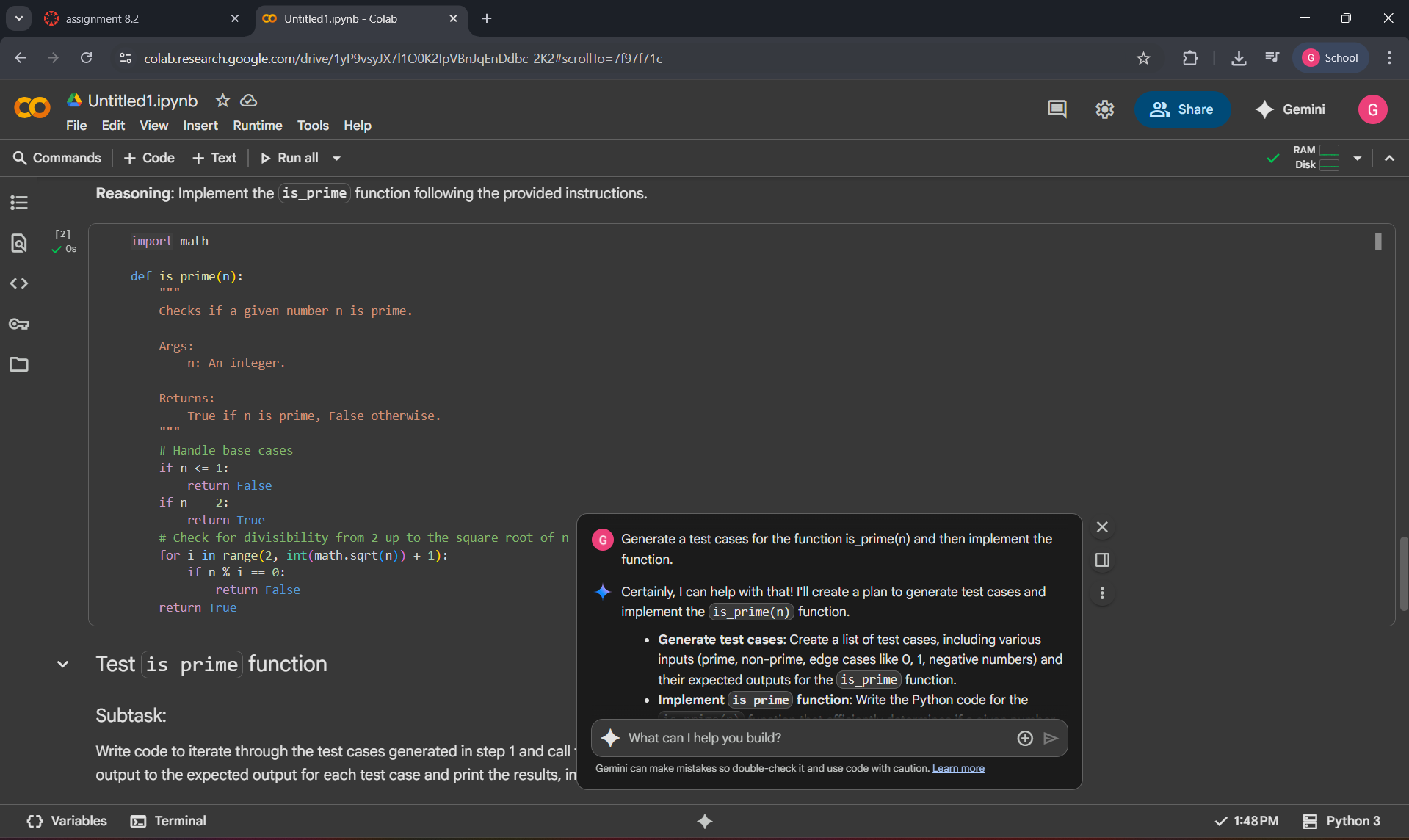
AI-ASSISTED CODING

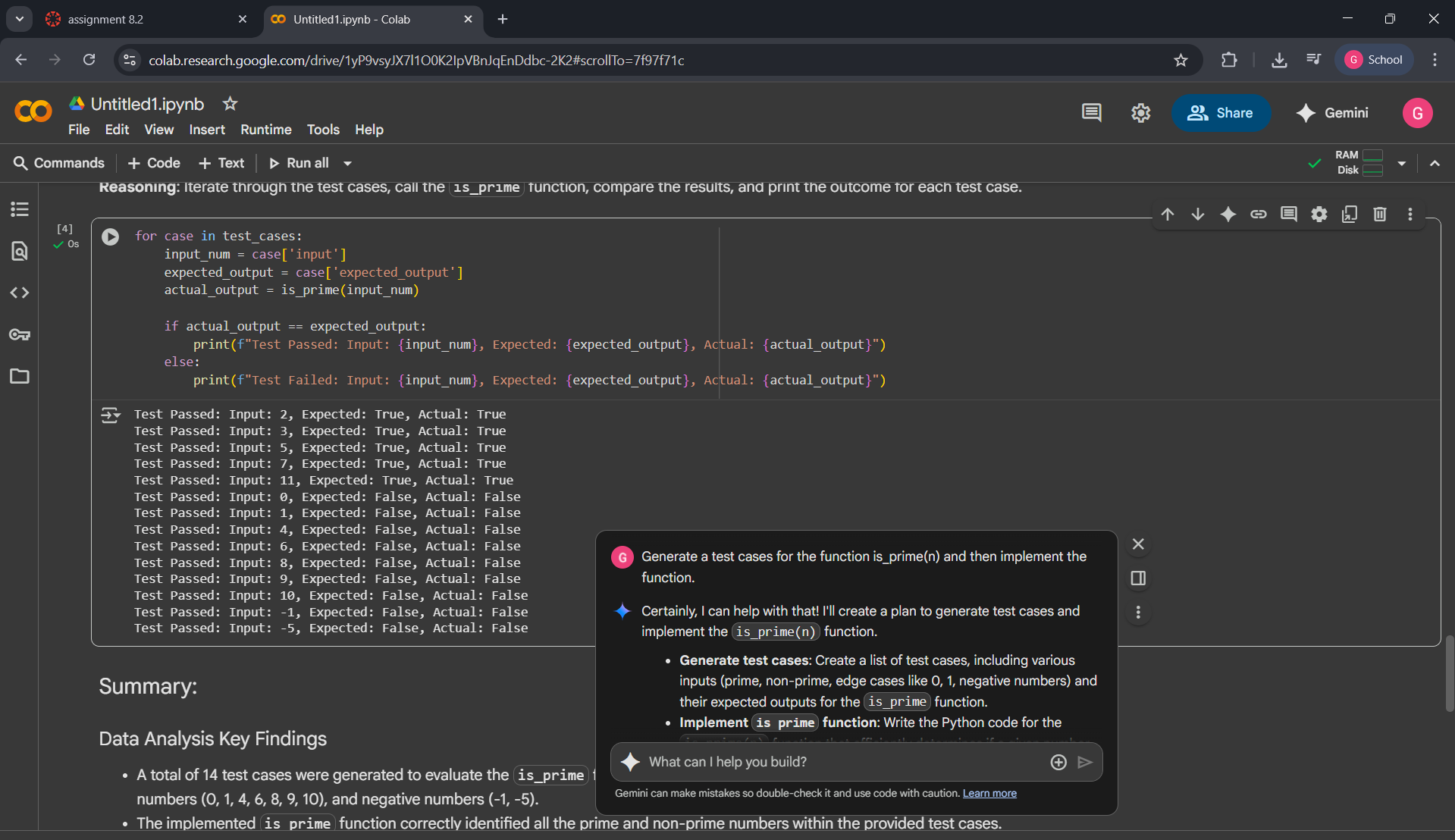
NAME : GUJJA RAJU

Ht. no.:2403A52018

ASSIGNMENT 7.2

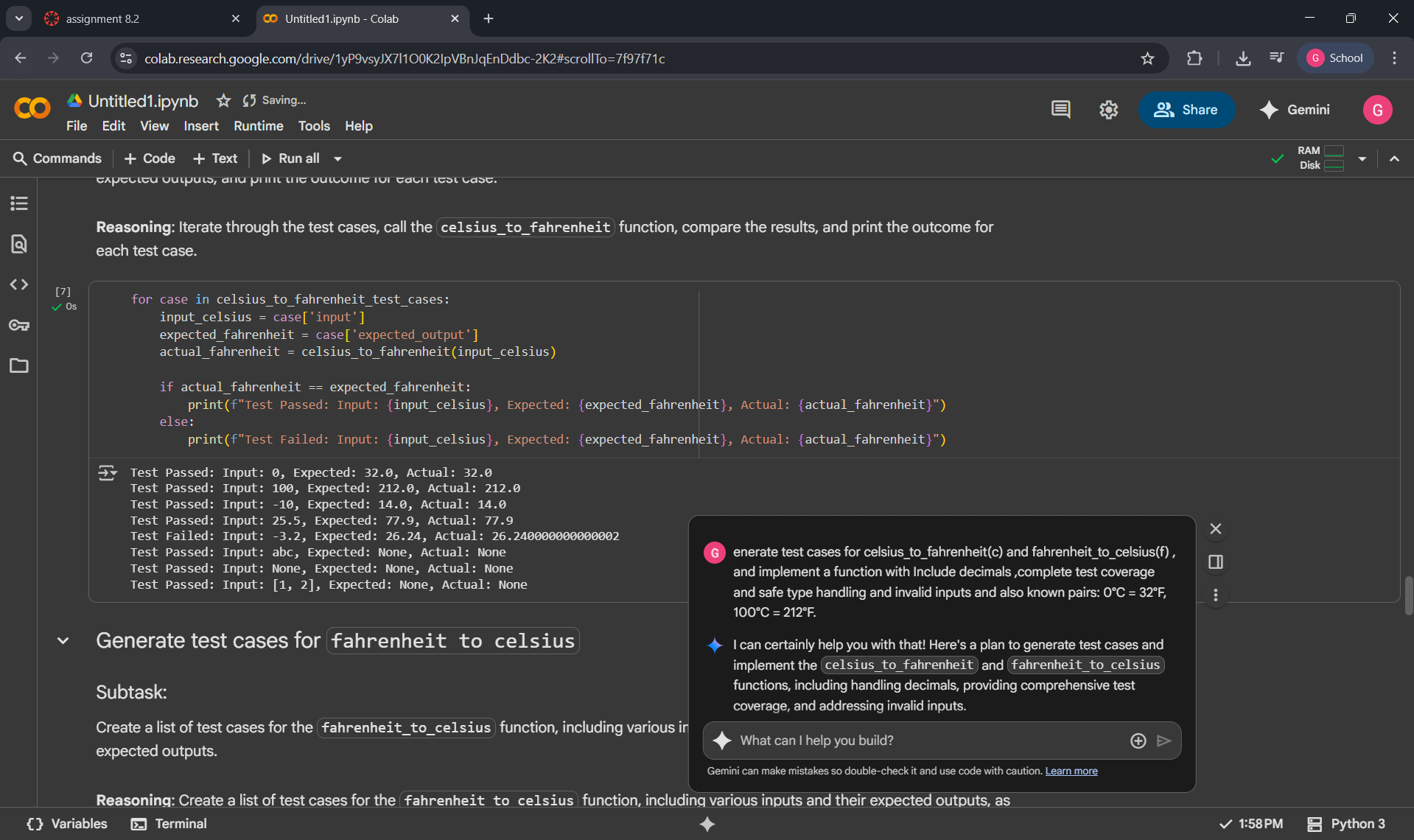
TASK 1:

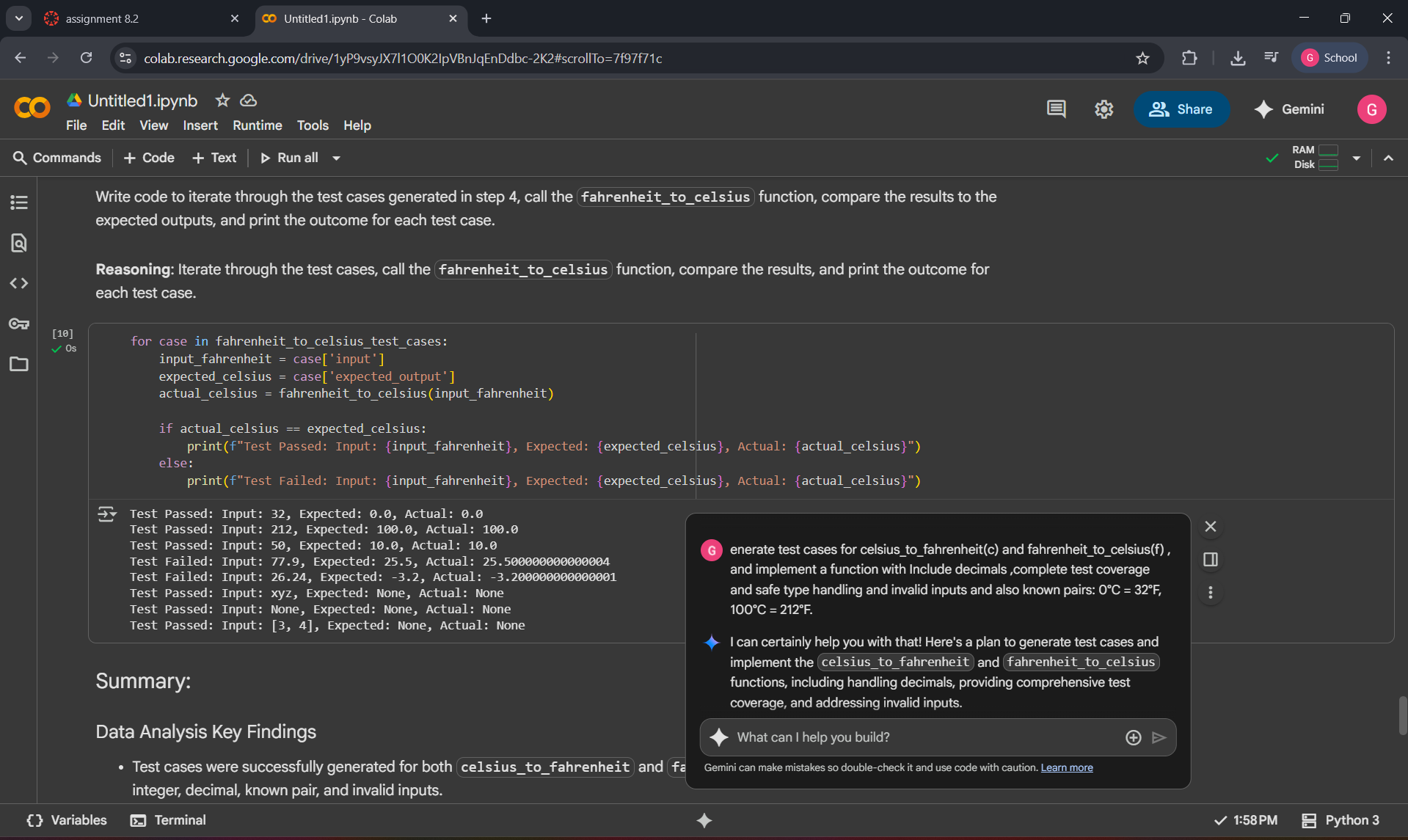




* EXPLANATION:
* The implemented is\_prime function correctly identified all the prime and non-prime numbers within the provided test cases.
* The function also correctly handled edge cases, returning False for 0, 1, and negative inputs.
* All 14 test cases passed, indicating that the is\_prime function's output matched the expected output for each input.

TASK 2:

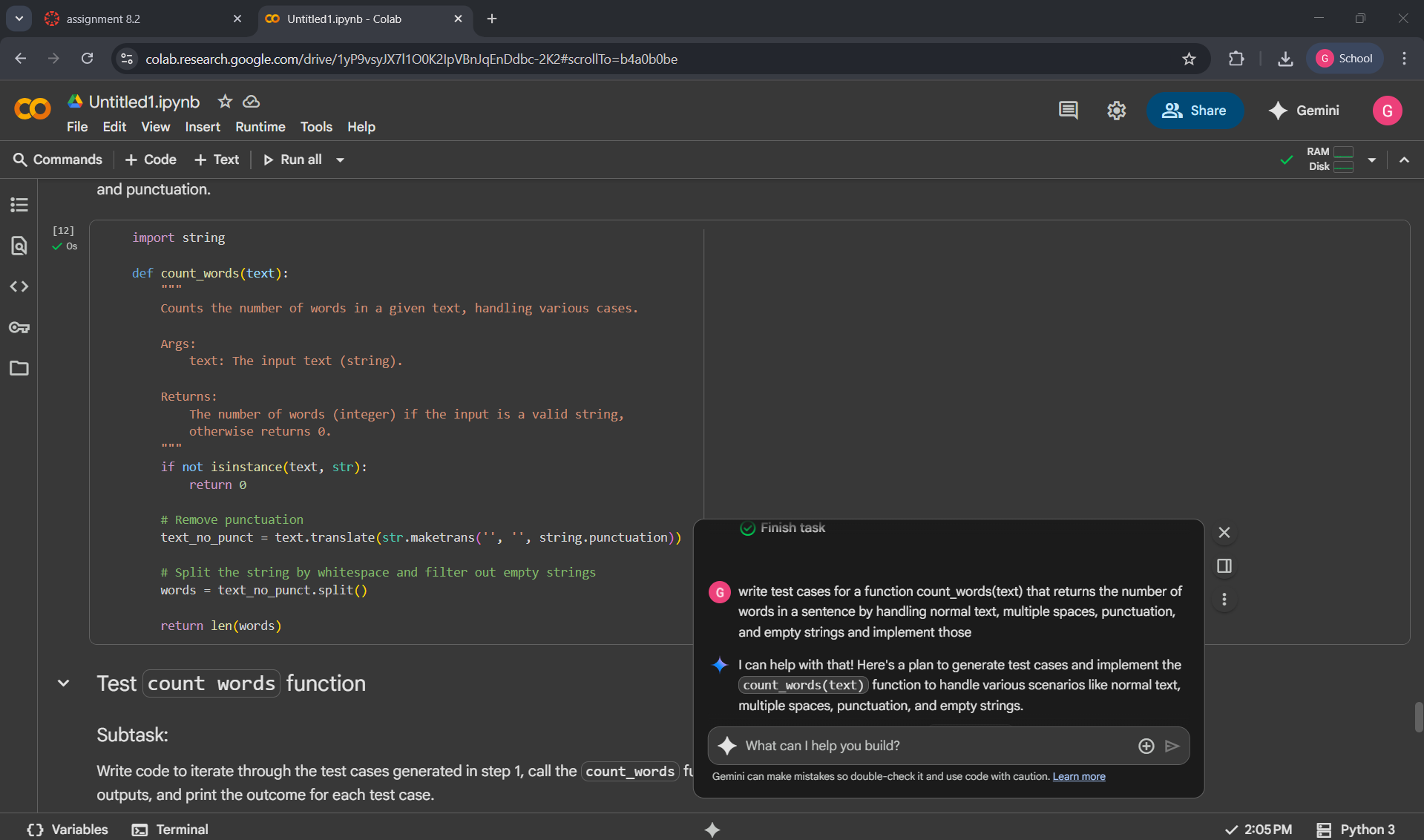


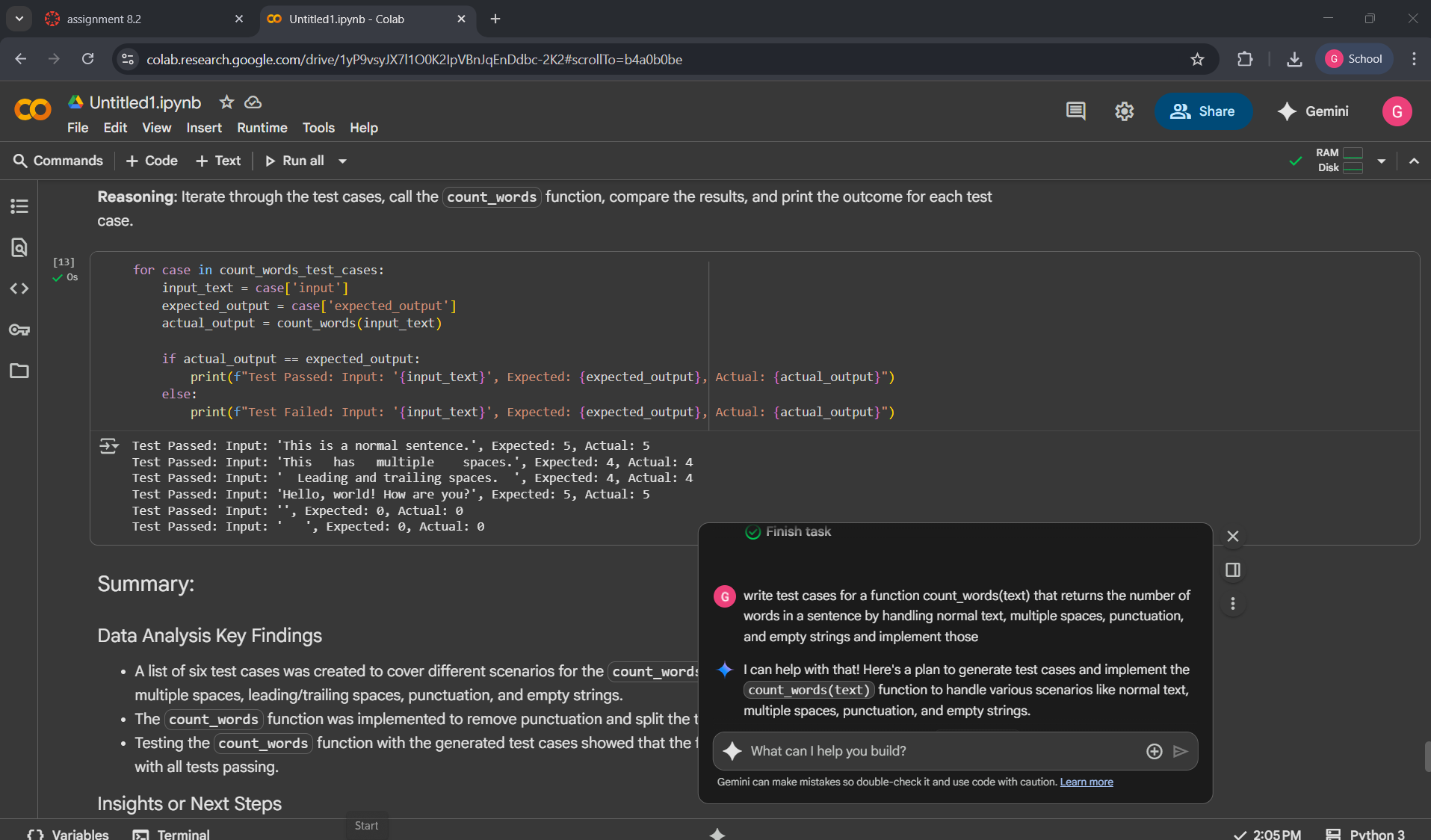


EXPLANATION:

* Test cases were successfully generated for both celsius\_to\_fahrenheit and fahrenheit\_to\_celsius functions, including integer, decimal, known pair, and invalid inputs.
* The celsius\_to\_fahrenheit function was implemented to correctly convert Celsius to Fahrenheit, handling decimal inputs and returning None for invalid types.

TASK 3:

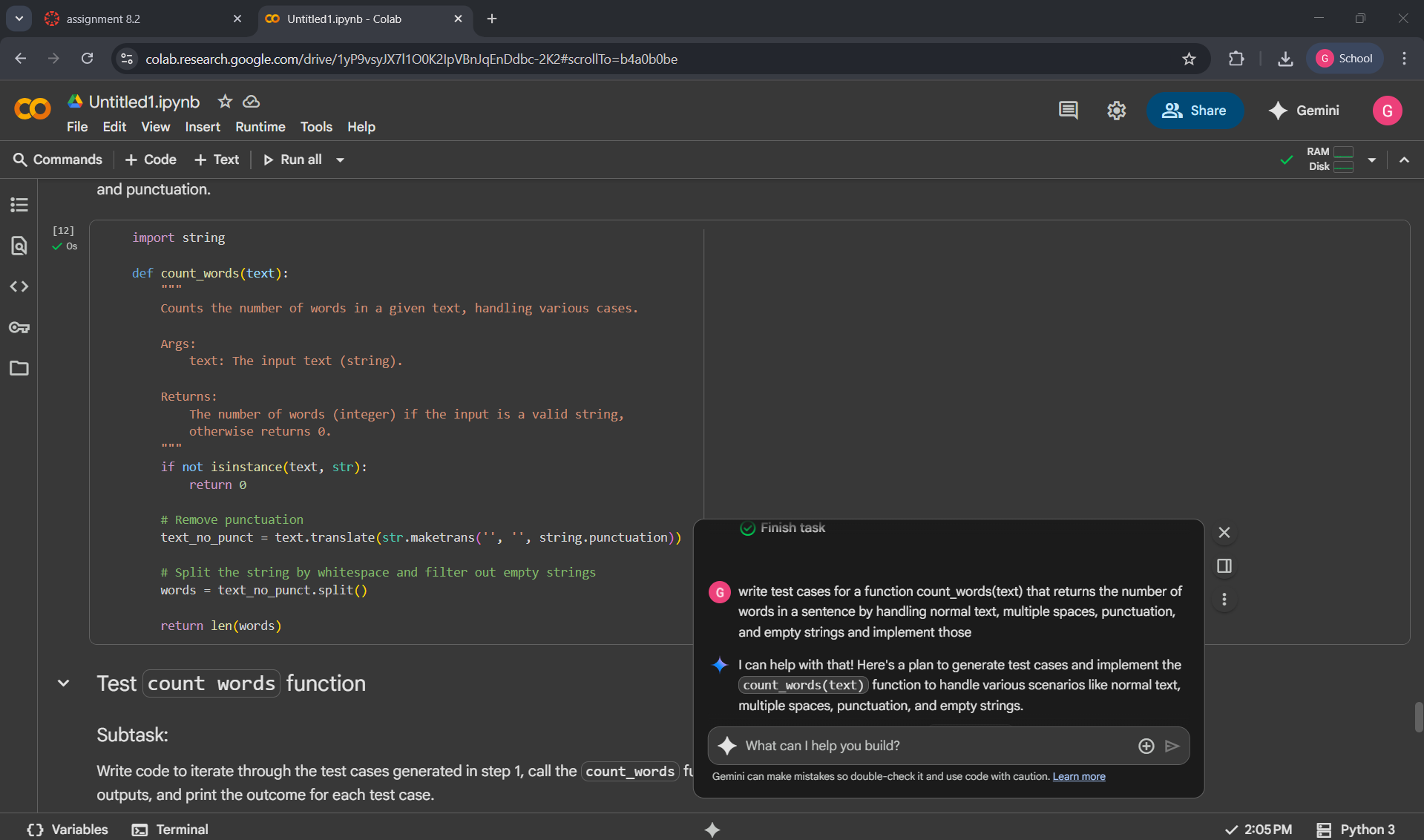


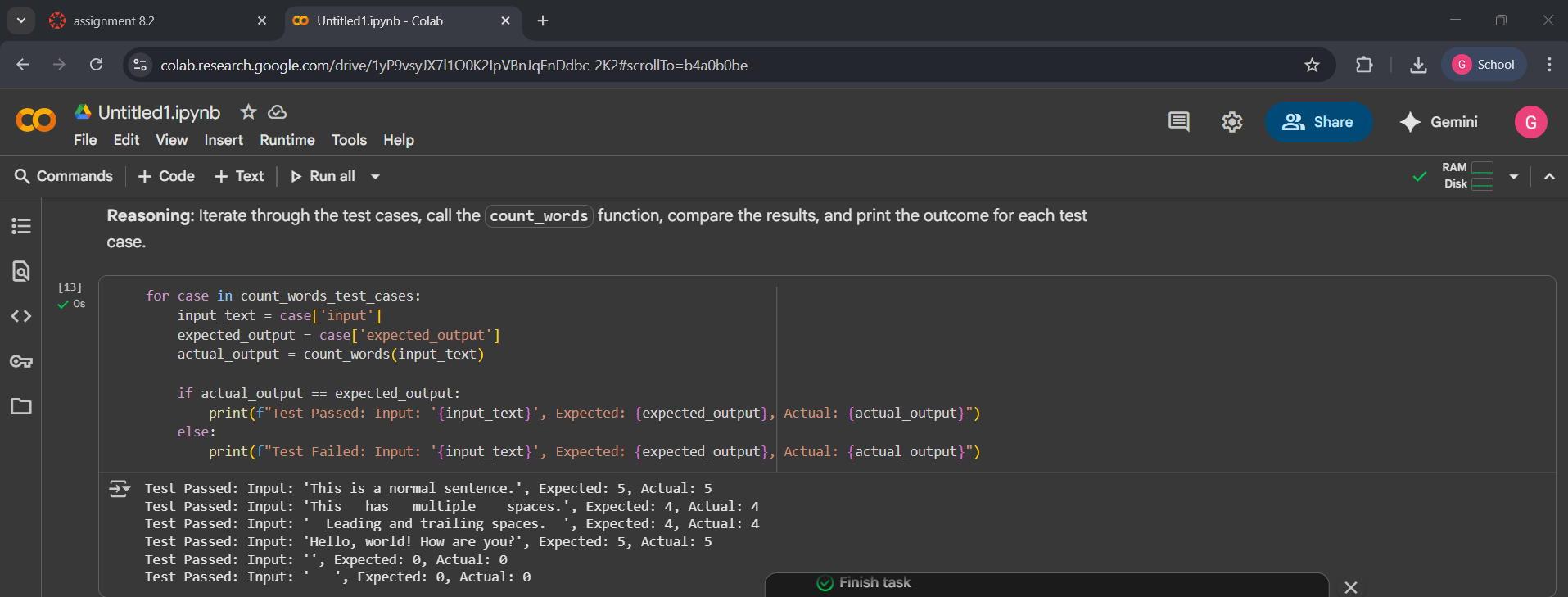


EXPLANATION:

* A list of six test cases was created to cover different scenarios for the count\_words function, including normal text, text with multiple spaces, leading/trailing spaces, punctuation, and empty strings.
* The count\_words function was implemented to remove punctuation and split the text by spaces to count words.
* Testing the count\_words function with the generated test cases showed that the function correctly handled all defined scenarios, with all tests passing.

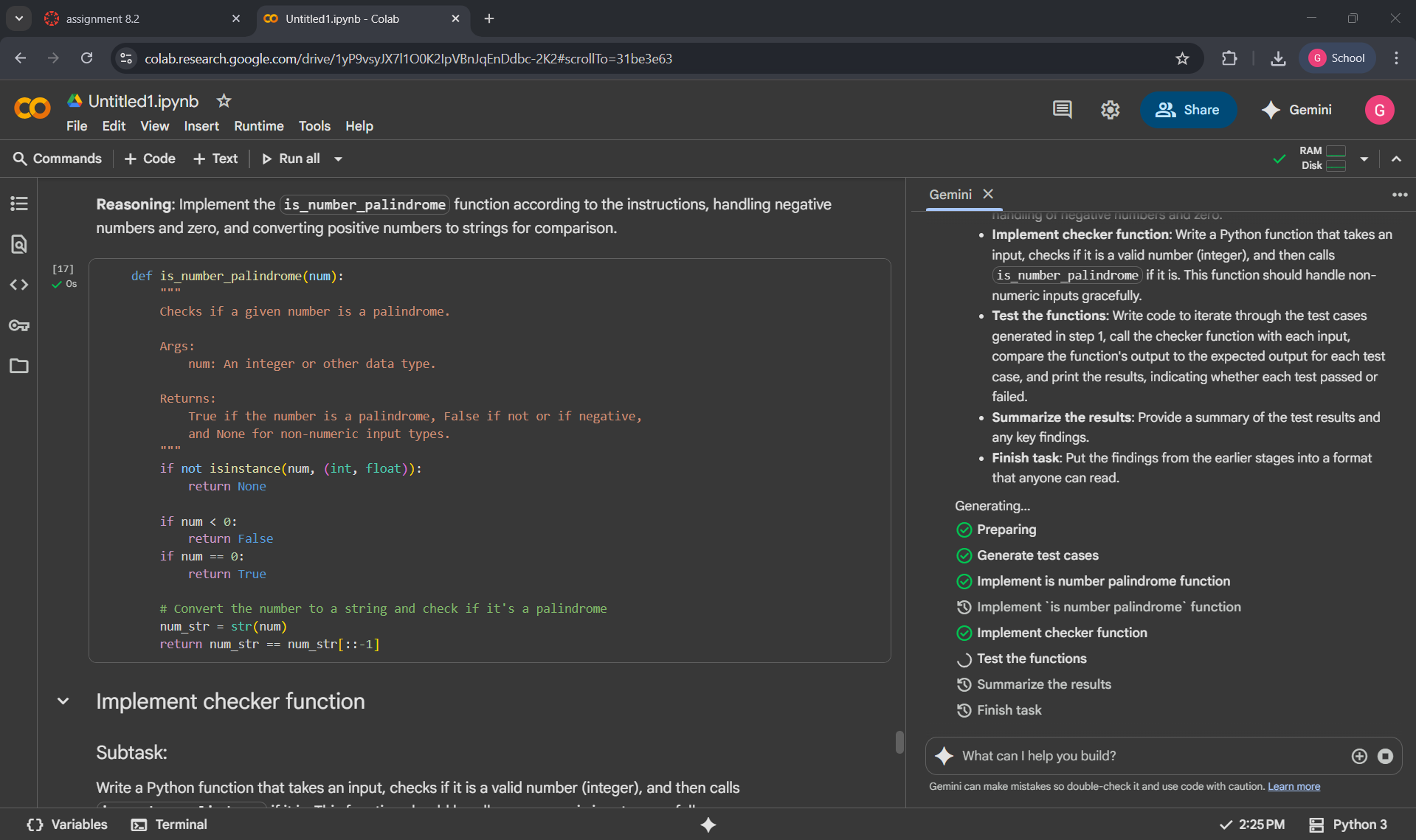
TASK 4:

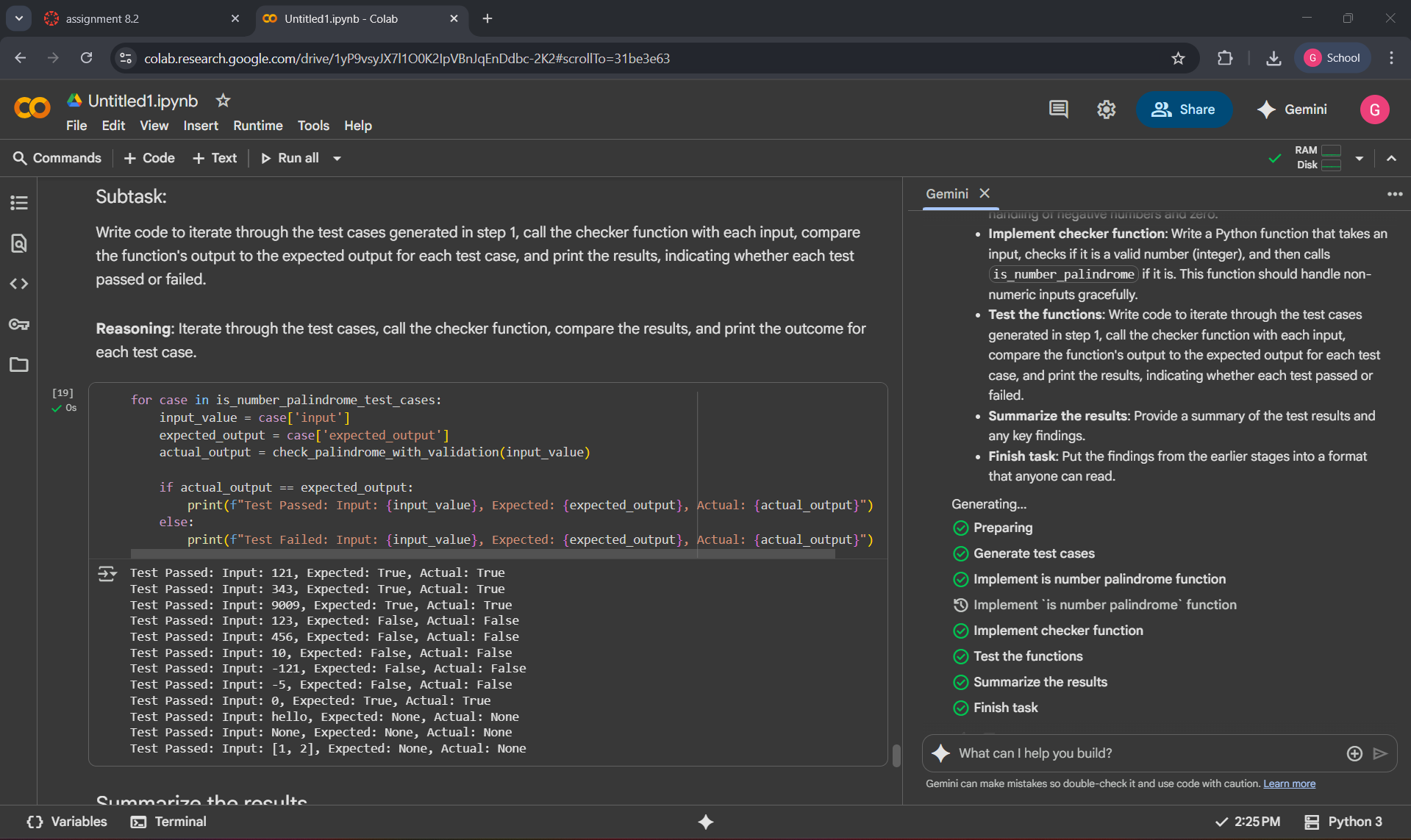




* EXPLANATION:
* A list of six test cases was created to cover different scenarios for the count\_words function, including normal text, text with multiple spaces, leading/trailing spaces, punctuation, and empty strings.
* The count\_words function was implemented to remove punctuation and split the text by spaces to count words.
* Testing the count\_words function with the generated test cases showed that the function correctly handled all defined scenarios, with all tests passing.

TASK 5:





EXPLANATION:

* A list of 13 test cases was generated for the is\_number\_palindrome function, including positive palindromes (121, 343, 9009), positive non-palindromes (123, 456, 10), negative numbers (-121, -5), zero (0), and non-numeric inputs ("hello", None, [1, 2]).
* The is\_number\_palindrome function was implemented to return True for positive integer palindromes and zero, False for negative numbers and positive non-palindromes, and None for non-numeric inputs.
* A check\_palindrome\_with\_validation function was implemented to first validate if the input can be converted to an integer using a try-except block. If successful, it calls is\_number\_palindrome; otherwise, it returns None.
* Testing the check\_palindrome\_with\_validation function with all 13 generated test cases resulted in all tests passing, indicating the function correctly handled all specified input types and scenarios.