```
/*
    Time complexity: O(V^3)
    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;

int getCycles(bool** graph, int v) {
    int cycleCount = 0;

    for (int i = 0; i < v - 2; ++i) {
        for (int j = i + 1; j < v - 1; ++j) {
            for (int k = j + 1; k < v; ++k) {
                if (graph[i][j] && graph[j][k] && graph[k][i]) {
                    ++cycleCount;
                }
            }
        }
    }

    return cycleCount;
}

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;
    }

    cout << getCycles(graph, v);

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

```
    delete[] graph;
}
```