TRUCK TOUR :

IN C++ :-

**#include <bits/stdc++.h>**

**using namespace std;**

**string ltrim(const string &);**

**string rtrim(const string &);**

**vector<string> split(const string &);**

**int truckTour(vector<vector<int>> pp) {**

**int res =0,ind,max1,tot=0;**

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

**tot += pp[i][0] - pp[i][1];**

**if(tot<0) {**

**ind = i+1;**

**tot =0;**

**}**

**}**

**return ind;**

**}**

**int main()**

**{**

**ofstream fout(getenv("OUTPUT\_PATH"));**

**string n\_temp;**

**getline(cin, n\_temp);**

**int n = stoi(ltrim(rtrim(n\_temp)));**

**vector<vector<int>> petrolpumps(n);**

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

**petrolpumps[i].resize(2);**

**string petrolpumps\_row\_temp\_temp;**

**getline(cin, petrolpumps\_row\_temp\_temp);**

**vector<string> petrolpumps\_row\_temp = split(rtrim(petrolpumps\_row\_temp\_temp));**

**for (int j = 0; j < 2; j++) {**

**int petrolpumps\_row\_item = stoi(petrolpumps\_row\_temp[j]);**

**petrolpumps[i][j] = petrolpumps\_row\_item;**

**}**

**}**

**int result = truckTour(petrolpumps);**

**fout << result << "\n";**

**fout.close();**

**return 0;**

**}**

**string ltrim(const string &str) {**

**string s(str);**

**s.erase(**

**s.begin(),**

**find\_if(s.begin(), s.end(), not1(ptr\_fun<int, int>(isspace)))**

**);**

**return s;**

**}**

**string rtrim(const string &str) {**

**string s(str);**

**s.erase(**

**find\_if(s.rbegin(), s.rend(), not1(ptr\_fun<int, int>(isspace))).base(),**

**s.end()**

**);**

**return s;**

**}**

**vector<string> split(const string &str) {**

**vector<string> tokens;**

**string::size\_type start = 0;**

**string::size\_type end = 0;**

**while ((end = str.find(" ", start)) != string::npos) {**

**tokens.push\_back(str.substr(start, end - start));**

**start = end + 1;**

**}**

**tokens.push\_back(str.substr(start));**

**return tokens;**

**}**

**Strongly Connected Digraphs**

**In python3 :-**

**mod = 10\*\*9+7**

**MAX = 1001**

**l1,l2,l3=[[0]\*MAX for i in range(MAX)],[0]\*MAX,[0]\*MAX**

**for j in range(MAX):**

**for k in range(j+1):**

**l1[j][k]=1 if k==0 or k==j else (l1[j-1][k-1]+l1[j-1][k])%mod**

**l2[j]=(pow(2, j\*(j-1), mod)-sum(l1[j][i]\*pow(2, (j-1)\*(j-i), mod)\*l2[i] for i in range(1,j)))%mod**

**l3[j]=(l2[j]+sum(l1[j-1][i-1]\*l3[i]\*l2[j-i] for i in range(1,j)))%mod**

**for \_ in range(int(input())):**

**print(l3[int(input())])**

**Cycle Detection:**

**In Python3 :-**

**def has\_cycle(head):**

**s=set()**

**while head is not None:**

**if head in s:**

**return True**

**else:**

**s.add(head)**

**head=head.next**

**return False**