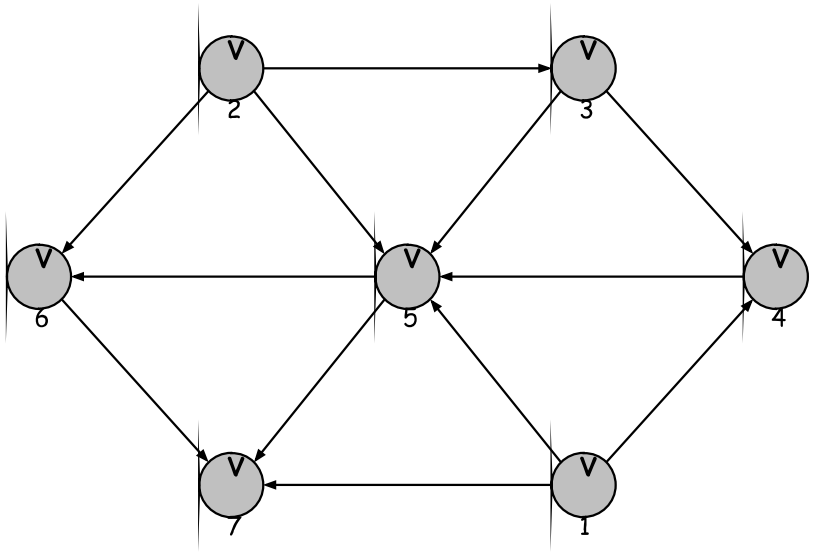
Topological Sort

Topological Sort is an ordering of vertices in a Directed Acyclic Graph (DAG), such that if there is a path from V_i to V_j then V_j appears after V_i in the ordering

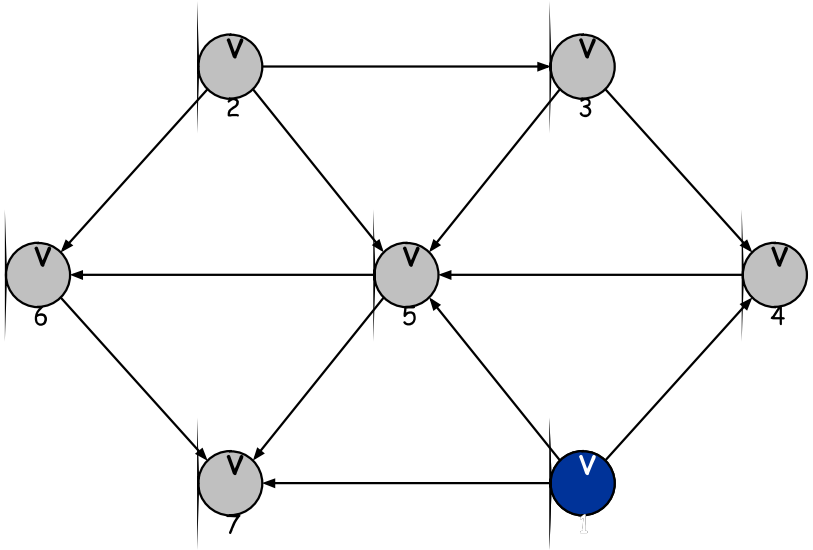
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	0
V_2	0
V_3	1
V_4	2
V_5	4
V_6	2
V_7	3

Enque V_1 V_2
QUEUE : V_1 V_2
Topological order:

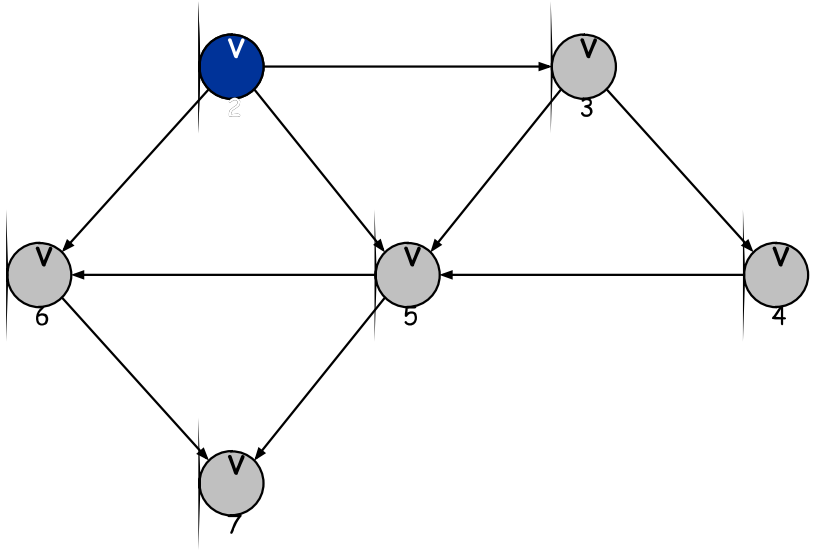
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	0
V_2	0
V_3	1
V_4	2
V_5	4
V_6	2
V_7	3

Enque V_1 V_2
QUEUE : V_1 V_2
Topological order:

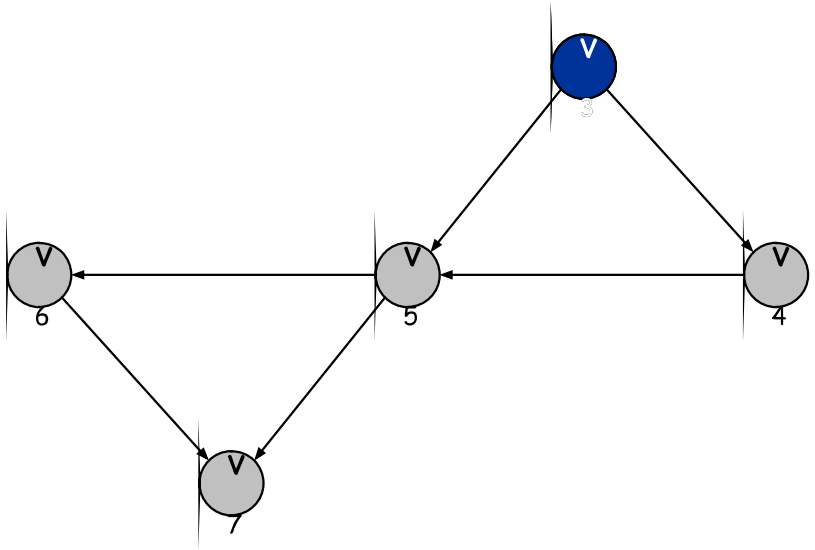
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	0
V_3	1
V_4	1
V_5	3
V_6	2
V_7	2

Deque V_1
Topological order: v_1
Enque -
QUEUE : V_2

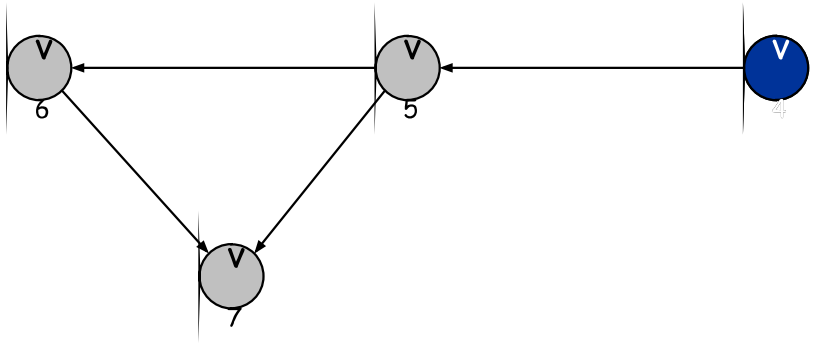
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	-
V_3	0
V_4	1
V_5	2
V_6	1
V_7	2

Deque V_2
Topological order: V_1 V_2
Enque V_3
QUEUE : V_3

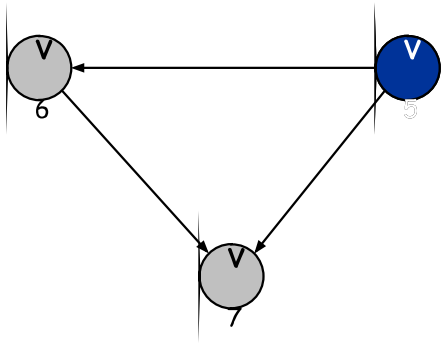
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	-
V_3	-
V_4	0
V_5	1
V_6	1
V_7	2

Deque V_3
Topological order: $V_1\ V_2\ V_3$
Enque V_4
QUEUE : V_4

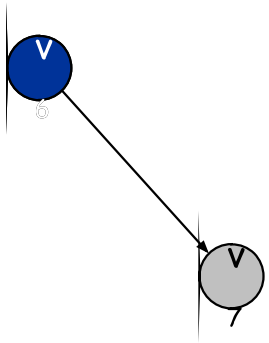
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	-
V_3	-
V_4	-
V_5	0
V_6	1
V_7	2

Deque V_4
Topological order: $V_1\ V_2\ V_3\ V_4$
Enque V_5
QUEUE : V_5

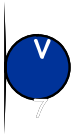
Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	-
V_3	-
V_4	-
V_5	-
V_6	0
V_7	1

Deque V_5
Topological order: $V_1\ V_2\ V_3\ V_4\ V_5$
Enque V_6
QUEUE : V_6

Topological Ordering Algorithm: Example



Vertex	*Indegree*
V_1	-
V_2	-
V_3	-
V_4	-
V_5	-
V_6	-
V_7	0

Deque V_6
Topological order: $V_1\ V_2\ V_3\ V_4\ V_5\ V_6$
Enque V_7
QUEUE : V_7

Topological Ordering Algorithm: Example

Vertex	*Indegree*
V_1	-
V_2	-
V_3	-
V_4	-
V_5	-
V_6	-
V_7	-

Deque V_7

Topological order: $V_1\ V_2\ V_3\ V_4\ V_5\ V_6\ V_7$

Topological Sort Algorithm

```
Algorithm to find a Topological Sort T of a Directed Acyclic Graph,
G

Step 1: Find the indegree INDEG(N) of every node in the graph
Step 2: Enqueue all the nodes with a zero in-degree
Step 3: Repeat Steps 4 and 5 until the QUEUE is empty
Step 4: Remove the front node N of the QUEUE by setting FRONT =
        FRONT + 1
Step 5:   Repeat for each neighbor M of node N:
          a) Delete the edge from N to M by setting INDEG(M) =
INDEG(M) - 1
          b) IF INDEG(M) = 0, then Enqueue M, that is, add M to
the rear of the queue
          [END OF INNER LOOP]
    [END OF LOOP]
Step 6: Exit
```