No of provinces in C++

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

class Solution {
  private:
    // dfs traversal function
    void dfs(int node, vector<int> adjLs[], int vis[]) {
        // mark the more as visited
        vis[node] = 1;
        for(auto it: adjLs[node]) {
            if(!vis[it]) {
                dfs(it, adjLs, vis);
            }
        }
    }
  public:
    int numProvinces(vector<vector<int>> adj, int V) {
        vector<int> adjLs[V];

        // to change adjacency matrix to list
        for(int i = 0;i<V;i++) {
            for(int j = 0;j<V;j++) {
                // self nodes are not considered
                if(adj[i][j] == 1 && i != j) {
                    adjLs[i].push_back(j);
                    adjLs[j].push_back(i);
                }
            }
        }
        int vis[V] = {0};
        int cnt = 0;
        for(int i = 0;i<V;i++) {
            // if the node is not visited
            if(!vis[i]) {
                // counter to count the number of provinces
                cnt++;
               dfs(i, adjLs, vis);
            }
        }
        return cnt;

    }
};

int main() {

    vector<vector<int>> adj
    {
        {1, 0, 1},
        {0, 1, 0},
        {1, 0, 1}
    };

    Solution ob;
    cout << ob.numProvinces(adj,3) << endl;

    return 0;
}
```

Input:

```
adj = {
  {1, 0, 1},
  {0, 1, 0},
  {1, 0, 1}
};
V = 3
```

✅ Adjacency Matrix ➡ List Conversion:

i	j	adj[i][j]	i != j	Action	adjLs
0	0	1	❌	skip	
0	1	0	✅	skip	
0	2	1	✅	add edge 0–2 and 2–0	0→[2], 2→[0]
1	0	0	✅	skip	
1	1	1	❌	skip	
1	2	0	✅	skip	
2	0	1	✅	already added	
2	1	0	✅	skip	
2	2	1	❌	skip	

🔧 Final Adjacency List:

0 → [2]
1 → []
2 → [0]

🚀 DFS + Province Counting

i	vis[i]	Action	DFS Called	Updated vis	cnt
0	0	Not visited → DFS(0)	✓	[1, 0, 1]	1
1	0	Not visited → DFS(1)	✓	[1, 1, 1]	2
2	1	Already visited	✗	-	-

🔁 DFS Traversal Details

◆ **DFS(0)**

node	vis[node]	Neighbors	Action	vis
0	0 → 1	2	DFS(2)	[1, 0, 0]
2	0 → 1	0	Already vis	[1, 0, 1]

◆ **DFS(1)**

node	vis[node]	Neighbors	Action	vis
1	0 → 1	none	Done	[1, 1, 1]

📜 **Final Result**

Variable	Value
cnt	2 (Answer)
vis	[1, 1, 1]

🟩 **Output: 2 provinces**

Output:-
2