```cpp
#include <fstream>
#include <string>
#include <map>
#include <sstream>
#include <iostream>
#include <vector>
#include <deque>

using namespace std;

using tabla = map < string, deque<int> >;


bool resuelveCaso() {

    int n;

    string frase, palabra;

    cin >> n;

    if (n == 0) {
        return false;
    }
    else {

        cin.ignore();

        tabla listado;

        for (int h = 1; h <= n; h++) {

            getline(cin, frase);

            stringstream ss(frase);

            while (ss >> palabra) {

                if (palabra.length() > 2) {

                    for (size_t i = 0; i < palabra.length(); i++) {
                        palabra[i] = tolower(palabra[i]);
                    }


                    auto it = listado.find(palabra);

                    if (it == listado.cend() || listado[palabra].back() != h) {
                        listado[palabra].push_back(h);
                    }
                }
            }
        }
```

```
        auto itm = listado.cbegin();

        while (itm != listado.cend()) {

            cout << itm->first;

            deque<int> mostrar = itm->second;

            while (!mostrar.empty()) {

                cout << " " << mostrar.front();
                mostrar.pop_front();
            }

            cout << endl;
            itm++;
        }

        cout << "---" << endl;


        return true;
    }
}

int main() {
# ifndef DOMJUDGE
    std::ifstream in("datos.txt");
    auto cinbuf = std::cin.rdbuf(in.rdbuf()); // save old buf and redirect std :: cin to
casos .txt
# endif

    while (resuelveCaso());

# ifndef DOMJUDGE // para dejar todo como estaba al principio
    std::cin.rdbuf(cinbuf);
    system(" PAUSE ");
# endif
    return 0;
}
```