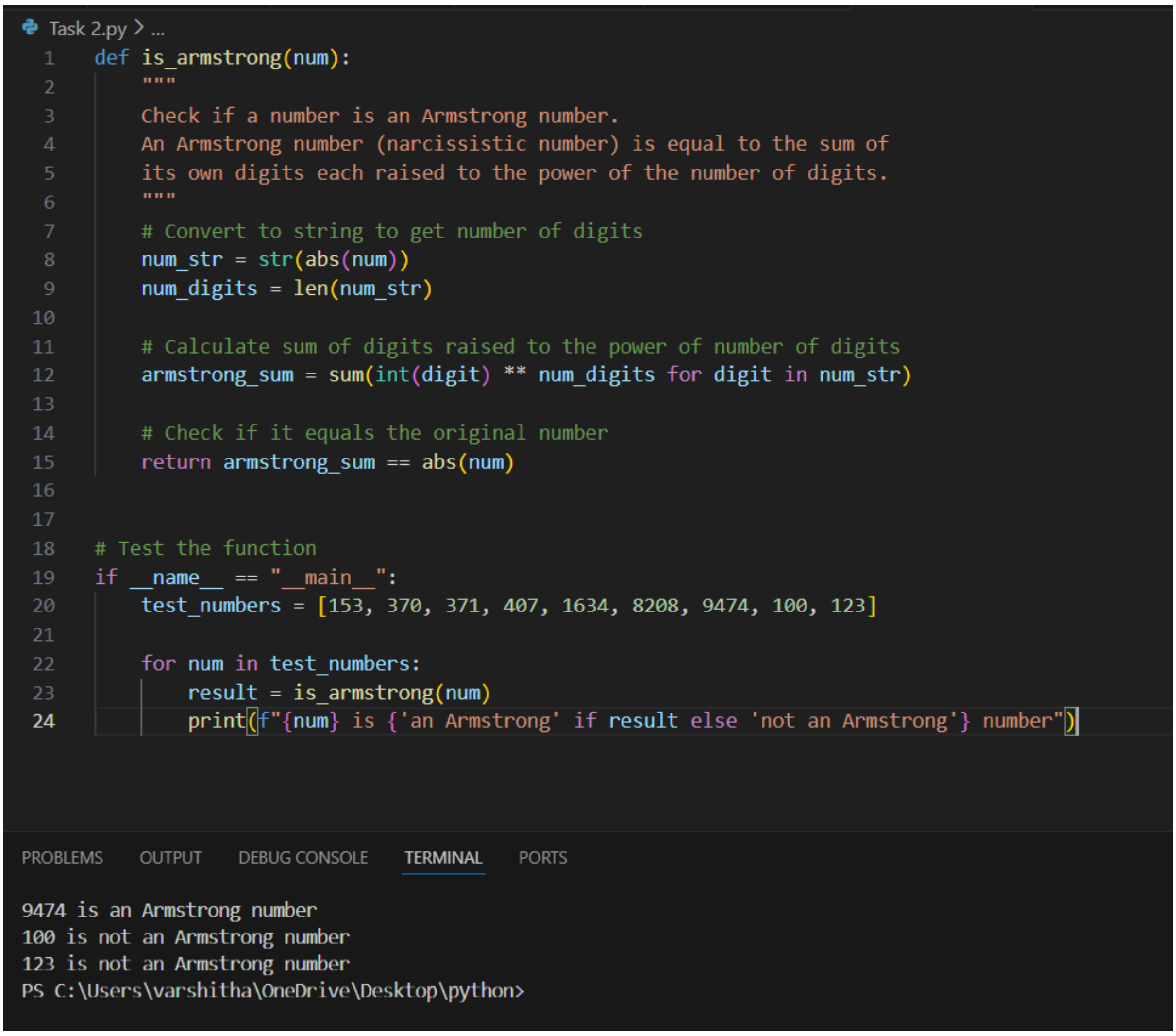
[3]
0s

```python
def is_armstrong(num):
    # Gemini tends to keep the number as an integer for digit extraction
    original_num = num
    num_digits = len(str(num))
    total_sum = 0

    temp = num
    while temp > 0:
        digit = temp % 10
        total_sum += digit ** num_digits
        temp //= 10

    return total_sum == original_num

# Test
print(is_armstrong(153)) # True
```

```
True
```



#Write a Python program to check if a number is an Armstrong number. The program should handle numbers of any length (n digits)



```python
def is_armstrong(num):
    """
    Check if a number is an Armstrong number.
    An Armstrong number (narcissistic number) is equal to the sum of
    its own digits each raised to the power of the number of digits.
    """
    # Convert to string to get number of digits
    num_str = str(abs(num))
    num_digits = len(num_str)

    # Calculate sum of digits raised to the power of number of digits
    armstrong_sum = sum(int(digit) ** num_digits for digit in num_str)

    # Check if it equals the original number
    return armstrong_sum == abs(num)


# Test the function
if __name__ == "__main__":
    test_numbers = [153, 370, 371, 407, 1634, 8208, 9474, 100, 123]

    for num in test_numbers:
        result = is_armstrong(num)
        print(f"{num} is {'an Armstrong' if result else 'not an Armstrong'} number")
```

```
9474 is an Armstrong number
100 is not an Armstrong number
123 is not an Armstrong number
PS C:\Users\varshitha\OneDrive\Desktop\python>
```