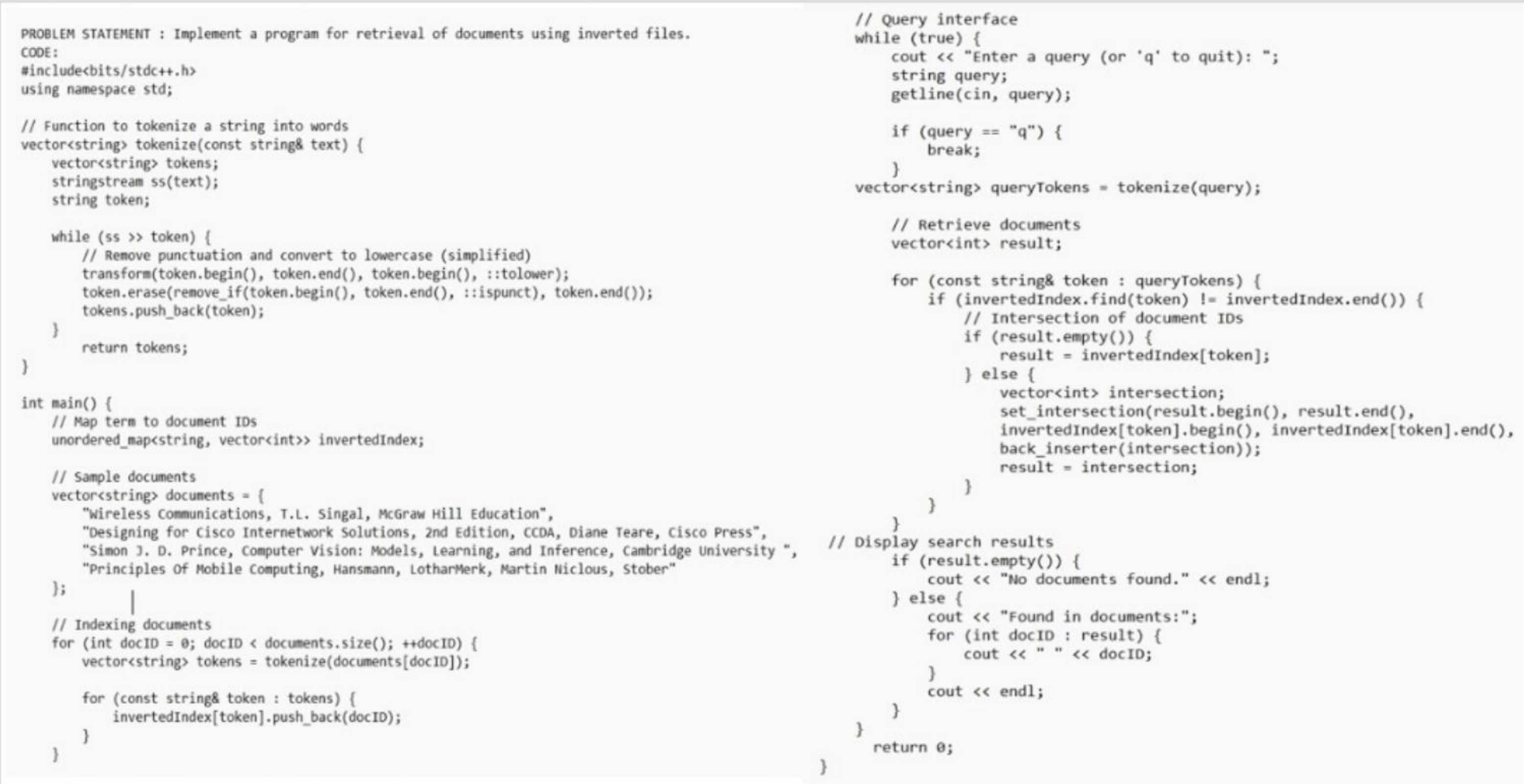
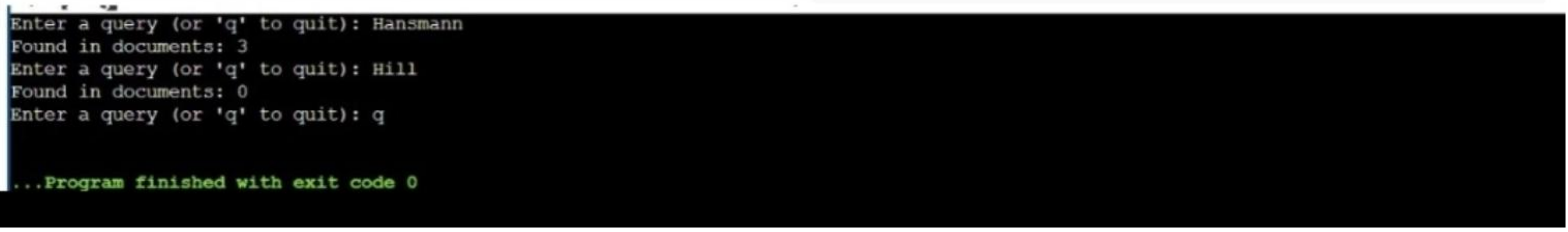


PROBLEM STATEMENT : Implement a program for retrieval of documents using inverted files.

CODE:

```
#include<bits/stdc++.h>
using namespace std;

// Function to tokenize a string into words
vector<string> tokenize(const string& text) {
    vector<string> tokens;
    stringstream ss(text);
    string token;

    while (ss >> token) {
        // Remove punctuation and convert to lowercase (simplified)
        transform(token.begin(), token.end(), token.begin(), ::tolower);
        token.erase(remove_if(token.begin(), token.end(), ::ispunct), token.end());
        tokens.push_back(token);
    }
        return tokens;
}

int main() {
    // Map term to document IDs
    unordered_map<string, vector<int>> invertedIndex;

    // Sample documents
    vector<string> documents = {
        "Wireless Communications, T.L. Singal, McGraw Hill Education",
        "Designing for Cisco Internetwork Solutions, 2nd Edition, CCDA, Diane Teare, Cisco Press",
        "Simon J. D. Prince, Computer Vision: Models, Learning, and Inference, Cambridge University ",
        "Principles Of Mobile Computing, Hansmann, LotharMerk, Martin Niclous, Stober"
    };

    // Indexing documents
    for (int docID = 0; docID < documents.size(); ++docID) {
        vector<string> tokens = tokenize(documents[docID]);

        for (const string& token : tokens) {
            invertedIndex[token].push_back(docID);
        }
    }
    // Query interface
    while (true) {
        cout << "Enter a query (or 'q' to quit): ";
        string query;
        getline(cin, query);

        if (query == "q") {
            break;
        }
    vector<string> queryTokens = tokenize(query);

        // Retrieve documents
        vector<int> result;

        for (const string& token : queryTokens) {
            if (invertedIndex.find(token) != invertedIndex.end()) {
                // Intersection of document IDs
                if (result.empty()) {
                    result = invertedIndex[token];
                } else {
                    vector<int> intersection;
                    set_intersection(result.begin(), result.end(),
                    invertedIndex[token].begin(), invertedIndex[token].end(),
                    back_inserter(intersection));
                    result = intersection;
                }
            }
        }
// Display search results
        if (result.empty()) {
            cout << "No documents found." << endl;
        } else {
            cout << "Found in documents:";
            for (int docID : result) {
                cout << " " << docID;
            }
            cout << endl;
        }
    }
      return 0;
}
```



```
Enter a query (or 'q' to quit): Hansmann
Found in documents: 3
Enter a query (or 'q' to quit): Hill
Found in documents: 0
Enter a query (or 'q' to quit): q


...Program finished with exit code 0
```