75 Days of Code
Day 70
Problem no : Leetcode 208
Problem Title : Implement Trie (Prefix Tree)
Problem type : Trie

.
A trie (pronounced as "try") or prefix tree is a tree data structure used to efficiently store and retrieve keys in a dataset of strings. There are various applications of this data structure, such as autocomplete and spellchecker.

Implement the Trie class:

Trie() Initializes the trie object.
void insert(String word) Inserts the string word into the trie.
boolean search(String word) Returns true if the string word is in the trie (i.e., was inserted before), and false otherwise.
boolean startsWith(String prefix) Returns true if there is a previously inserted string word that has the prefix prefix, and false otherwise.

Example 1:

Input
["Trie", "insert", "search", "search", "startsWith", "insert", "search"]
[[], ["apple"], ["apple"], ["app"], ["app"], ["app"], ["app"]]
Output
[null, null, true, false, true, null, true]

Explanation
Trie trie = new Trie();
trie.insert("apple");

```
trie.search("apple");   // return True
trie.search("app");     // return False
trie.startsWith("app"); // return True
trie.insert("app");
trie.search("app");     // return True
```

```
class TrieNode{
    children :Map<string,TrieNode>;
    isEndOfWord:boolean
    constructor(){
        this.children = new Map();
        this.isEndOfWord = false;
    }
}

class Trie {
    root :TrieNode;
    constructor() {
        this.root = new TrieNode();
    }

    insert(word: string): void {
        let node = this.root;
        for(const char of word){
            if(!node.children.has(char)){
                node.children.set(char,new TrieNode());
            }
            node = node.children.get(char)
        }
        node.isEndOfWord = true
    }

    search(word: string): boolean {
        let node = this.root;
        for(const char of word){
            if(!node.children.has(char)){
                return false;
            }
            node = node.children.get(char);
        }
        return node.isEndOfWord;

    }

    startsWith(prefix: string): boolean {
        let node = this.root;
        for(const char of prefix){
            if(!node.children.has(char)){
                return false;
            }
            node = node.children.get(char)
        }
        return true;
    }
}

/**
```

Accepted

Runtime

Details

177 ms

Beats 60.89% of users with TypeScript

Memory

72.05 MB

Beats 50.97% of users with TypeScript