Sir method

-----------------------------------------------------------

#include<bits/stdc++.h>

using namespace std;

unordered\_map<string,string> graph;

vector<string> solve(){

unordered\_map<string,bool> vis;

vector<string> res;

for(auto it: graph){

vis[it.second] = false;

if(vis.find(it.first)==vis.end()){

vis[it.first] = true;

}

}

string start;

for(auto it: vis){

if(it.second==true){

start = it.first;

break;

}

}

res.push\_back(start);

while(true){

if(graph.find(start)!=graph.end()){

res.push\_back(graph[start]);

start = graph[start];

}

else{

break;

}

}

return res;

}

int main(){

int n;

cin>>n;

for(int i=0;i<n;i++){

string s1,s2;

cin>>s1>>s2;

graph[s1] = s2;

}

vector<string> res = solve();

for(int i=0;i<res.size();i++){

if(i==res.size()-1){

cout<<res[i]<<".";

}

else{

cout<<res[i]<<" -> ";

}

}

}

My method

--------------------------------------------------------

#include<bits/stdc++.h>

using namespace std;

unordered\_map<string,string> graph;

int main(){

int n;

cin>>n;

for(int i=0;i<n;i++){

string s1,s2;

cin>>s1>>s2;

graph[s1] = s2;

}

unordered\_map<string,int> ind; // indegree

for(auto it: graph){

ind[it.first]+= 0;

ind[it.second]++;

}

queue<string> q;

for(auto it: ind){

if(ind[it.first]==0){

q.push(it.first);

}

}

vector<string> res;

while(q.size()>0){

string node = q.front();

q.pop();

res.push\_back(node);

if(graph.find(node)!=graph.end()){

string it = graph[node];

ind[it]--;

if(ind[it]==0){

q.push(it);

}

}

}

for(int i=0;i<res.size();i++){

if(i==res.size()-1){

cout<<res[i]<<".";

}

else{

cout<<res[i]<<" -> ";

}

}

}