2403A51286  
AI Assisted Coding 04-09-2025  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

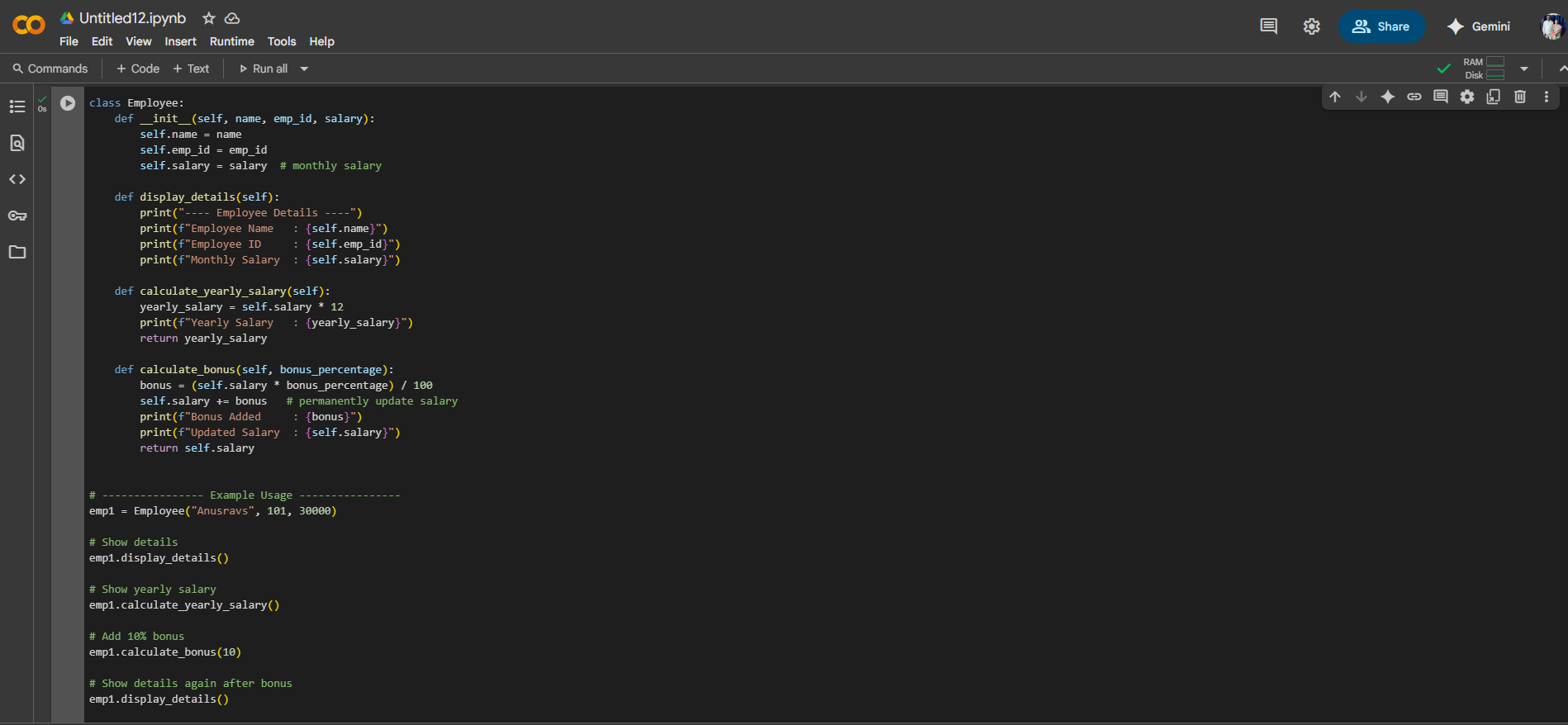
**Task#1**Q: 1 (Classes – Employee Management)

* Task: Use AI to create an Employee class with attributes (name, id, salary) and a method to calculate yearly salary.
* Instructions:
  + Prompt AI to generate the Employee class.
  + Analyze the generated code for correctness and structure.
  + Ask AI to add a method to give a bonus and recalculate salary.

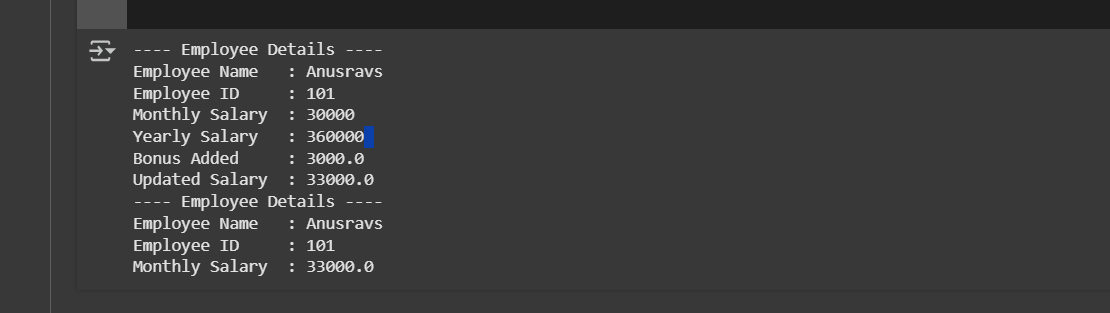
**Prompt:**

Create a Python class called Employee with the following features:

* Attributes: name, id, and salary.
* A method display\_details() to print employee information.
* A method calculate\_yearly\_salary() that returns yearly salary (salary × 12).
* A method calculate\_bonus(bonus\_percentage) that calculates bonus, adds it to salary, and updates the salary.

**Code:**  


**Output:**



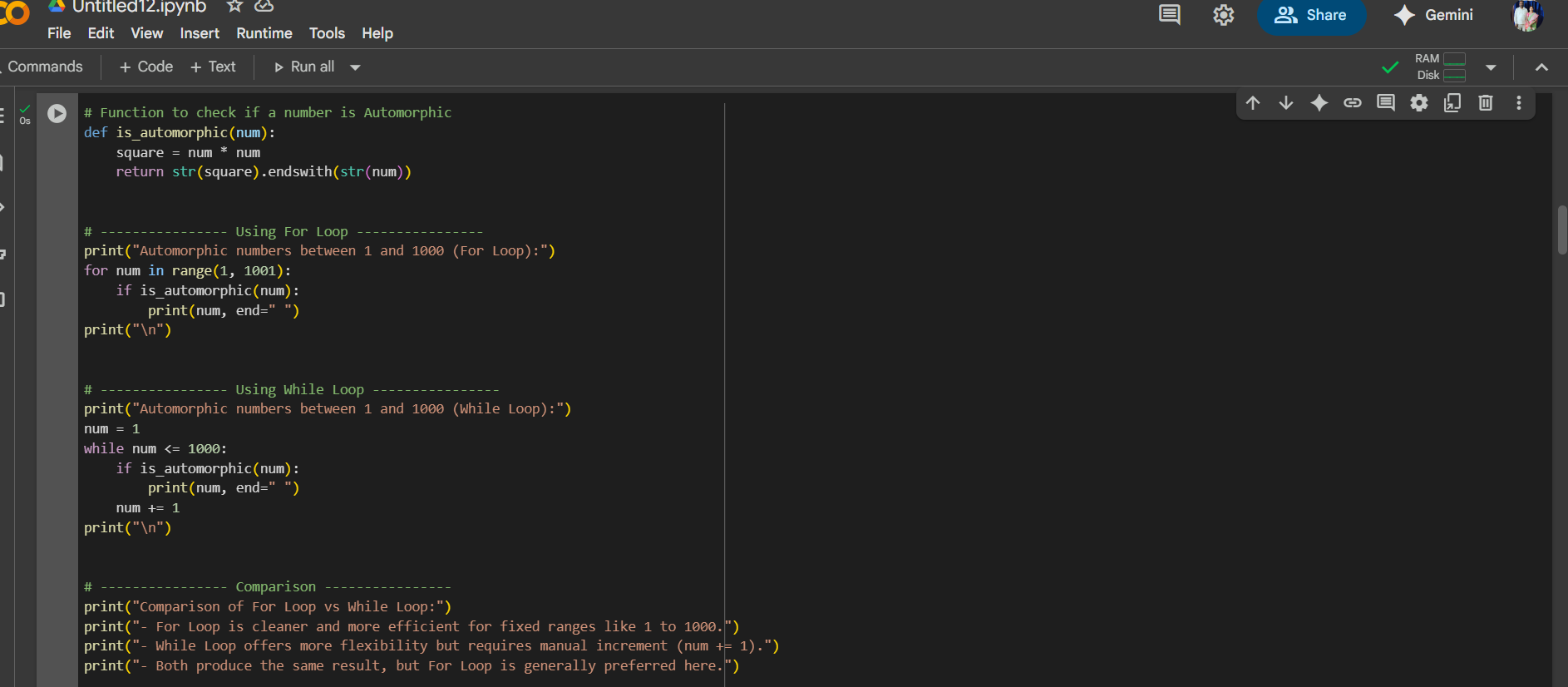
**Task#2**  
Q: 2 (Loops – Automorphic Numbers in a Range)

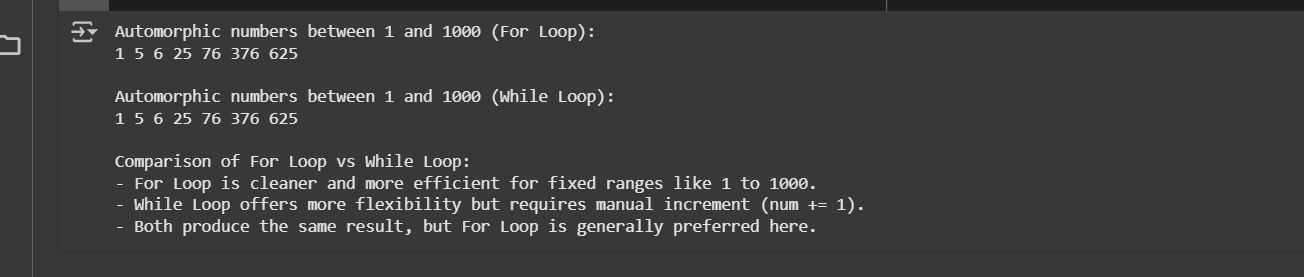
* Task: Prompt AI to generate a function that displays all Automorphic numbers between 1 and 1000 using a for loop.
* Instructions:
  + Get AI-generated code to list Automorphic numbers using a for loop.
  + Analyze the correctness and efficiency of the generated logic.
  + Ask AI to regenerate using a while loop and compare both implementations.

**Prompt:**

Write a Python program to find all Automorphic numbers between 1 and 1000.

1. First, generate the solution using a **for loop**.
2. Then, rewrite the solution using a **while loop**.
3. Finally, explain the difference between the two implementations and analyze which one is more efficient.

**Code:  
**

**Output:**

**Task#3**

Q:3 (Conditional Statements – Online Shopping Feedback Classification)

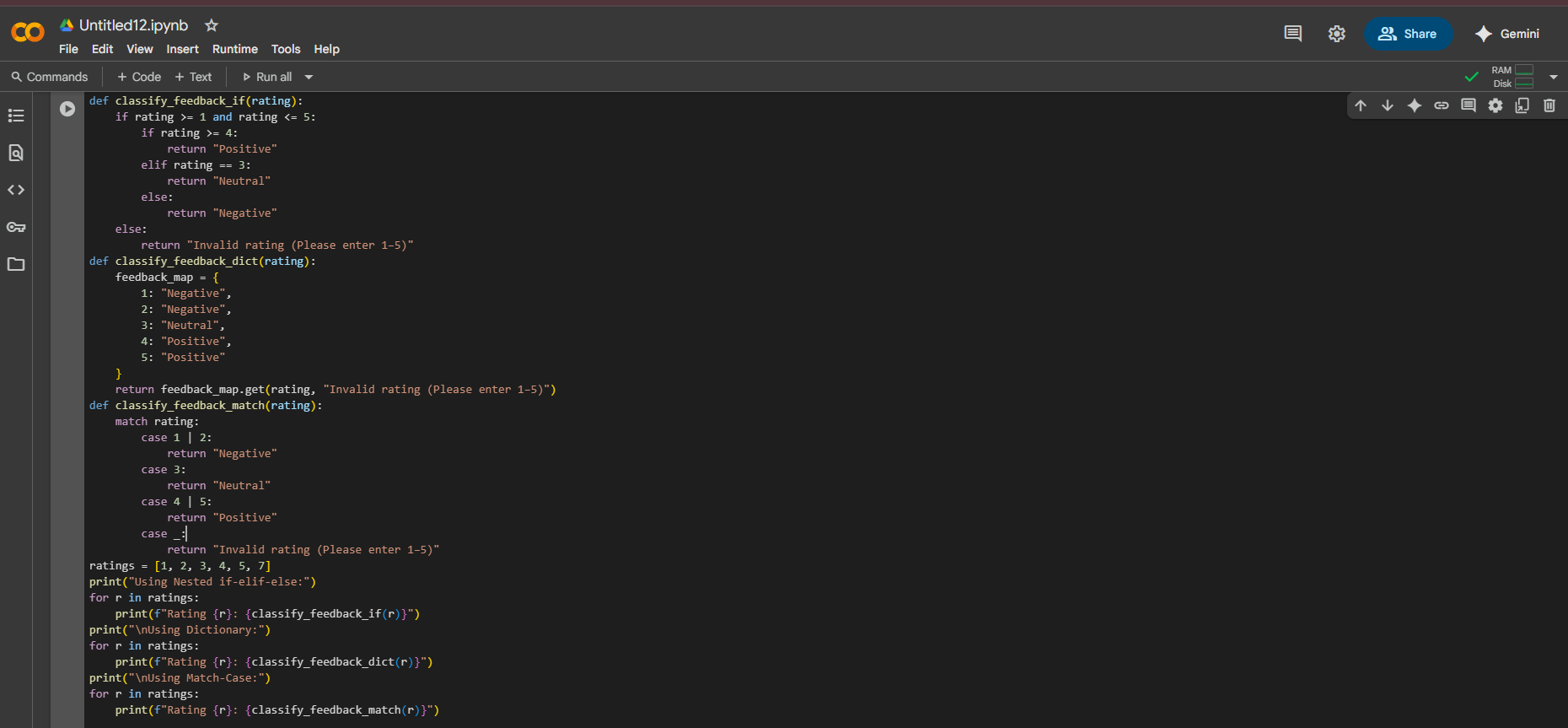
* Task: Ask AI to write nested if-elif-else conditions to classify online shopping feedback as Positive, Neutral, or Negative based on a numerical rating (1–5).
* Instructions:
  + Generate initial code using nested if-elif-else.
  + Analyze correctness and readability.
  + Ask AI to rewrite using dictionary-based or match-case structure.

**Prompt:**

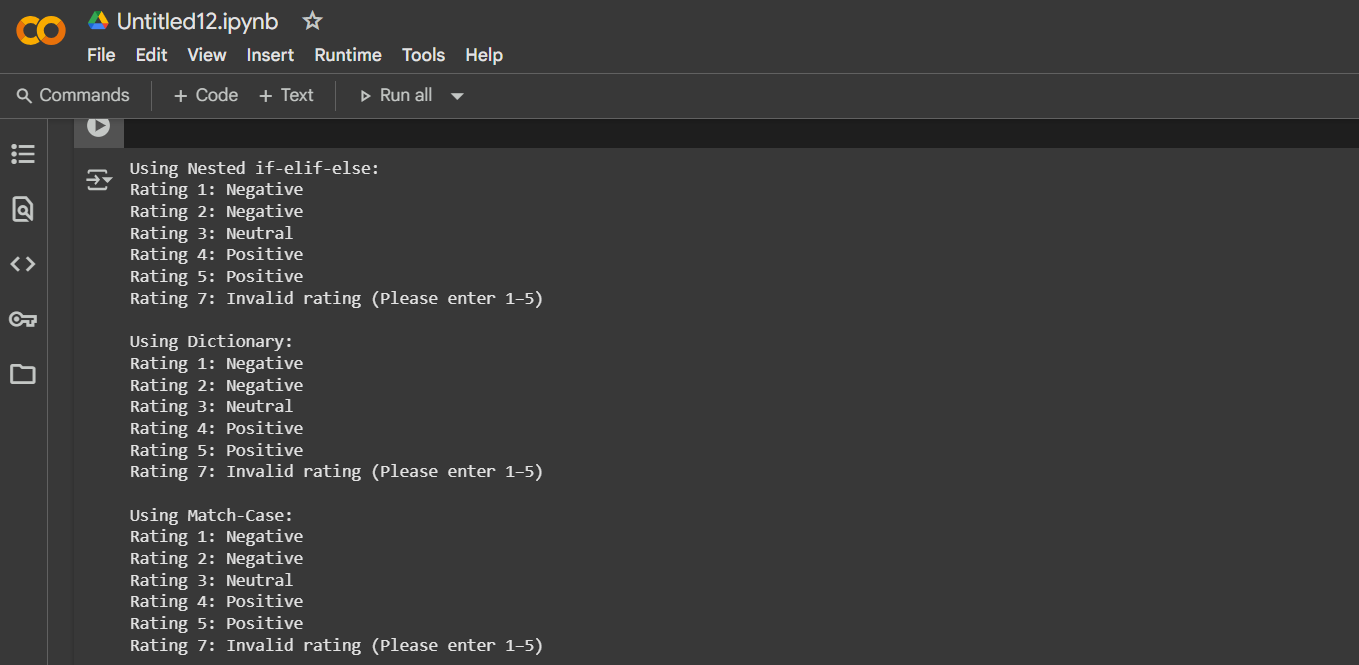
Write a Python program to classify online shopping feedback based on a rating (1–5).

1. First, implement it using **nested if-elif-else** statements:
   * Ratings 4–5 → Positive
   * Rating 3 → Neutral
   * Ratings 1–2 → Negative
2. Then, analyze the correctness and readability of the nested if-else code.
3. Rewrite the program using either a **dictionary-based approach** or Python’s **match-case** structure for cleaner code

**Code:**

****

**Output:**



**Task#4**

Q:4 Loops – Prime Numbers in a Range)

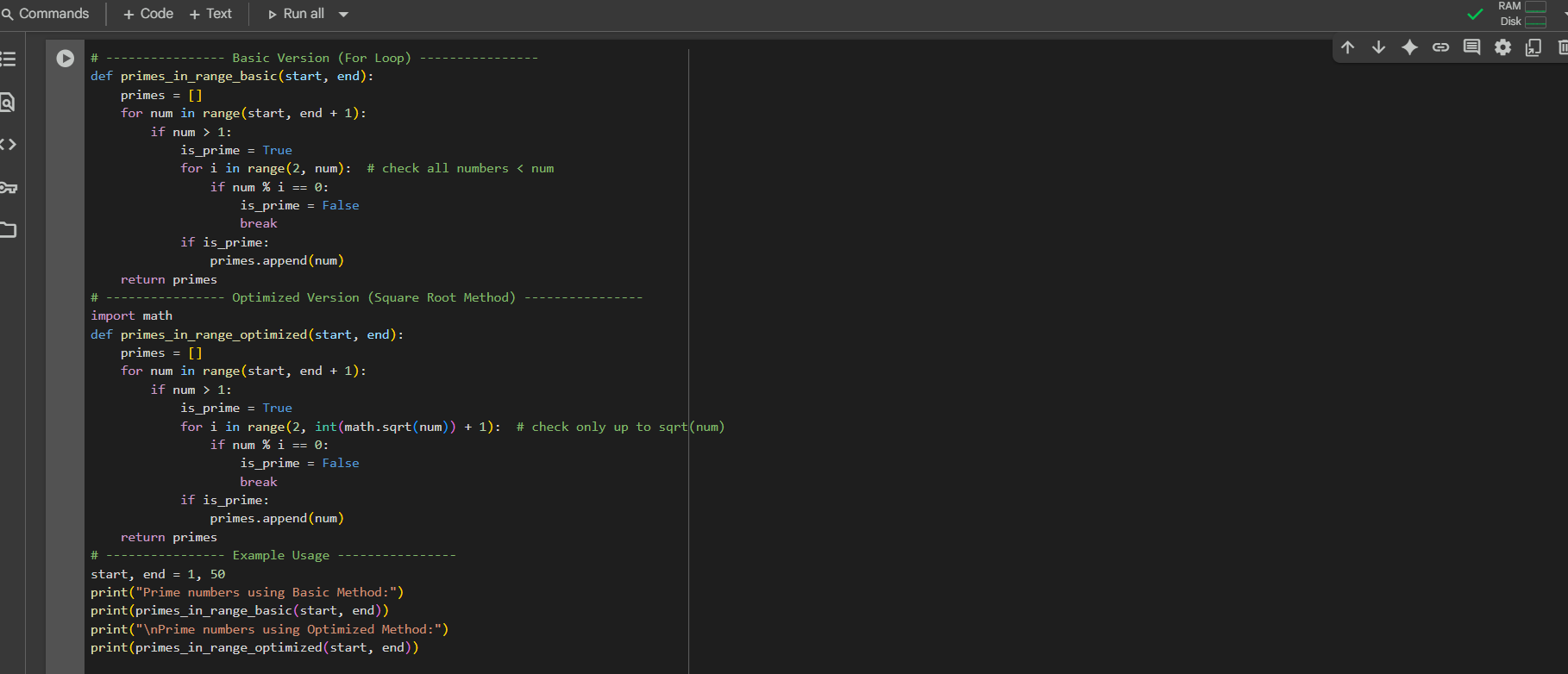
* Task: Generate a function using AI that displays all prime numbers within a user-specified range (e.g., 1 to 500).
* Instructions:
  + Get AI-generated code to list all primes using a for loop.
  + Analyze the correctness and efficiency of the prime-checking logic.
  + Ask AI to regenerate an optimized version (e.g., using the square root method).

**Prompt:**

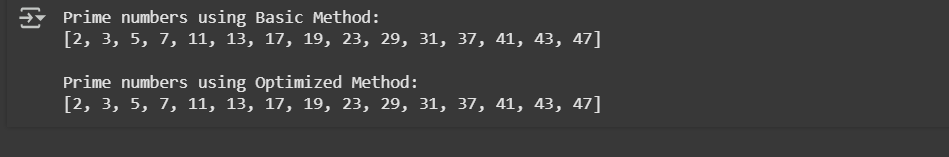
Write a Python program to display all prime numbers within a user-specified range (e.g., 1 to 500).

1. First, implement the solution using a **for loop** and a simple prime-checking method.
2. Analyze the correctness and efficiency of the prime-checking logic.
3. Rewrite the function using an **optimized approach** (checking divisibility only up to the square root of the number).

**Code:**

****

Output:



**Task#5:**

Q:5 (Classes – Library System)

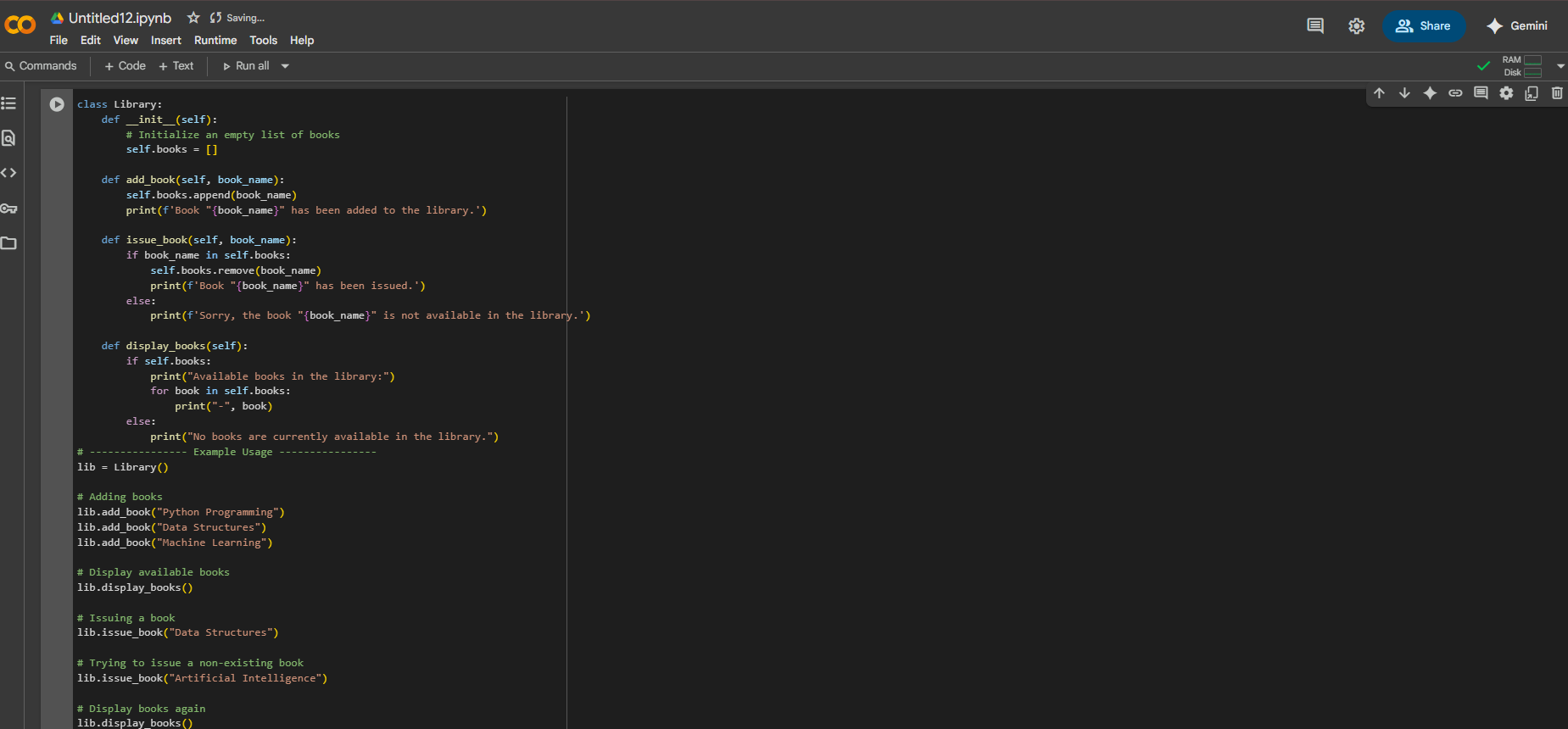
* Task: Use AI to build a Library class with methods to add\_book(), issue\_book(), and display\_books().
* Instructions:
  + Generate Library class code using AI.
  + Analyze if methods handle edge cases (e.g., issuing unavailable books).
  + Ask AI to add comments and documentation.

**Prompt:**

Write a Python program to implement a Library class with the following features:

1. A method add\_book(book\_name) to add books to the library.
2. A method issue\_book(book\_name) to issue a book if available, otherwise display a message if it’s not in the library.
3. A method display\_books() to show all available books.
4. Handle edge cases such as trying to issue a book that is not in the collection.
5. Add comments and documentation to explain the code clearly.

**Code:**



**Output:**

