Task 2: Trie for Prefix Checking. Implement a trie data structure in JAVA that supports insertion of strings and provides a method to check if a given string is a prefix of any word in the trie.

1. **Implementation:**
   - **TrieNode Class:** Represents each node in the trie.
   - **Trie Class:** Contains methods to insert strings and check if a given string is a prefix of any word in the trie.

2. CODE:

```
import java.util.HashMap;
import java.util.Map;
public class TrieNode {
    Map<Character, TrieNode> children;
    boolean isEndOfWord;
    public TrieNode() {
        children = new HashMap<>();
        isEndOfWord = false;
    }
}
public class Trie {
    private TrieNode root;
    public Trie() {
        root = new TrieNode();
    }
    public void insert(String word) {
        TrieNode currentNode = root;
        for (char ch : word.toCharArray()) {
            currentNode = currentNode.children.computeIfAbsent(ch, c -> new TrieNode());
        }
        currentNode.isEndOfWord = true;
```

```
    }
    public boolean startsWith(String prefix) {
        TrieNode currentNode = root;
        for (char ch : prefix.toCharArray()) {
            currentNode = currentNode.children.get(ch);
            if (currentNode == null) {
                return false;
            }
        }
        return true;    }          }
public class TrieTest {
    public static void main(String[] args) {
        Trie trie = new Trie();
        // Insert words into the trie
        trie.insert("apple");
        trie.insert("app");
        trie.insert("banana");
        trie.insert("band");
        trie.insert("bandana");

        // Check if certain prefixes are present in the trie
        System.out.println("Prefix 'app': " + trie.startsWith("app")); // true
        System.out.println("Prefix 'ban': " + trie.startsWith("ban")); // true
        System.out.println("Prefix 'band': " + trie.startsWith("band")); // true
        System.out.println("Prefix 'bat': " + trie.startsWith("bat")); // false
        System.out.println("Prefix 'cat': " + trie.startsWith("cat")); // false
    }
}
```

Explanation:

1. **TrieNode Class:**
   - Represents each node in the trie.
   - Uses a Map<Character, TrieNode> to store the children of each node, making it easy to traverse the trie.
   - A boolean flag isEndOfWord is used to mark the end of a word.
2. **Trie Class:**
   - Contains the root node of the trie.
   - insert(String word): Inserts a word into the trie. It traverses the trie character by character, creating new nodes as necessary. At the end of the word, it marks the last node as isEndOfWord.
   - startsWith(String prefix): Checks if a given string is a prefix of any word in the trie. It traverses the trie character by character. If it encounters a character that is not in the trie, it returns false. If it successfully traverses all characters of the prefix, it returns true.
3. **Main Method:**
   - Creates a Trie object and inserts several words into it.
   - Checks if certain prefixes are present in the trie and prints the results.