트라이의 insert, search, startsWith 메소드를 구현하라.

1.딕셔너리 활용

```
import collections

class TrieNode:
    def __init__(self):
        self.word = False
        self.children = collections.defaultdict(TrieNode)

class Trie:
    def __init__(self):
        """
        Initialize your data structure here.
        """
        self.root = TrieNode()


    def insert(self, word: str) -> None:
        """
        Inserts a word into the trie.
        """
        node = self.root
        for char in word:
            node = node.children[char]
        node.word = True


    def search(self, word: str) -> bool:
        """
        Returns if the word is in the trie.
        """
        node = self.root
        for char in word:
            if char not in node.children:
                return False
            node = node.children[char]
        return node.word


    def startsWith(self, prefix: str) -> bool:
        """
        Returns if there is any word in the trie that starts with the given prefix.
        """
        node = self.root
        for char in prefix:
            if char not in node.children:
                return False
            node = node.children[char]
        return True
```

```
# Your Trie object will be instantiated and called as such:
# obj = Trie()
# obj.insert(word)
# param_2 = obj.search(word)
# param_3 = obj.startsWith(prefix)
```