Name: B.Vishnu vardhan

Roll no: 2403A510F2

Batch no: 06

Subject: AI assisted coding

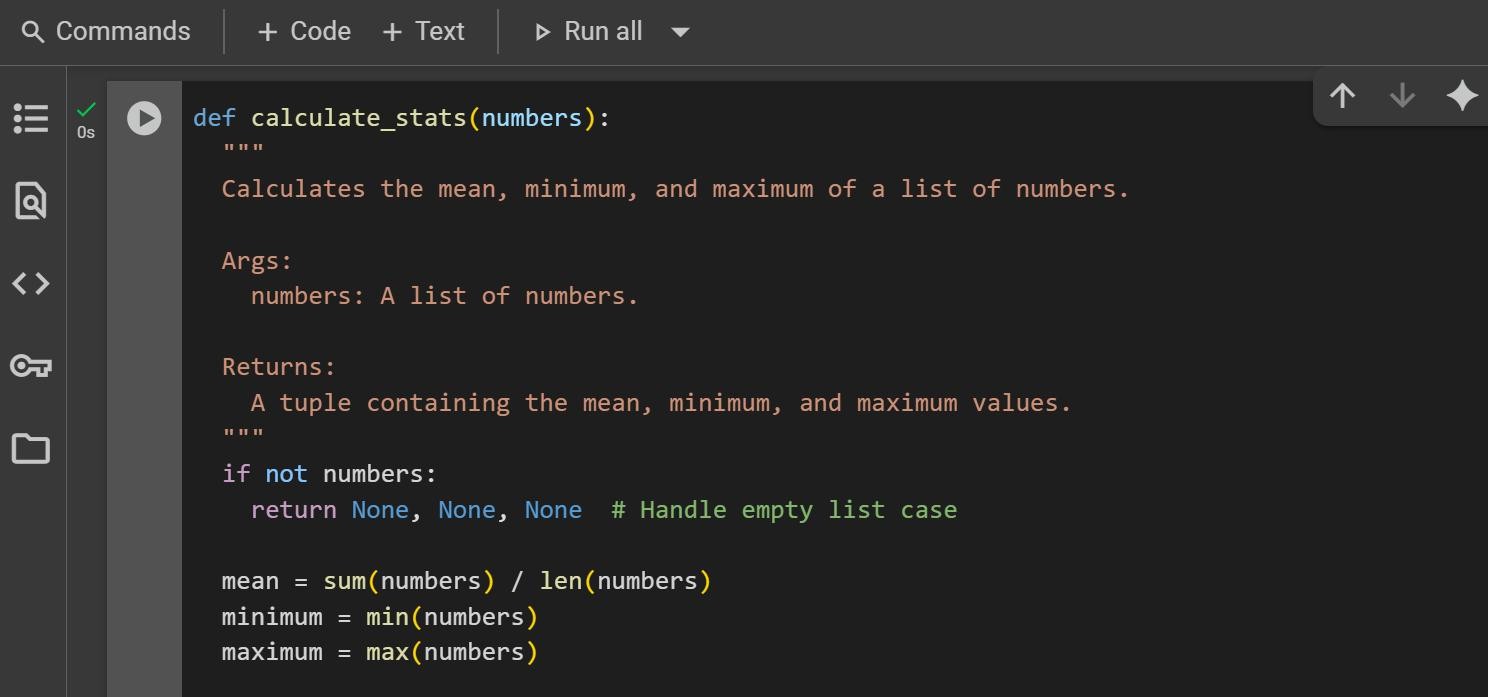


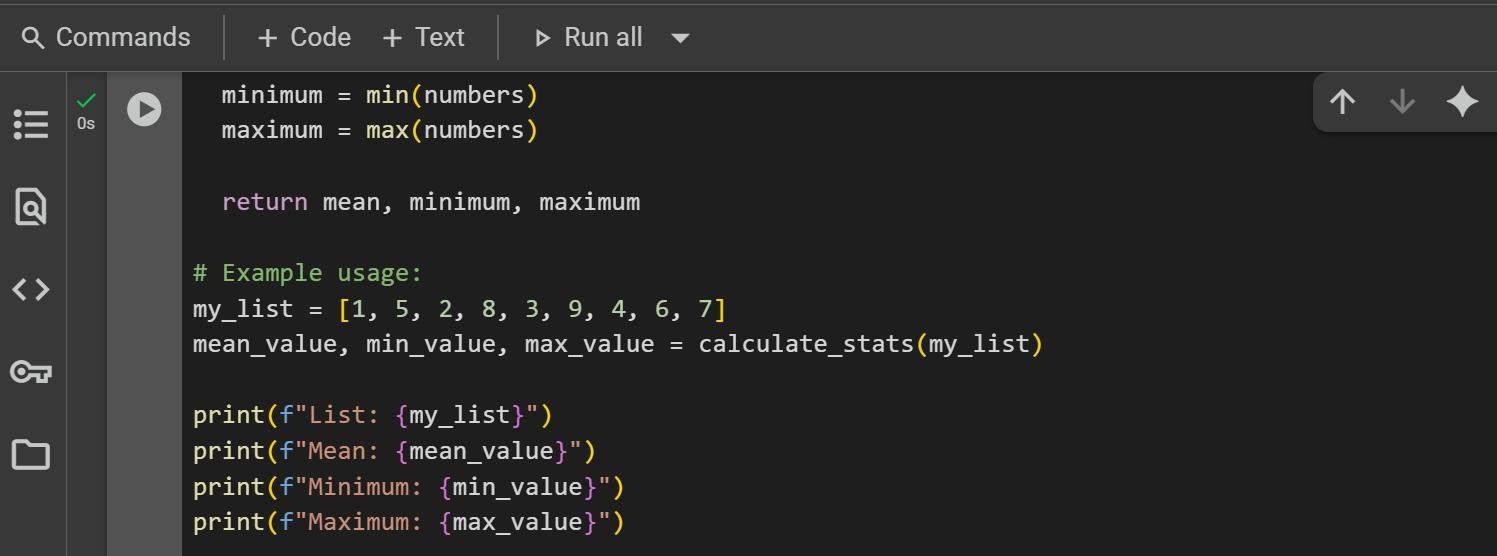
* Use Google Gemini in Colab to write a Python function that reads

a list of numbers and calculates the mean, minimum, and maximum values Expected Output

* Functional code with correct output and screenshot









List: [1, 5, 2, 8, 3, 9, 4, 6, 7]

Mean: 5.0

Minimum: 1

Maximum: 9

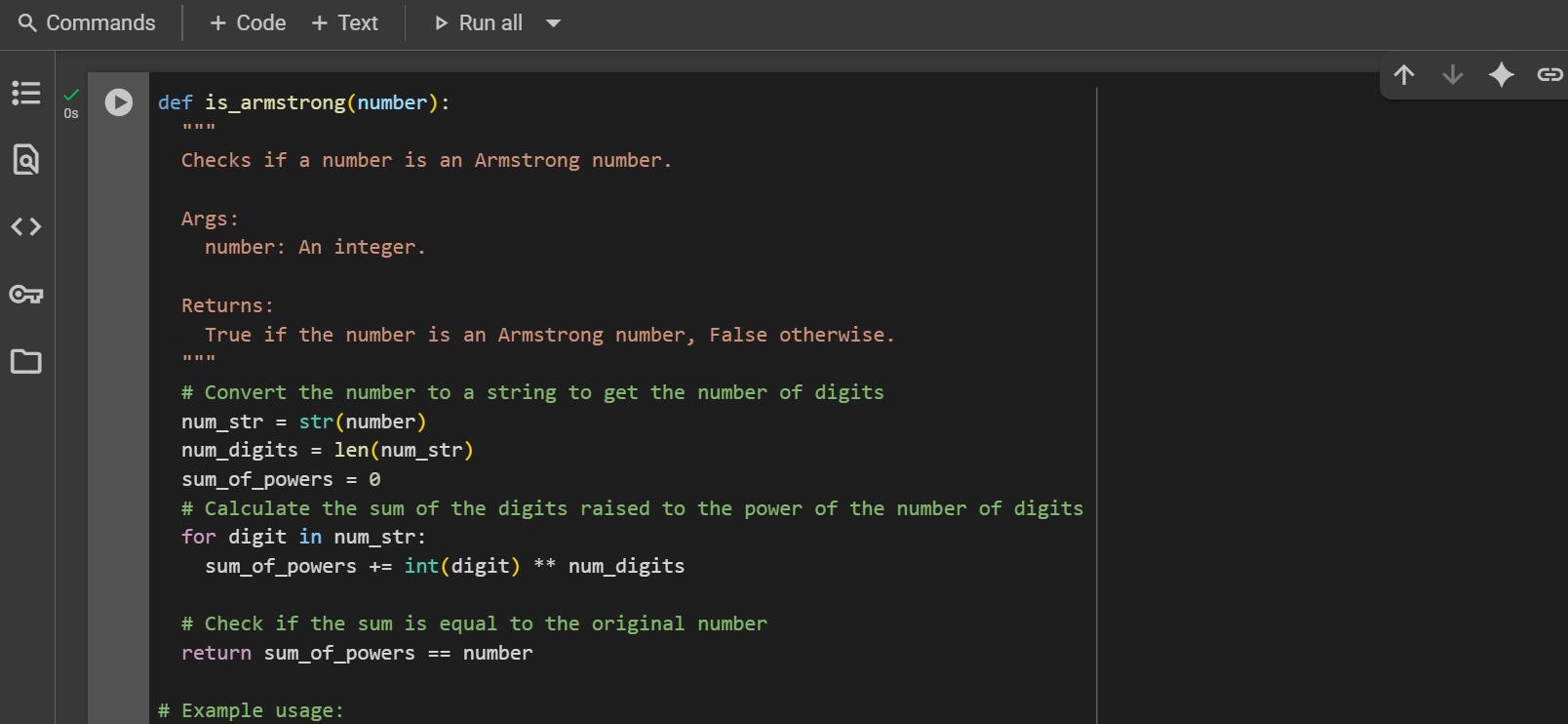


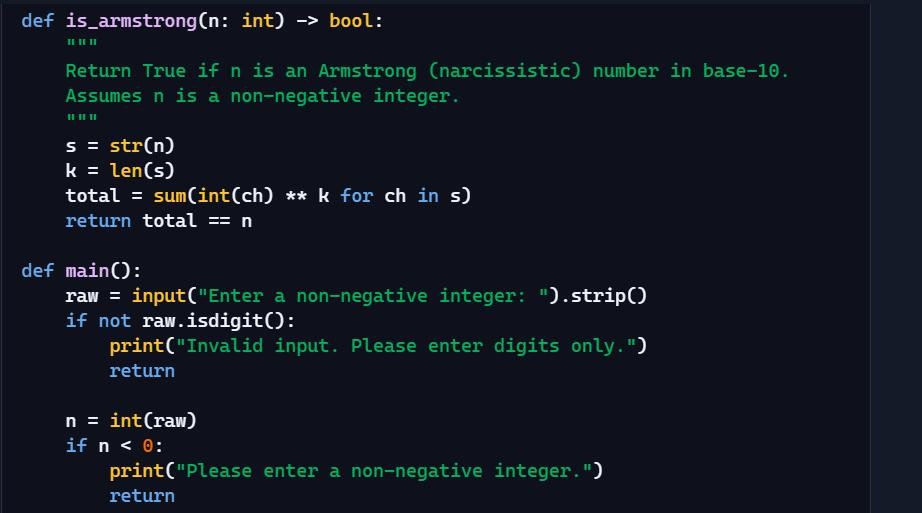
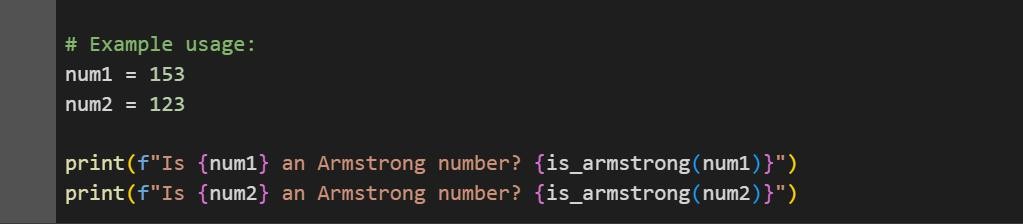
* Compare Gemini and Copilot outputs for a Python function that checks whether a number is an Armstrong number. Document the steps, prompts, and outputs.

Expected Output 2

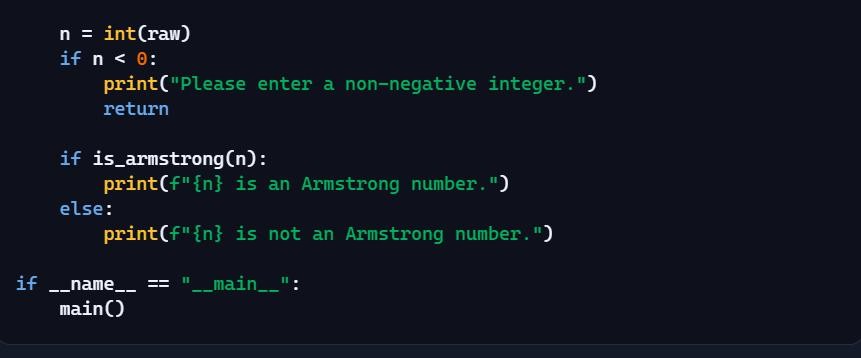
* Side-by-side comparison table with observations and screenshots







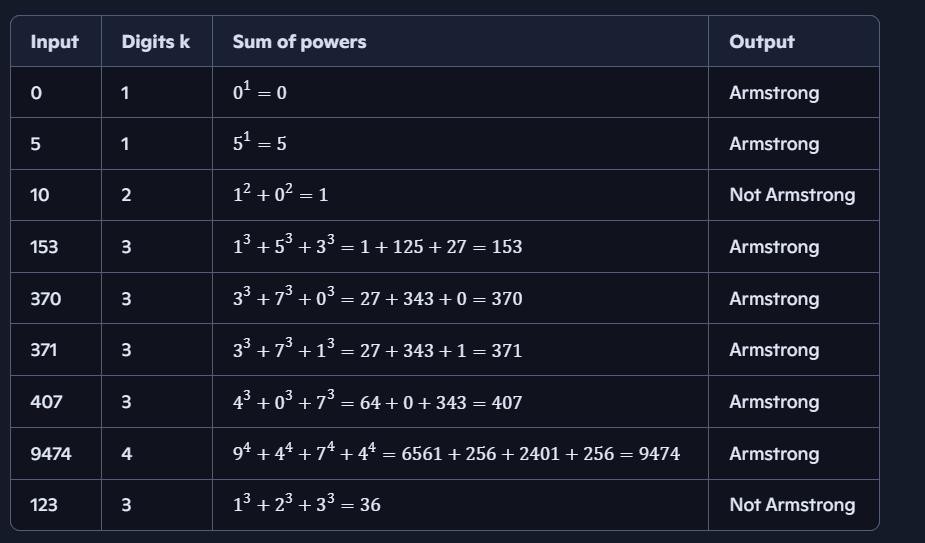






Is 153 an Armstrong number? True Is 123 an Armstrong number? False





As a part of the task we have to compare the differences in both(Gemini code and Microsoft copilot code) Codes. Let’s breakout the task.

**1. Function Signature & Documentation**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Gemini Code** | **Microsoft Copilot Code** |
| **Function name & parameters** | def is\_armstrong(number) — no type hints. | def is\_armstrong(n: int) -> bool — has type hints for input and return type. |
| **Docstring style** | Explains purpose, arguments, and return type in a simple comment block. | Explains purpose with clear assumptions (n is non- negative) in concise form. |

**Difference:**

Copilot code uses **Python type hints** and a more concise docstring; Gemini’s is more verbose but beginner-friendly.

1. Core Logic Implementation

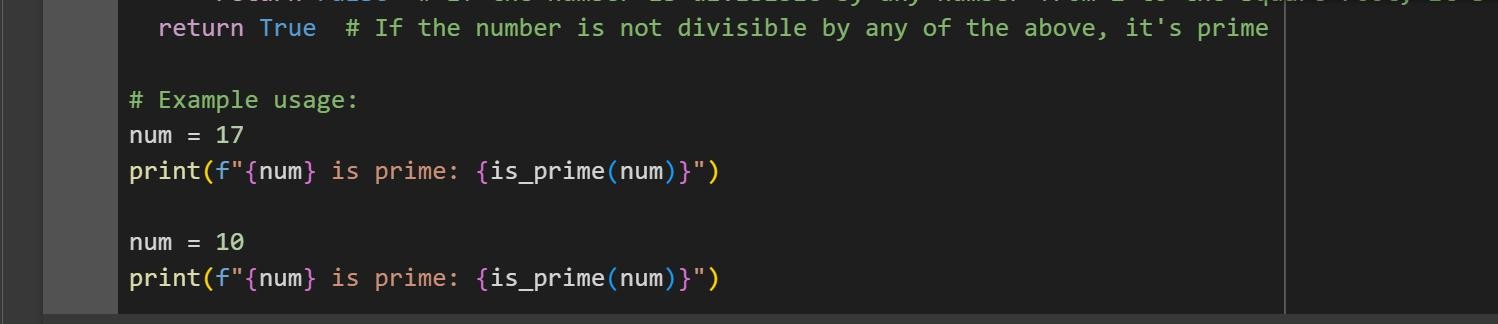
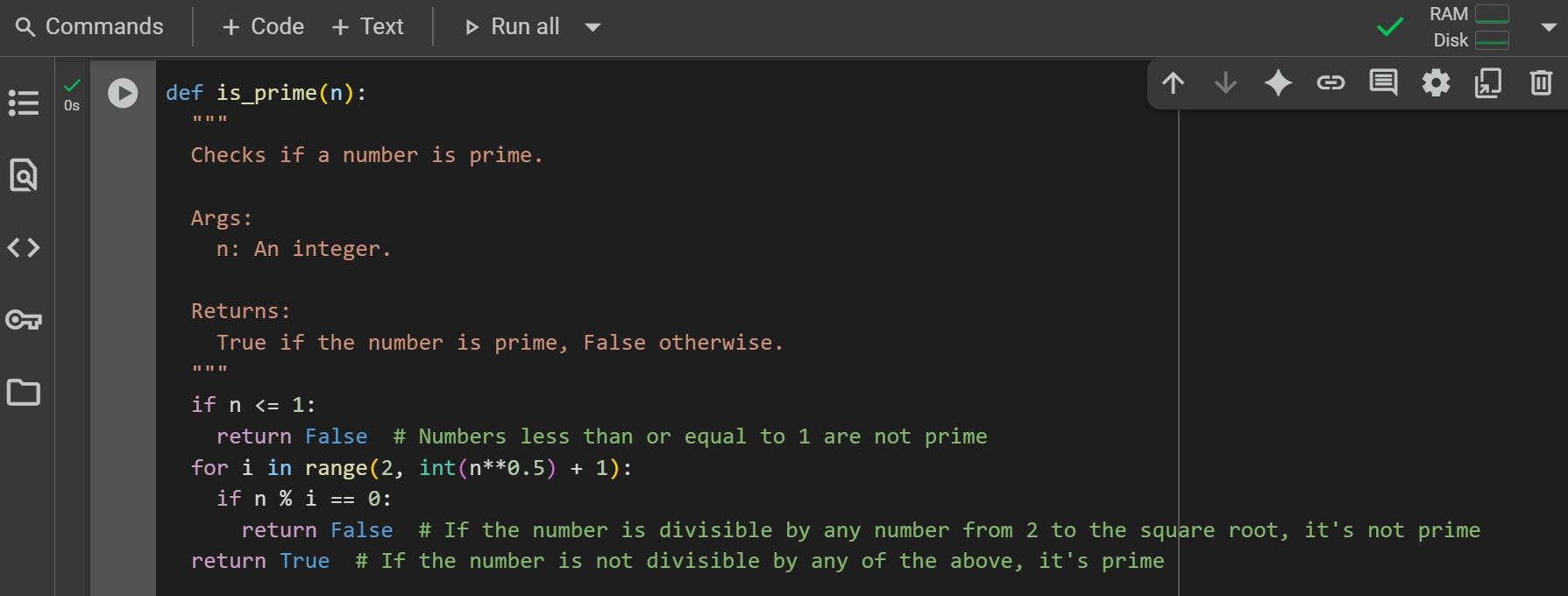
|  |  |  |
| --- | --- | --- |
| **Step** | **Gemini Code** | **Microsoft Copilot Code** |
| Convert to string | num\_str = str(number) | s = str(n) |
| Find number of digits | num\_digits = len(num\_str) | k = len(s) |
| Sum of digits to the power of digit count | Uses a for loop: |  |



* + Ask Gemini to explain a Python function (e.g., is\_prime(n) or is\_palindrome(s)) line by line.
  + Choose either a prime-checking or palindrome-checking function and document the explanation provided by Gemini.

Expected Output

* + Detailed explanation with the code snippet and Gemini’s response



17 is prime: True 10 is prime: False

Explanation:

This Python function, is\_ prime(n), checks if a given integer n is a prime number. Here's how it works:

* **if n <= 1: return False**: It first handles the base cases. Numbers less than or equal to 1 are not considered prime, so it immediately returns False.
* **`for I in range (2, int (n**0.5) + 1): \*\*: This line starts a loop. It iterates through numbers starting from 2 up to the square root of n (including the square root). We only need to check for divisors up to the square root because if a number` has a divisor greater than its square root, it must also have a divisor smaller than its square root.
* **if n % I == 0: return False**: Inside the loop, for each number I , it checks if n is divisible by I using the modulo operator (%). If the remainder of the division is 0, it means n is divisible by I , and therefore, n is not a prime number. In this case, the function immediately returns False.
* **return True**: If the loop finishes without finding any divisors (meaning the if n % i == 0: condition was never met), it means n is not divisible by any number from 2 up to its square root. This indicates that n is a prime number, and the function returns True.

In simple terms, the function tries to divide the input number by all possible integers starting from 2 up to its square root. If it finds any number that divides it evenly, it's not prime. If it goes through all these checks without finding a divisor, then the number is prime.

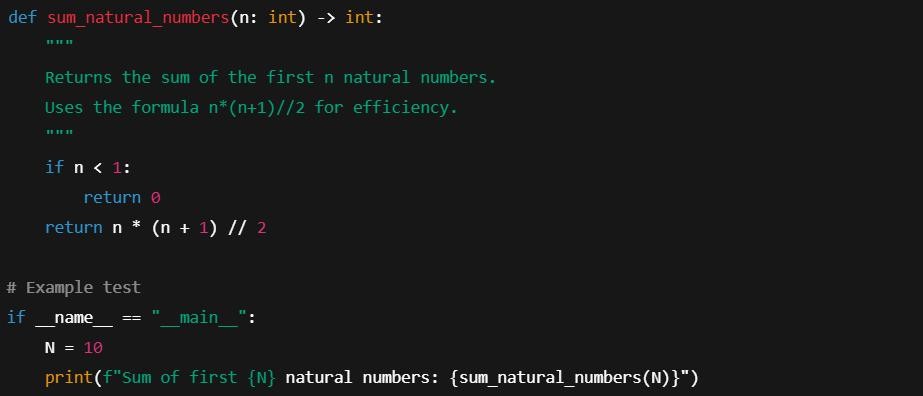
* + Install and configure Cursor AI. Use it to generate a Python function (e.g., sum of the first N natural numbers) and test its output.
  + Optionally, compare Cursor AI’s generated code with Gemini’s output.



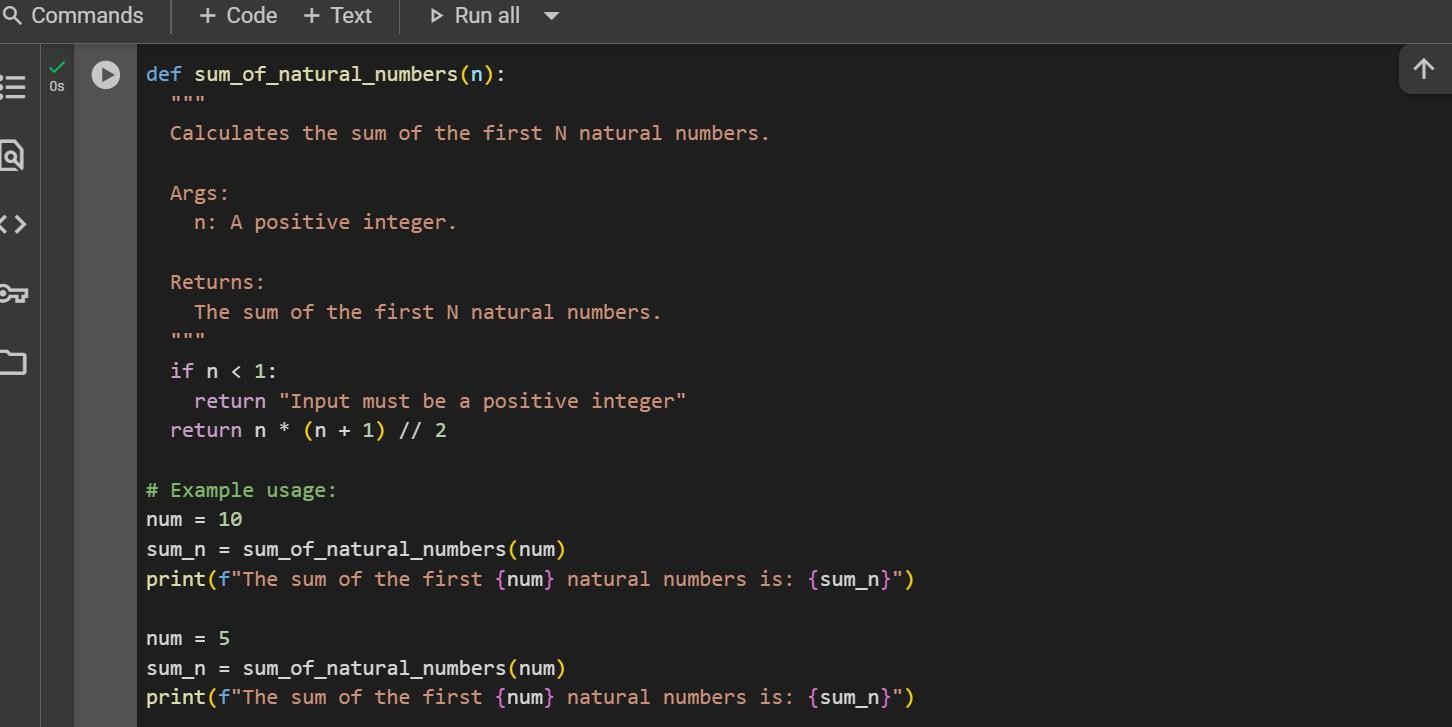
Expected Output 4

* + Screenshots of Cursor AI setup, prompts used, and generated code with output.











Sum of first 10 natural numbers: 55 Gemini AI Output:

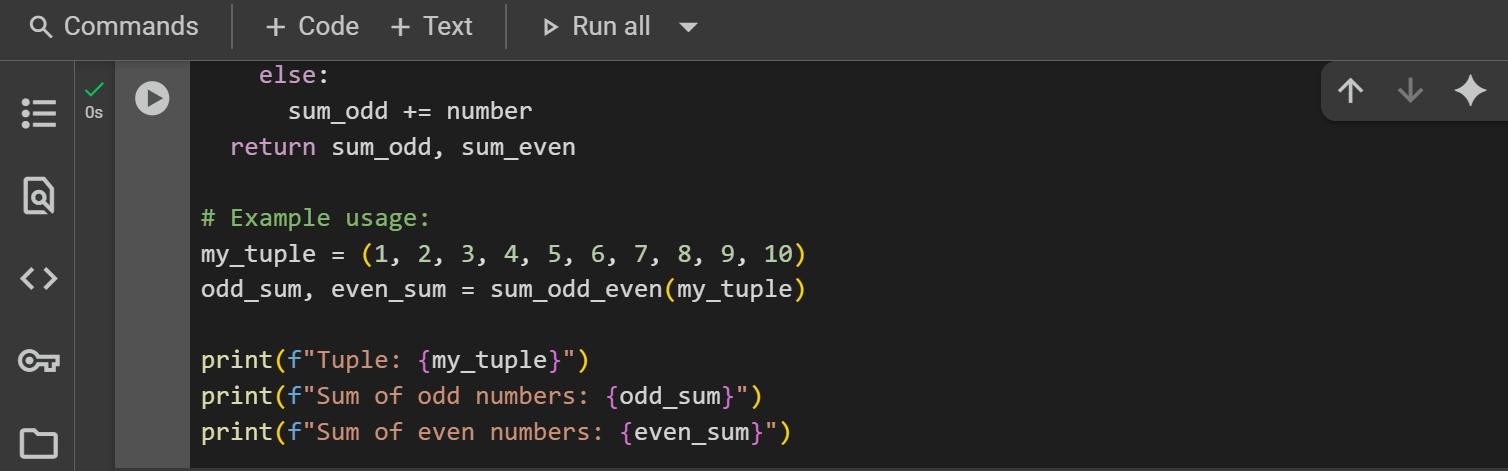
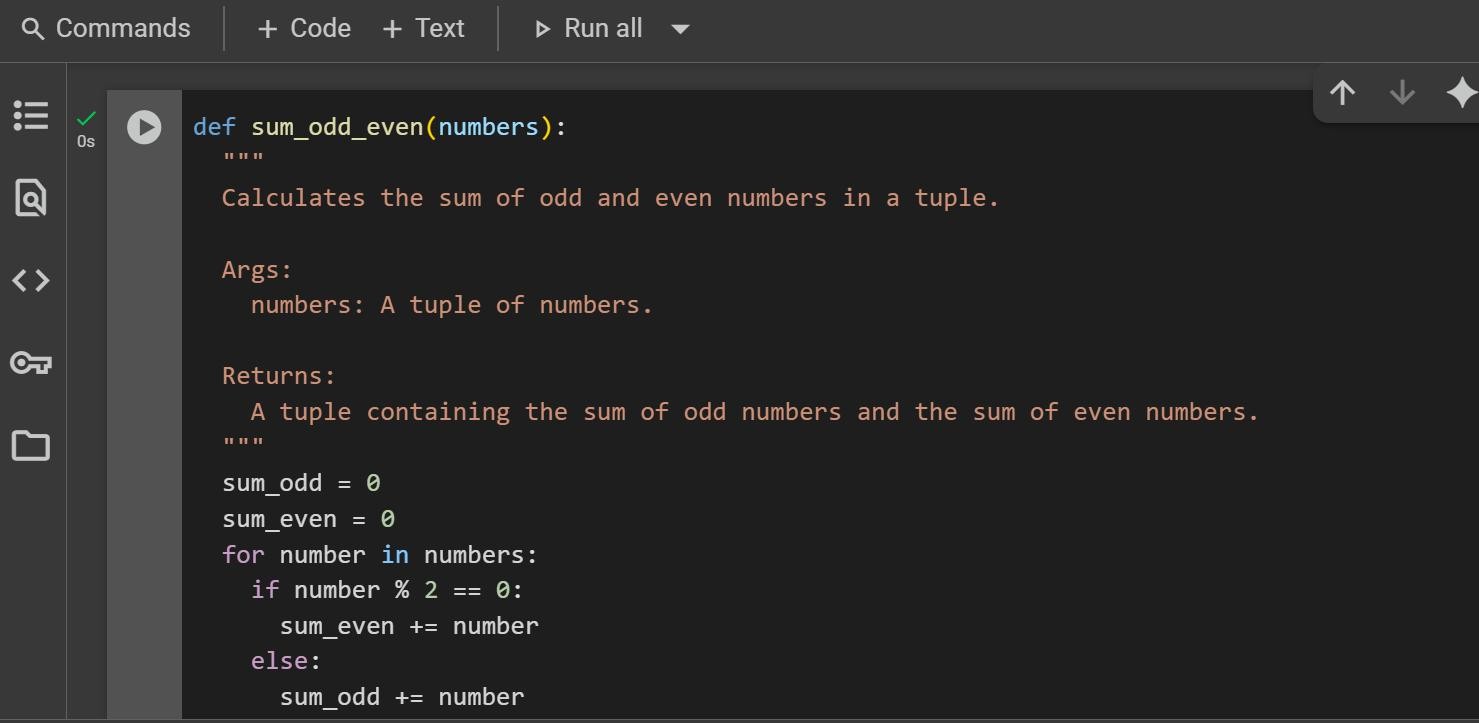
The sum of the first 10 natural numbers is: 55 The sum of the first 5 natural numbers is: 15

Comparision Table:

|  |  |  |
| --- | --- | --- |
| **Feature** | **Cursor AI** | **Gemini** |
| Approach | Formula-based (n\*(n+1)//2) | Loop-based sum |
| Efficiency | O(1) time | O(n) time |
| Readability | Short, efficient | Easy for beginners |
| Math Concept | Requires formula knowledge | No formula needed |



* + Students need to write a Python program to calculate the sum of odd numbers and even numbers in a given tuple.
  + Refactor the code to improve logic and readability. Expected Output 5
  + Student-written refactored code with explanations and output screenshots



Tuple: (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

Sum of odd numbers: 25 Sum of even numbers: 30