2403a51024

Vardhan

AI exam set I

## Subgroup I

### I.1 — [S13I1] Top-3 frequent words

Scenario (agritech):

Context:

Basic text analytics in agritech requires most frequent terms for summaries.

Your Task:

Return the top-3 words by frequency, breaking ties lexicographically.

Data & Edge Cases:

Normalize to lowercase, split on spaces; ignore punctuation for simplicity (optional).

AI Assistance Expectation:

Use AI to propose Counter/ sorting approach and tie-breaking mechanics.

Constraints & Notes:

Stable ordering by (-count, word).

Sample Input

to be or not to be that is the question

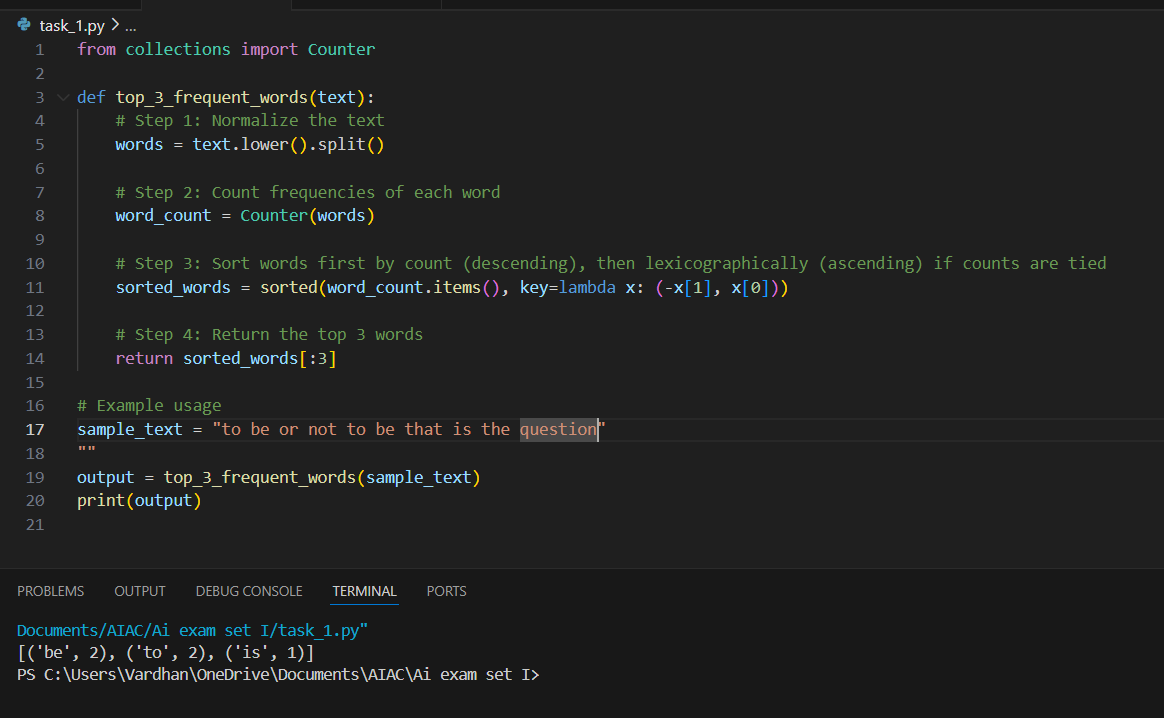
Sample Output

[('to', 2), ('be', 2), ('is', 1)]

Acceptance Criteria: Tie-breaking lexicographically

Prompt :

Create the code that using python program Basic text analytics in agritech requires most frequent terms for summaries. Return the top-3 words by frequency, breaking ties lexicographically.



## **I.2 — [S13I2] Generate Docstrings for Crop Yield Function**

### **Scenario (Agritech):**

**Context:**  
An agritech analytics team has a function that calculates **average crop yield per acre**, but it lacks documentation. The function works but is not self-explanatory. New developers find it confusing.

### **Your Task:**

* Write code **average crop yield per acre function, that took** total\_yield and acres as input and return **average crop yield per acre value.**
* Use AI assistance to generate a **clear, structured docstring** for the given function.
* Ensure the docstring covers:
  + **Parameters** (types and meaning)
  + **Return type**
  + **Example usage**
  + **Notes on edge cases**

### **Data & Edge Cases:**

* total\_yield = 500, acres = 50 → 10
* Division by zero should be highlighted as a possible error.
* Input values should be numeric.

### **AI Assistance Expectation:**

* Use AI to propose a **PEP-257 compliant docstring**.
* AI should also suggest **type hints** (float for return type).
* Generate **doctests** inside the docstring.

### **Constraints & Notes:**

* Do not modify function logic, only add documentation & type hints.
* Keep docstring concise but informative.

### **Sample Input:**

print(avg\_yield\_per\_acre(500, 50))

### **Sample Output:**

10.0

### **Acceptance Criteria:**

* Function includes a proper docstring.
* Type hints added:
* def avg\_yield\_per\_acre(total\_yield: float, acres: float) -> float
* Doctest runs correctly.
* Reviewed for readability and clarity.

Prompt :

Write the python program that using An agritech analytics team has a function that calculates **average crop yield per acre**, but it lacks documentation. The function works but is not self-explanatory. New developers find it confusing.

