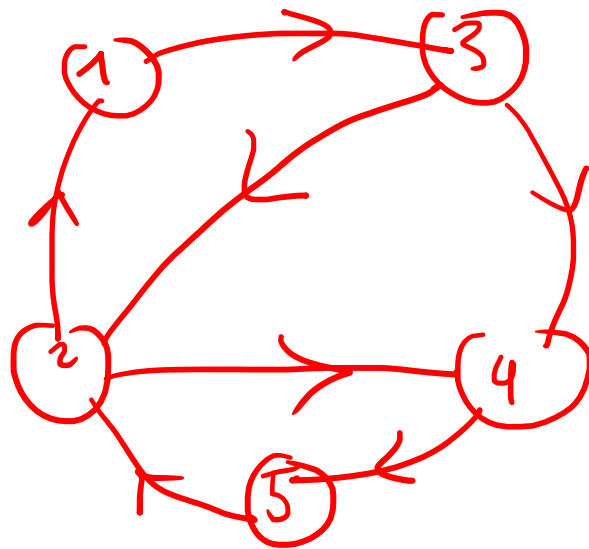
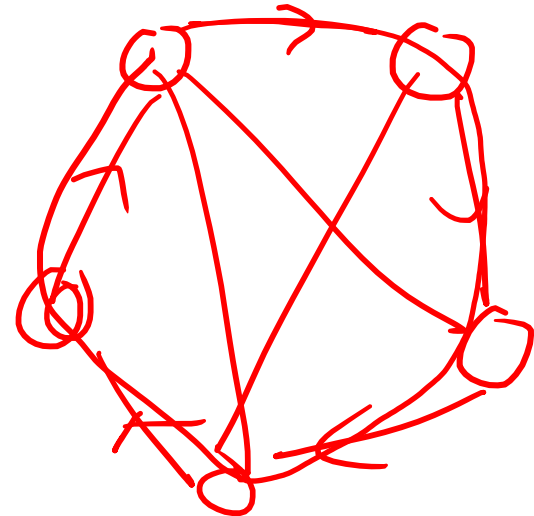


liên thông yếu  
nhưng không phải là  
liên thông mạnh  
(~~liên~~ số thì vô  
hướng này là  
liên thông)



liên thông mạnh  
(cũng là liên thông yếu)

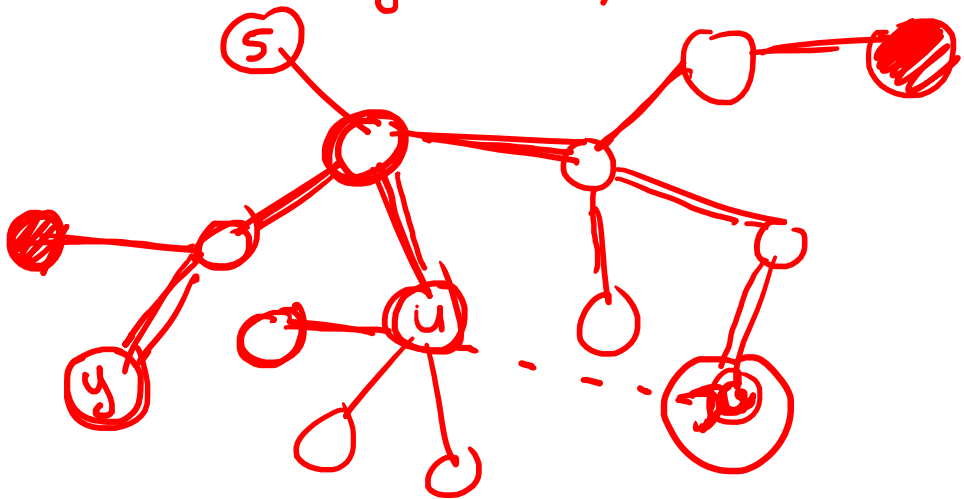
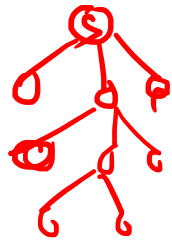


BT: Tìm đường đi dài nhất trên cây

Cây: đồ thị vô hướng, liên thông

Chỗ có chu trình

- Số đỉnh = số cạnh + 1
- Đường đi giữa 2 đỉnh trên cây là duy nhất
- Thêm 1 cạnh nối 2 đỉnh bất kỳ  $u$  và  $v$  của cây  $\Rightarrow$  tạo chu trình



Thuật toán:


- Áp dụng DFS hoặc BFS từ 1 đỉnh  $(s)$  bất kỳ  $\rightarrow$  tìm được đỉnh  $x$  cách xa  $s$  nhất
- Áp dụng DFS hoặc BFS từ đỉnh  $x$  (chọn ở pha 1) ta sẽ tìm được đỉnh  $y$  cách xa  $x$  nhất.
- Đường đi từ  $x$  đến  $y$  chính là đường đi dài nhất trên cây.



~~Sắp xếp TOPO.~~

Tìm thành phần liên thông mạnh  
Strongly Connected Components.

Cho đồ thị có hướng  $G = (V, E)$

Tìm các thành phần liên thông mạnh  
của  $G$ . 

Áp dụng DFS

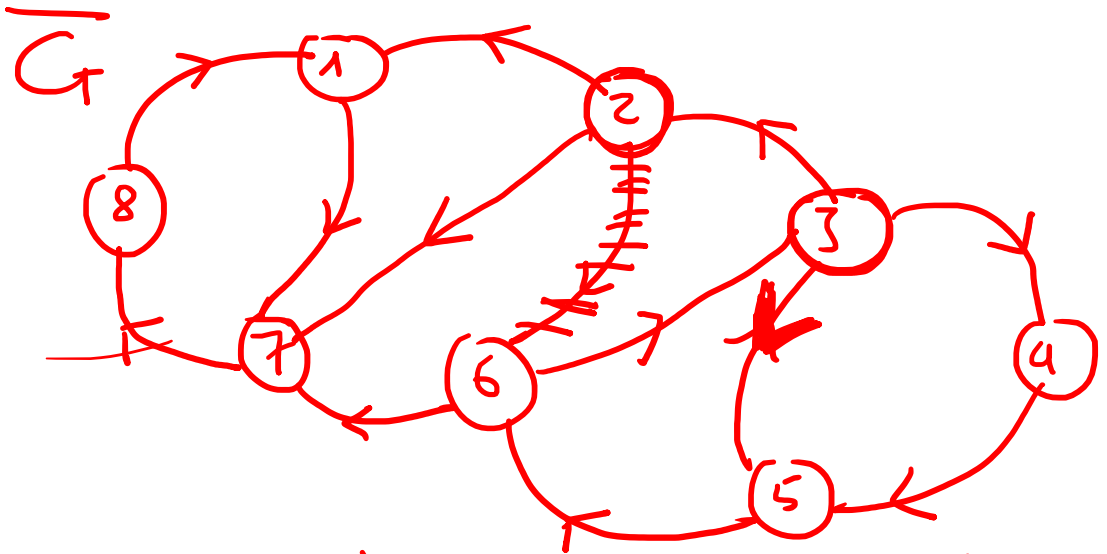
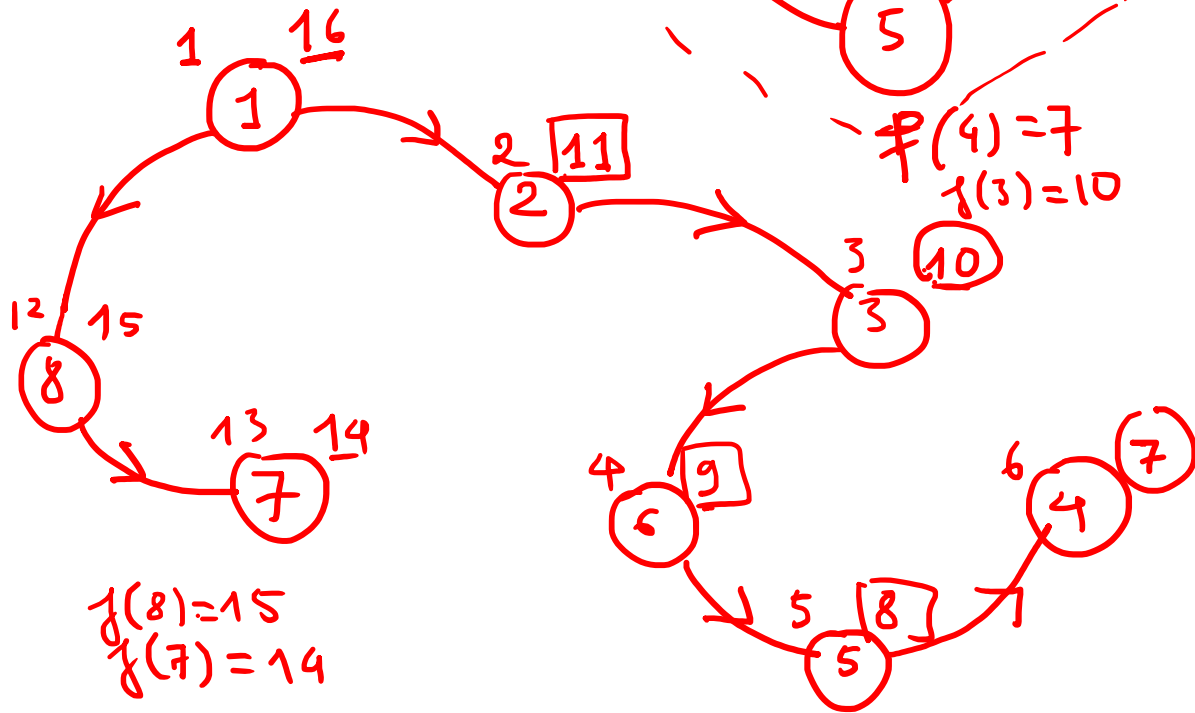
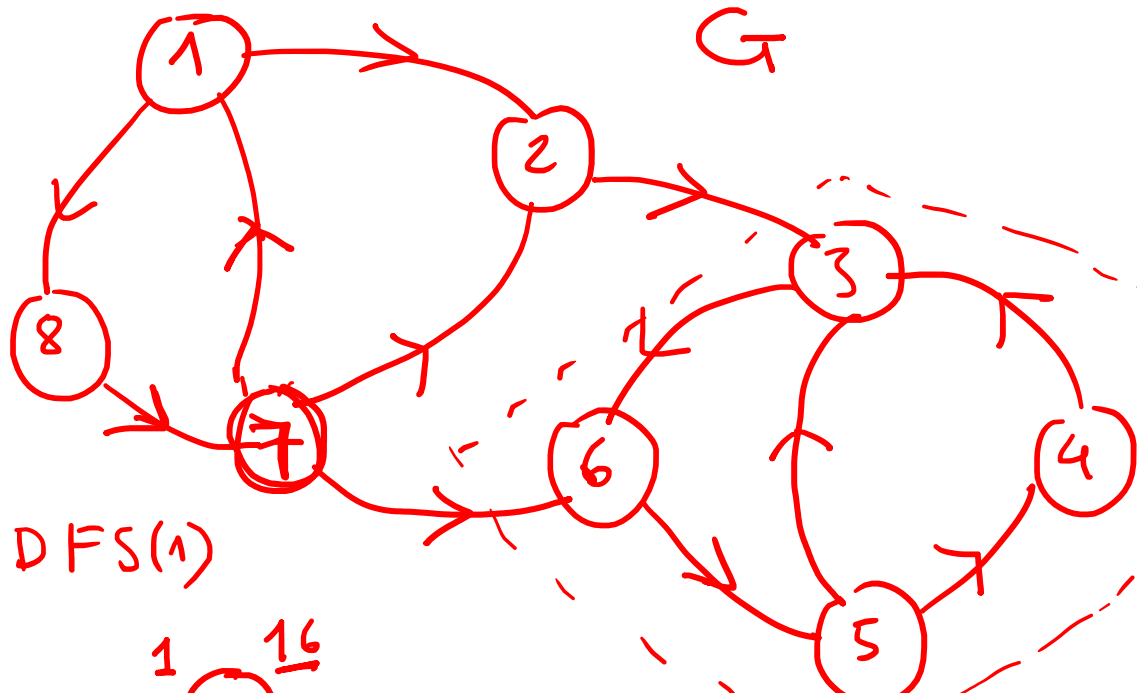
Thuật toán 1: DFS 2 pha,

- DFS để tìm  $f[v]$  là tđ' đỉnh  $v$   
được duyệt xong

- Xây dựng đồ thị bù  $\overline{G} = (V, \overline{E})$   
 $(u, v) \in E \iff (v, u) \in \overline{E}$

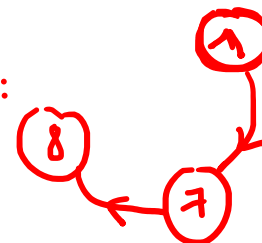
- Chạy DFS trên  $\overline{G}$  theo thứ tự  
các đỉnh: giảm dần của  $f$





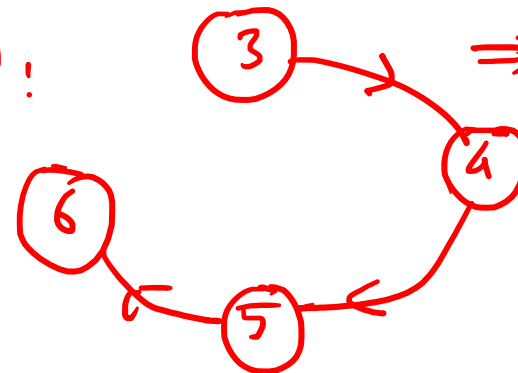
DFS( $\bar{G}$ ): thứ tự đỉnh : giảm dần của  $f$   
SCC[1]: 1, 7, 8

DFS(1):



DFS(2):

DFS(3):



SCC[2]: 2

$\Rightarrow$  SCC[3] = 3, 4, 5, 6

# Thuật toán 2: DFS tìm tập hợp tin

$num[v]$ : thứ tự đỉnh  $v$  được thăm

$low[v]$ : là chỉ  $num\_nhỏ\_nhất$  của 1

đỉnh  $x$  nào đó mà có ứng ngược  
( $u, x$ ) với  $u$  là con cháu của  $v$  trên  
cây DFS

$num[3]=1, num[6]=2$

$num[0]=3, num[1]=4$

$num[2]=5, num[4]=6$

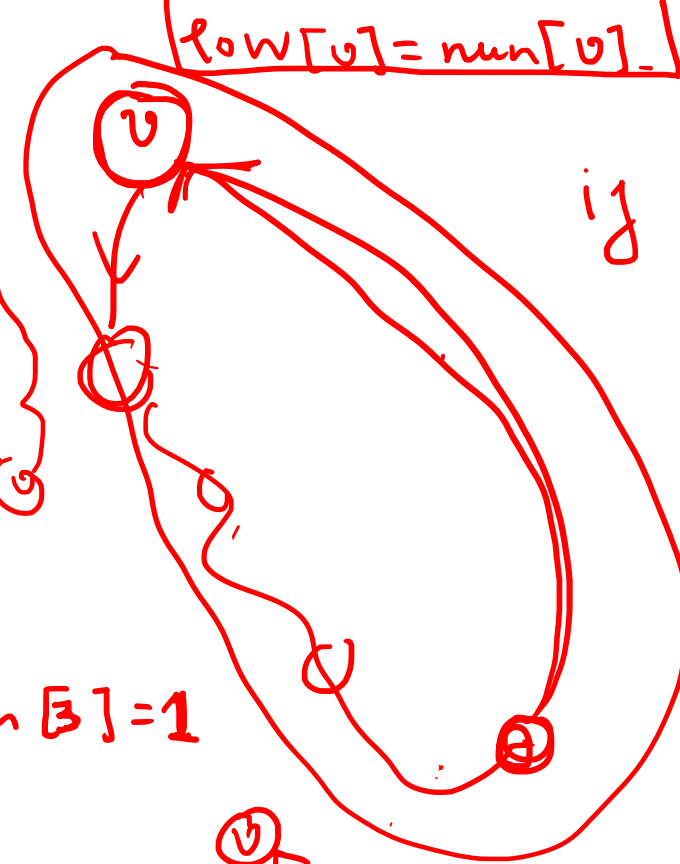
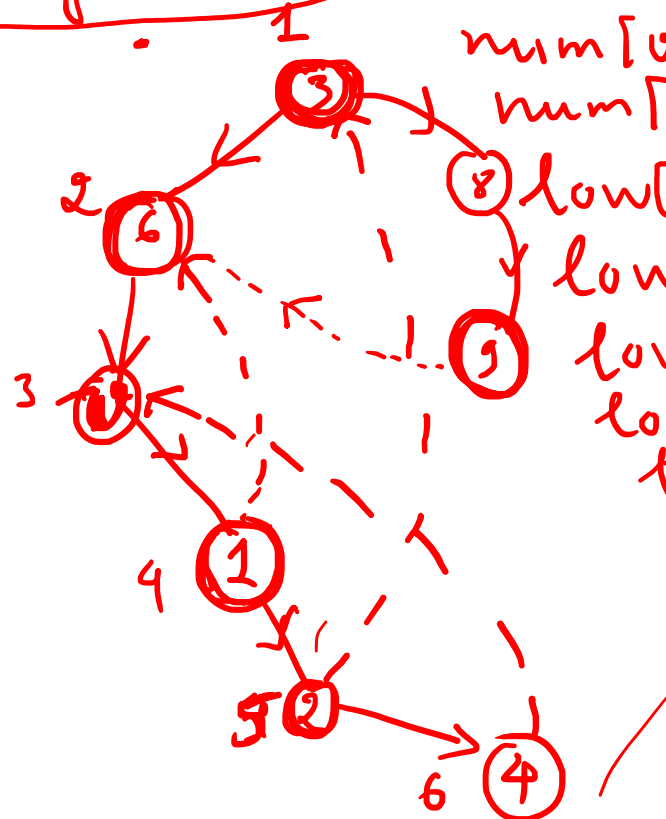
$low[3]=1, low[2]=num[3]=1$

$low[1]=num[3]=1$

$low[4]=num[0]=3$

$low[6]=num[3]=1$

$low[0]=num[3]=1$



if  $low[v] = num[v]$   
thì  $v$  và  
các đỉnh  
con cháu  
của nó  
tạo thành  
1 SCC.



$low[u] = \min(low[u], num[v])$   
(chỉ xét khi  $v$   
còn nằm trên stack)

DFS(u) {

$t = t + 1$ ; num[u] = t; low[u] = t

Stack.push(u); inStack[u] = True;

for  $v \in A[u]$  do { // duyệt các đỉnh kề u

if num[v] > 0 then { // v đã được thăm

if inStack[v] then  $\text{low}[u] = \min(\text{low}[u], \text{num}[v])$  (u,v) là cùng một chu trình

} else { // v chưa được thăm (u,v) là cạnh thường

DFS(v);

$\text{low}[u] = \min(\text{low}[u], \text{low}[v])$

}

}

if low[u] == num[u] then {

pop các đỉnh ra khỏi stack  
và đó là các đỉnh  $\in$  SCC ~~đó~~ mới

}

