Sparse Arrays

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

using namespace std;

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

/*
 * Complete the 'matchingStrings' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
 * The function accepts following parameters:
 *  1. STRING_ARRAY strings
 *  2. STRING_ARRAY queries
 */

vector<int> matchingStrings(vector<string> strings,
vector<string> queries) {
    int s1=strings.size();
    int s2=queries.size();
    vector<int>ans;
    for(int i=0;i<s2;i++){
        int count=0;
        for(int j=0;j<s1;j++){
            if(queries[i]==strings[j]){
                count++;
            }
        }
        ans.push_back(count);
    }
    return ans;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string strings_count_temp;
    getline(cin, strings_count_temp);

    int strings_count = stoi(ltrim(rtrim(strings_count_temp)));

    vector<string> strings(strings_count);
    for (int i = 0; i < strings_count; i++) {
        string strings_item;
        getline(cin, strings_item);

        strings[i] = strings_item;
    }

    string queries_count_temp;
    getline(cin, queries_count_temp);

    int queries_count = stoi(ltrim(rtrim(queries_count_temp)));

    vector<string> queries(queries_count);

    for (int i = 0; i < queries_count; i++) {
        string queries_item;
        getline(cin, queries_item);

        queries[i] = queries_item;
    }

    vector<int> res = matchingStrings(strings, queries);

    for (size_t i = 0; i < res.size(); i++) {
        fout << res[i];

        if (i != res.size() - 1) {
            fout << "\n";
        }
    }

    fout << "\n";

    fout.close();

    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;
}
```