```
/*
    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>
using namespace std;

bool hasPathHelper(bool** graph, int v, int start, int end, bool* visited) {
    if (start == end) {
        return true;
    }

    visited[start] = true;

    for (int i = 0; i < v; ++i) {
        if (graph[start][i] && !visited[i]) {
            if (hasPathHelper(graph, v, i, end, visited)) {
                return true;
            }
        }
    }

    return false;
}

bool hasPath(bool** graph, int v, int start, int end) {
    bool* visited = new bool[v]();

    return hasPathHelper(graph, v, start, end, visited);

    delete[] visited;
}

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;

    cout << (hasPath(graph, v, start, end) ? "true" : "false");

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

    delete[] graph;
}
```