```cpp
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

#include <vector>
#include <unordered_map>
using namespace std;

class TrieNode {
public:
  unordered_map<char, TrieNode *> children;
};

class ModifiedSuffixTrie {
public:
  TrieNode *root;

  ModifiedSuffixTrie(string str) {
    this->root = new TrieNode();
    this->populateModifiedSuffixTrieFrom(str);
  }

  void populateModifiedSuffixTrieFrom(string str) {
    for (int i = 0; i < str.length(); i++) {
      this->insertSubstringStartingAt(i, str);
    }
  }

  void insertSubstringStartingAt(int i, string str) {
    TrieNode *node = this->root;
    for (int j = i; j < str.length(); j++) {
      char letter = str[j];
      if (node->children.find(letter) == node->children.end()) {
        TrieNode *newNode = new TrieNode();
        node->children.insert({letter, newNode});
      }
      node = node->children[letter];
    }
  }

  bool contains(string str) {
    TrieNode *node = this->root;
    for (char letter : str) {
      if (node->children.find(letter) == node->children.end()) {
        return false;
      }
      node = node->children[letter];
    }
    return true;
  }
};

// O(b^2 + ns) time | O(b^2 + n) space
vector<bool> multiStringSearch(string bigString, vector<string> smallStrings) {
  ModifiedSuffixTrie modifiedSuffixTrie(bigString);
  vector<bool> solution;
  for (string smallString : smallStrings) {
    solution.push_back(modifiedSuffixTrie.contains(smallString));
  }
  return solution;
}

```