

---

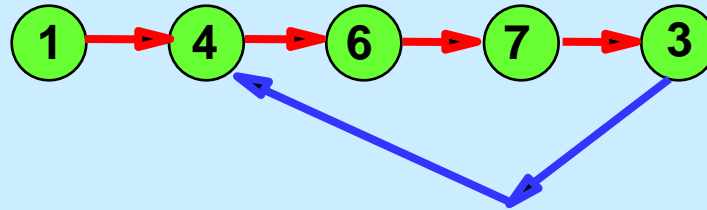
**15.082 and 6.855J**

**Topological Ordering**

# Preliminary to Topological Sorting

---

**LEMMA.** If each node has at least one arc going out, then the first inadmissible arc of a depth first search determines a directed cycle.

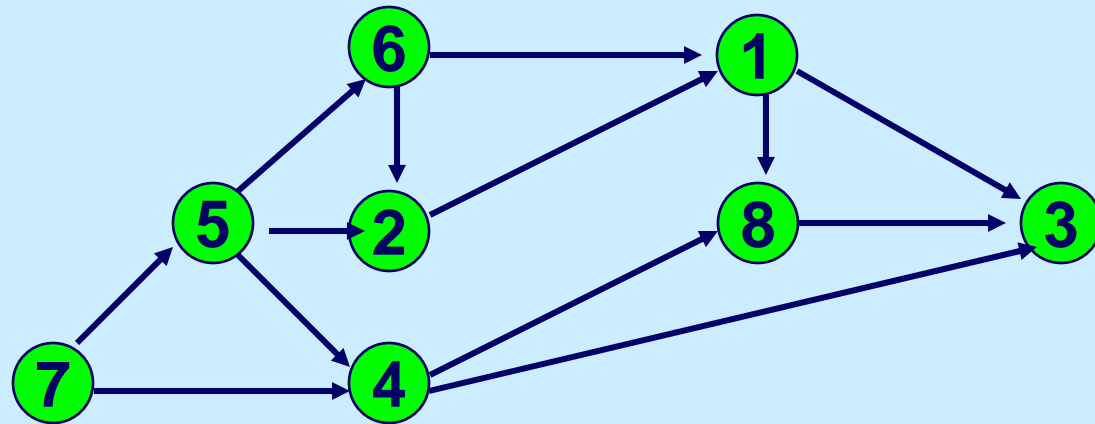


**COROLLARY 1.** If  $G$  has no directed cycle, then there is a node in  $G$  with no arcs going. And there is at least one node in  $G$  with no arcs coming in.

**COROLLARY 2.** If  $G$  has no directed cycle, then one can relabel the nodes so that for each arc  $(i,j)$ ,  $i < j$ .

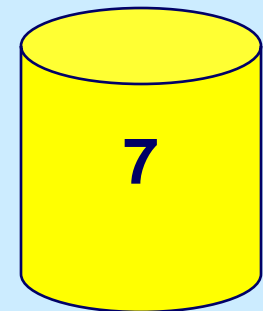
# Initialization

“Next” will be the label of nodes in the topological order.



next 0

Node	1	2	3	4	5	6	7	8
Indegree	2	2	3	2	1	1	0	2



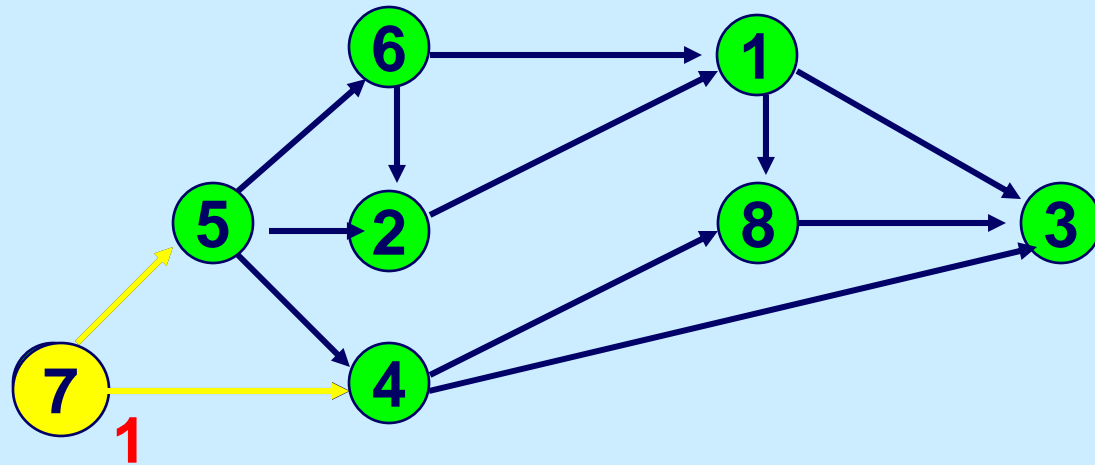
LIST

# Select a node from LIST

next := next +1  
 order(i) := next;

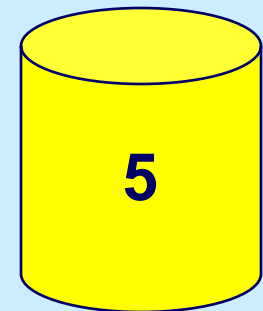
update  
 indegrees

update LIST



next 1

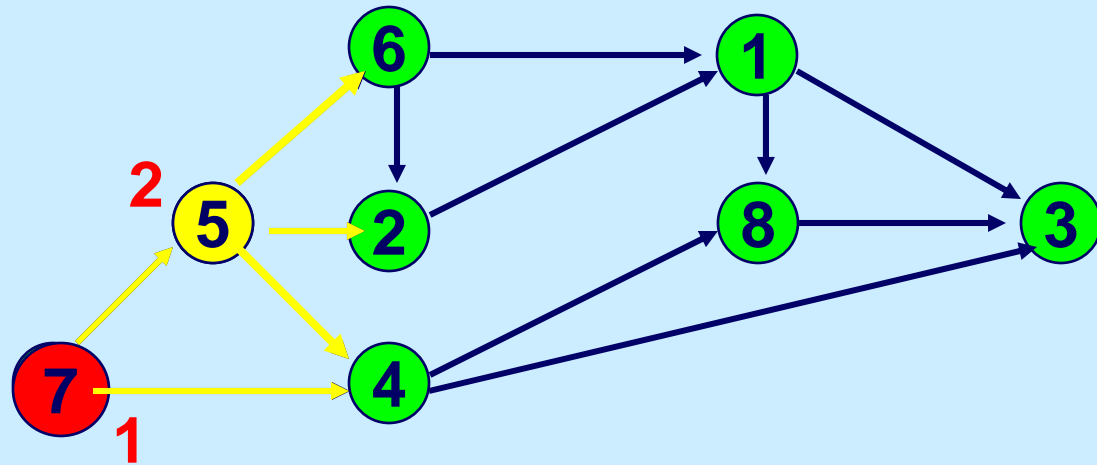
Node	1	2	3	4	5	6	8
Indegree	2	2	3	1	0	1	2



LIST

# Select a node from LIST

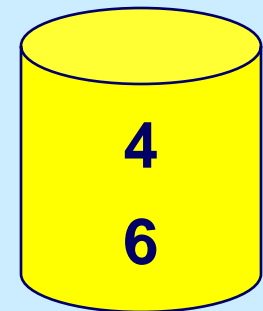
$next := next + 1$   
 $order(i) := next;$   
 update  
 indegrees  
 update LIST



next 2

Node	1	2	3	4	6	8
Indegree	2	1	3	0	0	2

5



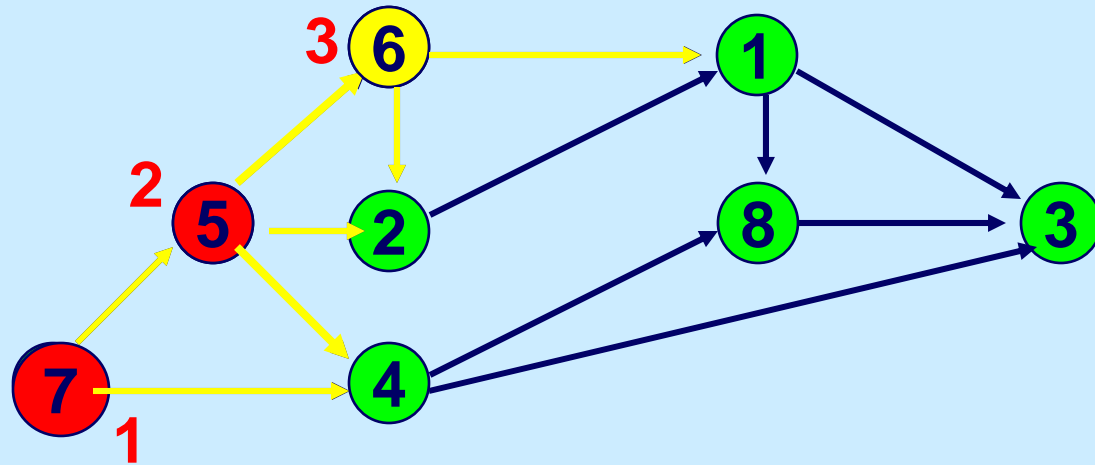
LIST

# Select a node from LIST

next := next +1  
 order(i) := next;

update  
 indegrees

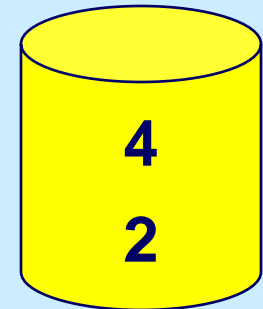
update LIST



next 3

Node	1	2	3	4
Indegree	1	0	3	0

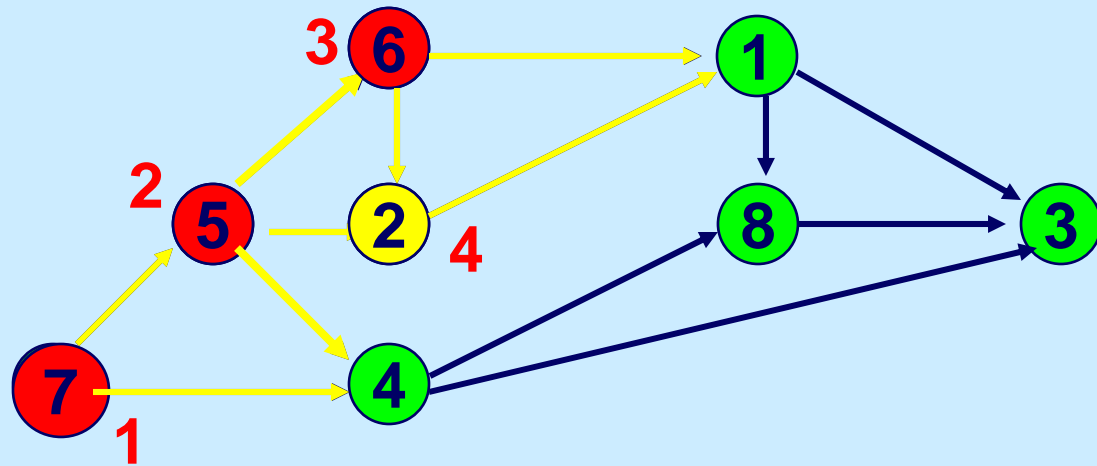
8
2



LIST

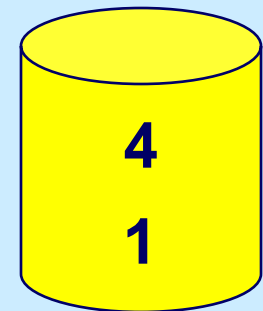
# Select a node from LIST

`next := next + 1`  
`order(i) := next;`  
 update  
 indegrees  
 update LIST



next 4

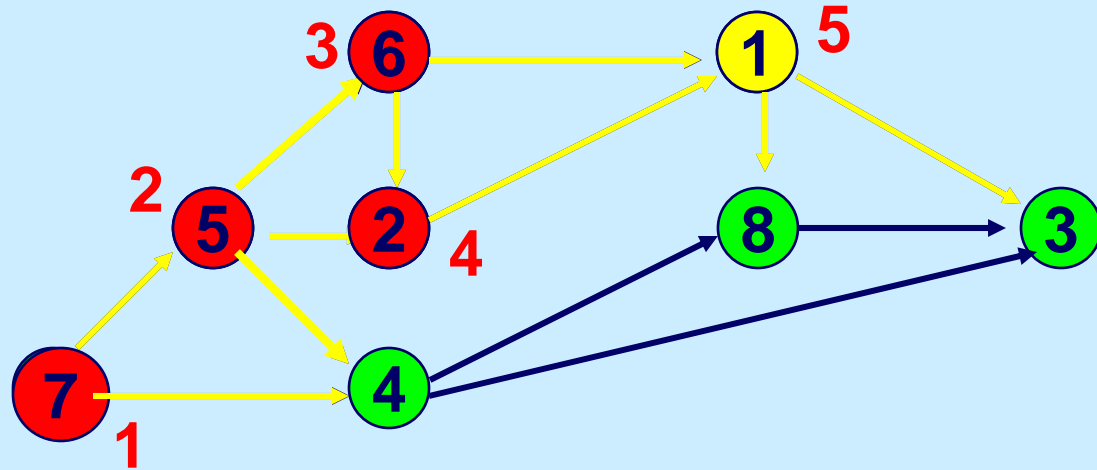
Node	1	3	4	8
Indegree	0	3	0	2



LIST

# Select a node from LIST

$next := next + 1$   
 $order(i) := next;$   
 update  
 indegrees  
 update LIST

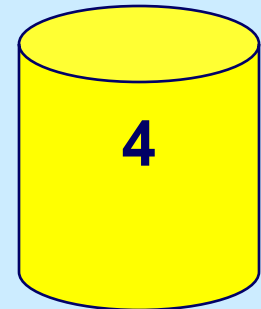


next 5

Node  
Indegree

3	4
2	0

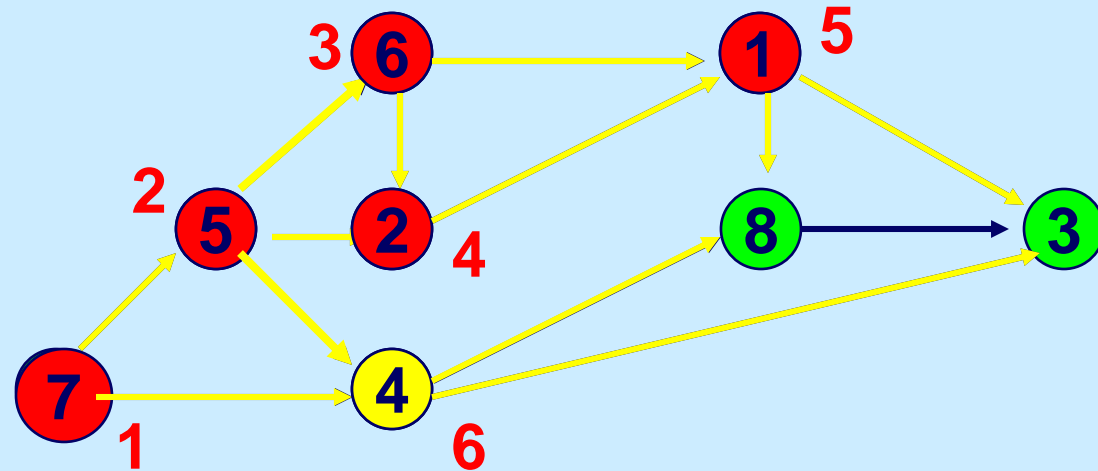
8
1



LIST

# Select a node from LIST

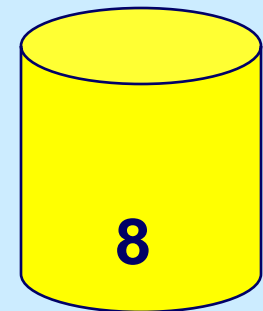
$next := next + 1$   
 $order(i) := next;$   
 update  
 indegrees  
 update LIST



next 6

Node	3
Indegree	1

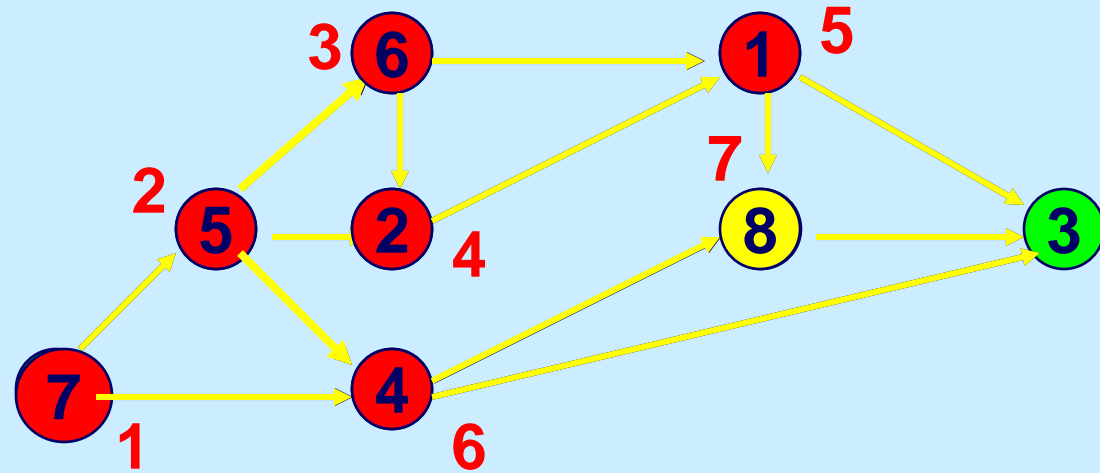
Node	8
Indegree	0



LIST

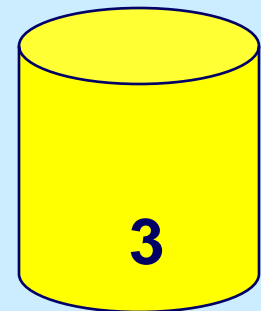
# Select a node from LIST

$next := next + 1$   
 $order(i) := next;$   
 update  
 indegrees  
 update LIST



next 7

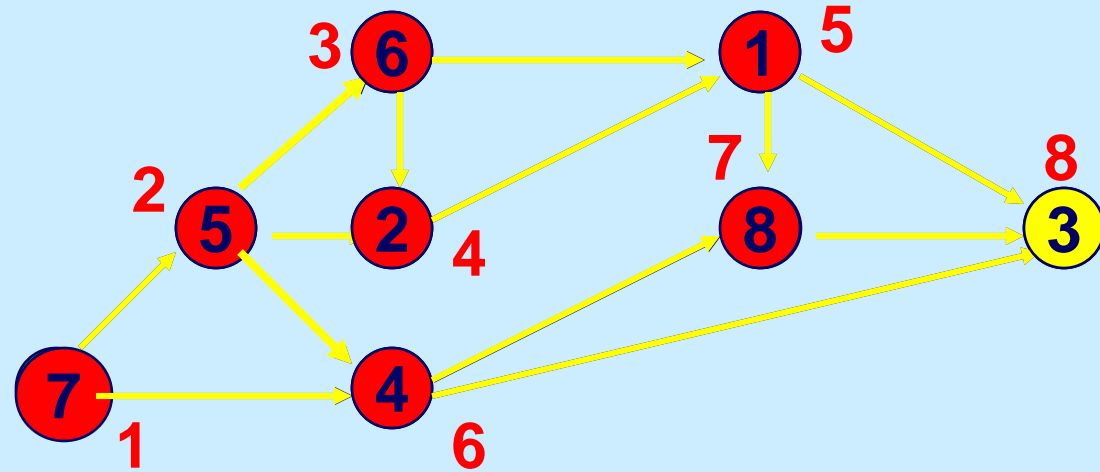
Node	3
Indegree	0



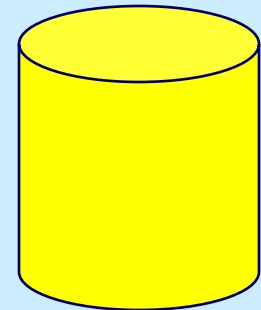
**LIST**

# Select a node from LIST

`next := next +1`  
`order(i) := next;`  
update  
indegrees  
update LIST



next 8



LIST

List is empty.

The algorithm  
terminates with a  
topological order of  
the nodes

Node  
Indegree