No Prefix Set

```
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);

/*
 * Complete the 'noPrefix' function below.
 *
 * The function accepts STRING_ARRAY words as parameter.
 */

struct TrieNode{
    bool isEnd;
    unordered_map<char,TrieNode*> children;
    TrieNode():isEnd(false){}
};

bool insertWord(TrieNode* root,const string& word){
    TrieNode* curr=root;
    for(int i=0;i<word.size();i++){
        char c=word[i];
        if(!curr->children[c]) curr->children[c]=new TrieNode();
        curr=curr->children[c];
        if(curr->isEnd) return false;
    }
    if(!curr->children.empty()) return false;
    curr->isEnd=true;
    return true;
}

void noPrefix(vector<string> words){
    TrieNode* root=new TrieNode();
    for(string& word:words){
        if(!insertWord(root,word)){
            cout<<"BAD SET\n"<<word<<"\n";
            return;
        }
    }
    cout<<"GOOD SET\n";
}
```

```
int main()
{
    string n_temp;
    getline(cin, n_temp);

    int n = stoi(ltrim(rtrim(n_temp)));

    vector<string> words(n);

    for (int i = 0; i < n; i++) {
        string words_item;
        getline(cin, words_item);

        words[i] = words_item;
    }

    noPrefix(words);

    return 0;
}

string ltrim(const string &str) {
    string s(str);

    s.erase(
        s.begin(),
        find_if(s.begin(), s.end(), not1(ptr_fun<int,
int>(isspace)))
    );

    return s;
}

string rtrim(const string &str) {
    string s(str);

    s.erase(
        find_if(s.rbegin(), s.rend(), not1(ptr_fun<int,
int>(isspace))).base(),
        s.end()
    );

    return s;
}
```