Implement Trie

Try to solve the Implement Trie problem.

We'll cover the following

Statement

Trie is a tree-like data structure used to store strings. The tries are also called **prefix trees** because they provide very efficient prefix-matching operations. Implement a trie data structure with three functions that perform the following tasks:

- **Insert (word):** This inserts a word into the trie.
- **Search (word):** This searches the given word in the trie and returns TRUE, if found. Otherwise, return FALSE.
- **Search prefix (prefix):** This searches the given prefix in the trie and returns TRUE, if found. Otherwise, return FALSE.

Constraints:

- $1 \leq$ `word.length`, `prefix.length` ≤ 2000
- The strings consist only of lowercase English letters.
- At most, 3×10^3 calls in total will be made to the functions.

Examples

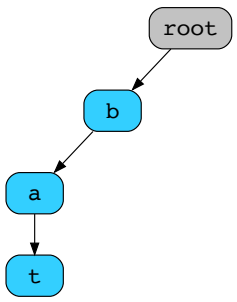
The **Insert** function does not return anything. The **Search** and **Search prefix** functions return TRUE if the input is found in the trie. Otherwise, they will return FALSE.

Sample example 1

Input

```
Insert('bat')
```

Output



Understand the problem

Let's take a moment to make sure you've correctly understood the problem. The quiz below helps you check if you're solving the correct problem:

Implement Trie

1 Given a trie with the words `[bat, bag, make, broom, bait]`, what will be the output of the following query?

```
search('bait')
```

A) TRUE

B) FALSE

Submit Answer

Question 1 of 3
0 attempted

Reset Quiz

Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

> **Note**: The game below is only for the **Insert** function.

Drag and drop the cards to rearrange them in the correct sequence.

Begin from the root node and iterate over the string one character at a time.

For each character, check if the current character exists as a child node of the current node.

If the character is found, move to that child node and continue to the next character.

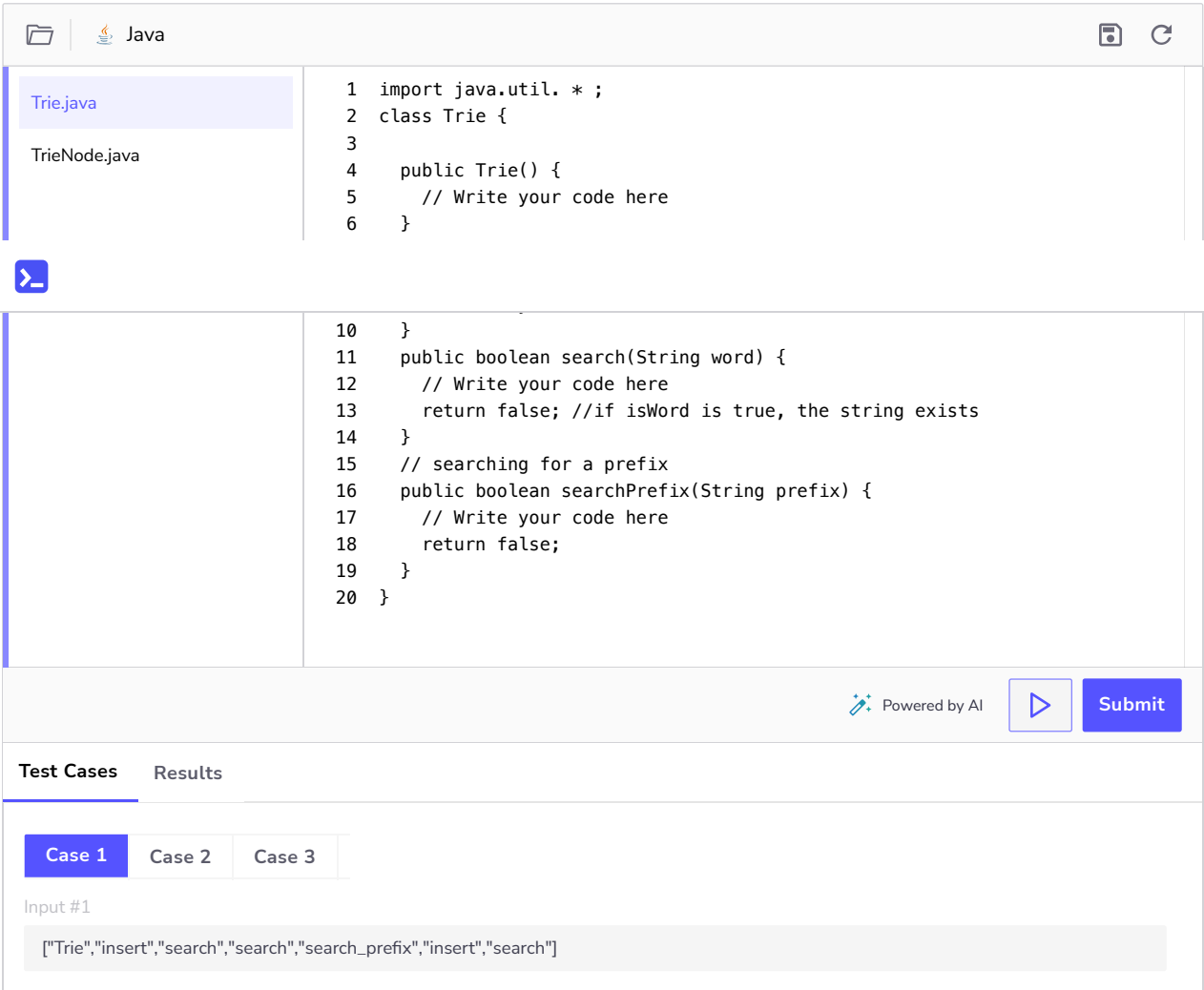
If the character is not found, create a new node for that character, link it as a child of the current node, and move to the newly created node.

For the last character of the word, set a boolean variable to TRUE for the corresponding node to indicate the end of the word.

Reset | Show Solution | Submit

Try it yourself

Implement your solution in `Trie.java` in the following coding playground. You will need the provided supporting code to implement your solution.



Input #2

[[],["apple"],["apple"],["app"],["app"],["app"],["app"]]

Implement Trie

Back

Trie: Introduction

Next

Solution: Implement T...

Mark as Completed