```cpp
/*
    Time complexity: O(V + E)
    Space complexity: O(V^2)

    where V is the number of vertices in the input graph and
    E is the number of edges in the input graph
*/

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

vector<int>* getDFSPathHelper(bool** graph, int v, int start, int end, bool* visited) {
    if (start == end) {
        vector<int>* output = new vector<int>();
        output->push_back(end);
        return output;
    }

    visited[start] = true;

    for (int i = 0; i < v; ++i) {
        if (graph[start][i] && !visited[i]) {
            vector<int>* smallOutput = getDFSPathHelper(graph, v, i, end, visited);
            if (smallOutput != NULL) {
                smallOutput->push_back(start);
                return smallOutput;
            }
        }
    }

    return NULL;
}

vector<int>* getDFSPath(bool** graph, int v, int start, int end) {
    bool* visited = new bool[v];

    for (int i = 0; i < v; i++) {
        visited[i] = false;
    }

    vector<int>* output = getDFSPathHelper(graph, v, start, end, visited);

    delete[] visited;

    return output;
}

int main() {
    int v, e;
    cin >> v >> e;

    bool** graph = new bool*[v];

    for (int i = 0; i < v; ++i) {
        graph[i] = new bool[v]();
    }

    for (int i = 0, a, b; i < e; ++i) {
        cin >> a >> b;
        graph[a][b] = true;
        graph[b][a] = true;
    }

    int start, end;
    cin >> start >> end;

    vector<int>* output = getDFSPath(graph, v, start, end);
    if (output != NULL) {
        for (int i = 0; i < output->size(); ++i) {
            cout << output->at(i) << " ";
        }
        delete output;
    }

    for (int i = 0; i < v; ++i) {
        delete[] graph[i];
    }

    delete[] graph;
}
```