程式設計(⼆)

姓名: 蔡秉岑學號: 411446510

科系: 電機⼯程學系電機資訊組

# 題⽬: [c131. 00615 - Is It A Tree?](https://zerojudge.tw/ShowProblem?problemid=c131)

出處: UVa615

編譯器: Visual Studio Code

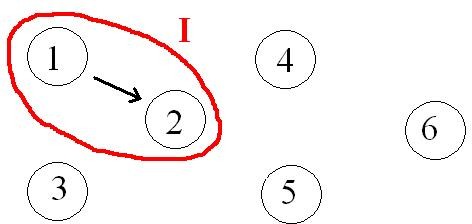
程式語⾔: C++

# 解題思路

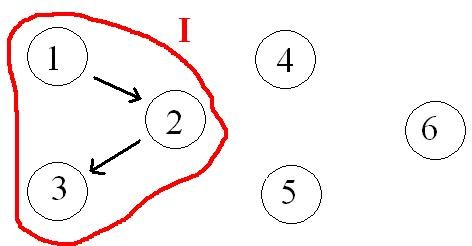
若⼀圖形為tree的結構，表⽰此圖形中所有的點都互相連通且必無cycle存在，因此判斷cycle的存在是本題最重要的關鍵。⼀個判斷有無cycle存在的簡單⽅法是利⽤集合的概念，把⽬前已知的連線關係歸類到同⼀個集合中，我們設計⼀個範例來逐步解釋這個過程:

**Input** 1 2 2 3 4 6 2 5 2 4 5 6 0 0

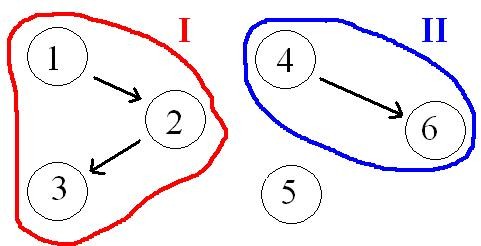
⾸先我們假設所有的點都不在任何⼀個集合中。 第⼀個輸入：1 2，代表1指向2，所以我們把1與2歸類成集合I。



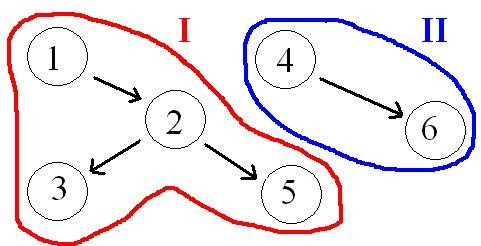
第⼆個輸入：2 3，表⽰2指向3，所以3也屬於集合I。



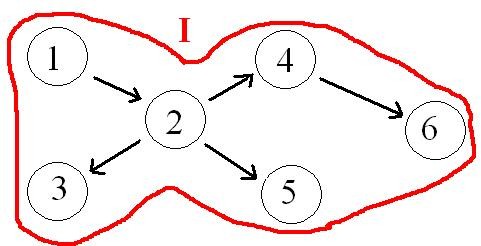
第三個輸入：4 6，表⽰4指向6，所以4與6屬於另⼀個集合II



下⼀個輸入：2 5，所以5也屬於集合I。



下⼀個輸入：2 4，所以包含4的這個集合內的全部元素都應該歸類到集合I。



總結: 判斷是否沒有任何節點，若沒有任何節點也是 Tree，否則隨便挑⼀個沒有被接入的節點開始 DFS 判斷是否有環，最後判斷是否所有節點都有被遍歷到即可。

複雜度: $O(N)$

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

int u, v;

unordered\_map<int, vector<int>> G; unordered\_map<int, int> inDegree; unordered\_set<int> visited;

void init(){

G.clear();

inDegree.clear(); visited.clear();

}

void readGraph(){

while (!(!u && !v)){

G[u].push\_back(v); inDegree[u] += 0;

++inDegree[v]; cin >> u >> v;

}

}

// 如果有環則回傳 false bool dfs(int u){

visited.insert(u);

for (auto& v : G[u]){

if (visited.count(v)) return false; // 有環

if (!dfs(v)) return false;

}

return true;

}

bool solve(){

// 沒有節點也是樹

if (inDegree.empty()) return true;

for (auto& [u, deg] : inDegree){ if (deg) continue;

if (!dfs(u)) return false; break;

}

// 檢查是否有節點還沒有遍歷過

for (auto& [u, \_] : inDegree)

if (!visited.count(u)) return false; return true;

}

int main(){

int Case = 0;

while (cin >> u >> v && !(u == -1 && v == -1)){ init();

readGraph();

cout << "Case " << ++Case << " is "; if (solve()) cout << "a tree.\n";

else cout << "not a tree.\n";

}

}

# 解題⼼得

當初在解這題的時候，花費了不少時間在複習DFS的⽤法以及Set還有map的運⽤，也開始回想老師上課講map 映射的⽤法，我透過參考了 **"演算法筆記 Set"** 裡⾯的概念成功解決了這題。上⼀次的我做的是貪婪演算法，比起這次難度其實低很多，也證明了我在競程能⼒上⾯的成⻑，還是希望之後有老師的課程能夠直接上機考試以及教課，這樣的效率我認為是比較好的，不然只空聽老師講不⾃⼰實做看看也很難領悟到其中的精隨。



# 參考文獻

[冰塊的UVa解題冷藏庫](https://kos74185foracm.blogspot.com/2011/08/615-is-it-tree.html)

[深度優先搜尋(DFS)和廣度優先搜尋(BFS)演算法，實⽤的節點搜尋法](https://magiclen.org/dfs-bfs/)[演算法筆記 Set](https://web.ntnu.edu.tw/~algo/Set.html#5)

[C++11 新的容器：array、沒排序的 set 與 map](https://viml.nchc.org.tw/containers-in-cpp-11/)