How to solve a classic a strongly connected components problem

CODECHEF visiting friend:

DIFFICULTY:

Medium

PREREQUISITES:

Dfs ,Strongly connected components, topological sort, dynamic programming on Directed acyclic graph.

Problem link:

Editorial:

প্রব্লেমটার কথা খুব সহজ,একটা ডিরেকশনাল গ্রাফ দেয়া আছে,এবং প্রতি টা নোডের জন্য একটা ভ্যালু দেয়া আছে। তোমাকে বের করতে হবে প্রতি নোড হতে যেখানে যেখানে যাওয়া যায় (ঐ নোড সহ) তাদের সমষ্টি বের করতে হবে।

প্রথমেই আসি ব্রুট ফোর্স সমাধানেঃ আমরা কোন নোড হতে কোন কোন নোডে যাওয়া যায় তা ওই নোড হতে DFS চালিয়ে দিলেই বের করতে পারব।

কোড টা হবে অনেকটা এরকমঃ

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

memset(visited,0,sizeof visited);

dfs(i);

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

if(visited[j]==1) ans[i]=ans[i]+cost[j];

}
```

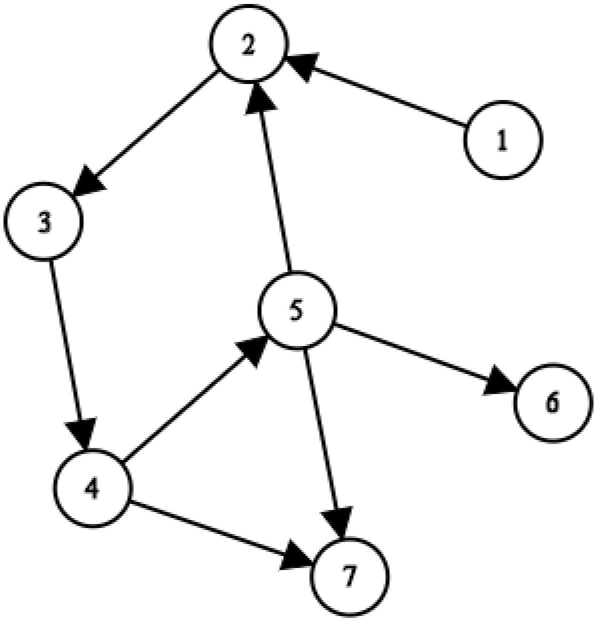
ans অ্যারের মধ্য আমাদের উত্তর রয়েছে।

তাহলে এখানে O(n) কমপ্লেক্সিটির একটা লুপের মাঝে dfs চলছে যার কমপ্লেক্সিটি O(n) এবং আরেকটা লুপ চলেছে যার কমপ্লেক্সিটি O(n), তাহলে মোট কমপ্লেক্সিটি O(n^2). যা বড় টেস্ট কেসের জন্য কখনই এক্সেপ্টেড হবে না।

তাহলে এবার অন্য সলুশনে যাওয়া যাক।

আমরা স্যাম্পল টেস্ট কেসের গ্রাফ টা একে ফেলি বুঝার সুবিধার জন্যঃ

```
7 8
1 2 3 4 5 6 7
1 2
2 3
3 4
4 5
5 2
5 6
5 7
4 7
```



এখন তাহলে এই প্রব্লেম সল্‌ভ করার জন্য আমাদের এই গ্রাফের strongly connected component (SCC) গুলো বের করে ফেলব। SCC বের করার উপায় হল,প্রথমে টপলজিক্যাল সর্ট করে ফেলব dfs চালিয়ে ।আবার সেই সর্টিং অর্ডারে রিভার্স গ্রাফে আবার dfs চালালে,কে কোন কম্পনেন্টে রয়েছে তা বের হয়ে যাবে।

উপরের গ্রাফ হতে পাচ্ছিঃ

Component 1: 1

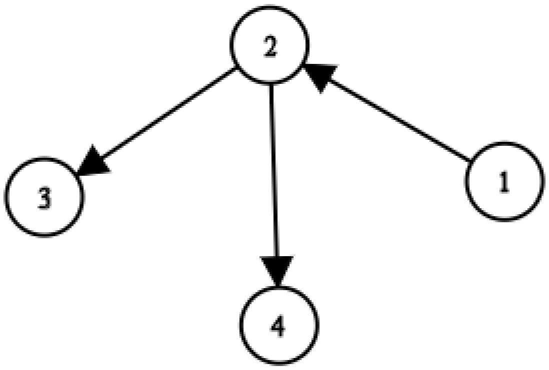
Component 2: 2,3,4,5

Component 3: 6

Component 4: 7

এখন এই এলগোরিদম আমাদের একটা dag রিটার্ন করবে। আমাদের কাজ হবে সেই dag টা বানিয়ে ফেলা।

অনেকটা এমন



এখন যদি এই dag এর টপসর্ট করি ,তাহলে কোন dag নোডের ডানে বা বামে কারা থাকবে আমরা সহজে বের করতে পারব।

তবে সমস্যা হল,যেমন ২ নাম্বার নোডের সাথে ৩ ও ৪ নাম্বার নোড যুক্ত আছে,আমাদের তাহলে বের করতে হবে কোন dag নোড দিয়ে গেলে ম্যাক্সিমাম হবে তা বের করা। এখন আমরা টপসর্টের প্রোপার্টি হতে জানি টপ সর্টের যে শেষে থাকে তার out degree হয় শুন্য। মানে তার কারো উপর ডিপেন্ডেন্সি নেই।

তাহলে আমরা যদি প্রথমে তাদের এন্সার বের করে ফেলি,তাহলে তাদের উপর যাদের ডিপেন্ডেন্সি রয়েছে তাদের গুলা bottom-up approach এ বের করা ফেলা যাবে।

এখন শুধু কোন নোড কোন scc এ রয়েছে জানলেই আমাদের উত্তর বের হয়ে যাবে।

এখানে প্রতিটা সাবটাস্কের কমপ্লেক্সিটি O(n).