**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：10397: Connect the Campus**

**解題者：陳彥維**

**解題日期：2025年4月17日**

**題意：**

給定 N 個(1 ≤ N ≤ 750) 笛卡兒座標的點 (x, y)，再給定 M 個(0 ≤ M ≤ 1000) 連接這些點的邊，要求加上新的邊使每個點都有邊連接到，並輸出新的邊的總長度。

**題意範例：**



**解法：**

 Kruskal’s algorithm for finding MST(最小生成樹)

**解法範例：**



**討論：**

 (1) Kruskal v.s. Prim? Kruskal通常用於稀疏圖，而Prim則處理稠密圖

 (2) 使用 Union-Find 來避免形成環圈

 (3) N = 750時，邊的數量最大有$\left(\genfrac{}{}{0pt}{}{750}{2}\right)$ ≈ 280,000條，可以考慮使用「剪枝」加速。

**程式：**

#include <iostream>

#include <algorithm>

#include <cmath>

#include <vector>

#include <iomanip>

using namespace std;

struct Edge{

    int u, v;

    double w;

    bool operator<(const Edge& other){

        return w < other.w;

    }

};

int find(vector<int>& parent, int x){

    return (x == parent[x]) ? x : parent[x] = find(parent, parent[x]);

}

void unite(vector<int>& parent, int x, int y){

    parent[find(parent, x)] = find(parent, y);

}

int main(){

    int n;

    while(cin >> n){

        vector<pair<int, int>> points(n+1);

        for(int i = 1; i <= n; i++){

            cin >> points[i].first >> points[i].second;

        }

        vector<Edge> maps;

        for(int i = 1; i <= n; i++){

            for(int j = i + 1; j <= n; j++){

                double dist = hypot(points[i].first - points[j].first, points[i].second - points[j].second);

                maps.push\_back({i, j, dist});

            }

        }

        vector<int> parent(n+1);

        for(int i = 1; i <= n; i++){

            parent[i] = i;

        }

        int m;

        cin >> m;

        for(int i = 0; i < m; i++){

            int u, v;

            cin >> u >> v;

            unite(parent, u, v);

        }

        sort(maps.begin(), maps.end());

        double result = 0;

        for(auto e : maps){

            if(find(parent, e.u) != find(parent, e.v)){

                unite(parent, e.u, e.v);

                result += e.w;

            }

        }

        cout << fixed << setprecision(2) << result << '\n';

    }

}