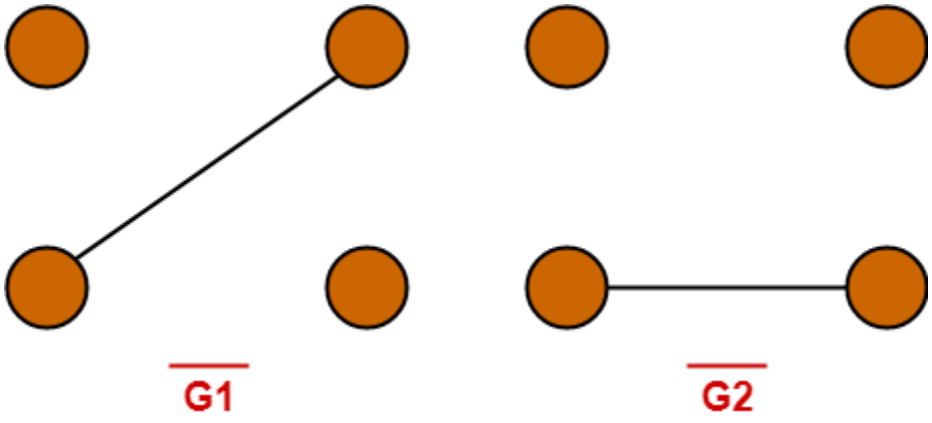
Date: 17/1/2024
Roll No. and Name: 22BCE510 (Aarshit Jolapara)
Course Code and Name: 2CSDE56 – Graph Theory
Practical No.: 2
AIM: Write a program to check whether two graphs are isomorphic to each other or not.

```cpp
#include<bits/stdc++.h>
#include "../g.h" // file containing graph creation and printing functions
using namespace std;

bool isIsomorphic(vector<vector<int>>& graph1, vector<vector<int>>& graph2) {
    int n1 = graph1.size();
    int n2 = graph2.size();

    if (n1 != n2) {
        return false;
    }

    vector<int> perm(n1);
    for (int i = 0; i < n1; i++) {
        perm[i] = i;
    }

    do {
        bool isMappingValid = true;

        for (int i = 0; i < n1; i++) {
            int v1 = perm[i];
            int v2 = i;

            if (graph1[v1].size() != graph2[v2].size()) {
                isMappingValid = false;
                break;
            }

            for (int j = 0; j < graph1[v1].size(); j++) {
                int u1 = graph1[v1][j];
                int u2 = graph2[v2][j];

                if (perm[u1] != u2) {
                    isMappingValid = false;
                    break;
```

```
                }
            }

            if (!isMappingValid) {
                break;
            }
        }

        if (isMappingValid) {
            return true;
        }
    } while (next_permutation(perm.begin(), perm.end()));

    return false;
}


int main() {
    cout << "Enter the number of vertices and edges in graph 1: ";
    int n1, e1;
    cin >> n1 >> e1;
    vector<vector<int>> graph1 = createGraph(n1, e1);

    cout << "Enter the number of vertices and edges in graph 2: ";
    int n2, e2;
    cin >> n2 >> e2;
    vector<vector<int>> graph2 = createGraph(n2, e2);

    cout << "Graph 1: " << endl;
    print(graph1);

    cout << "Graph 2: " << endl;
    print(graph2);

    cout << (isIsomorphic(graph1, graph2) ? "Graphs are isomorphic" : "Graphs
are not isomorphic") << endl;

    return 0;
}
```

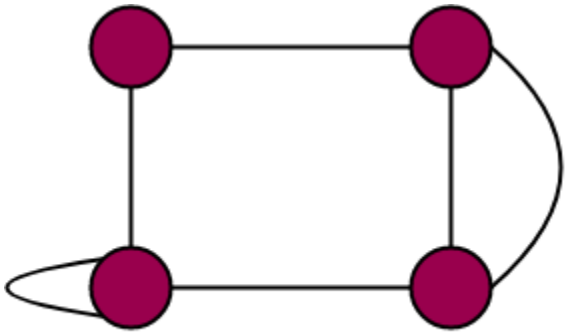
Input / Output:

Example 1:

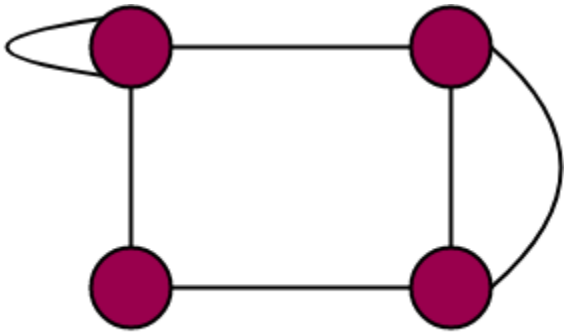



```
Enter the number of vertices and edges in graph 1: 4 1
1 2
Enter the number of vertices and edges in graph 2: 4 1
2 3
Graph 1:
Adjacency List:
1: 2
2: 1
3:
4:

Graph 2:
Adjacency List:
1:
2: 3
3: 2
4:

Graphs are isomorphic
```

Example 2 :



G1



G2

```
Enter the number of vertices and edges in graph 1: 4 6
0 1
1 2
2 3
3 0
1 2
3 3
Enter the number of vertices and edges in graph 2: 4 6
0 1
1 2
2 3
3 0
1 2
0 0
Graph 1:
Adjacency List:
1: 0 2 2
2: 1 3 1
3: 2 0 3 3
4:

Graph 2:
Adjacency List:
1: 0 2 2
2: 1 3 1
3: 2 0
4:

Graphs are not isomorphic
```