Depth-First Search

DFS-VISIT(u) — 1 call per vertex

mark u as processing . . . $O(1)$

for each neighbors v of the current vertex u

  if v still unseen then DFS-VISIT(v) $O(1)$

  (if v is processing then we have a cycle)

mark u as finished $O(1)$

$O(n+m)$ using adj list

[$O(n^2)$ for adj matrix]

unseen

✓ processing (recursive call still active)

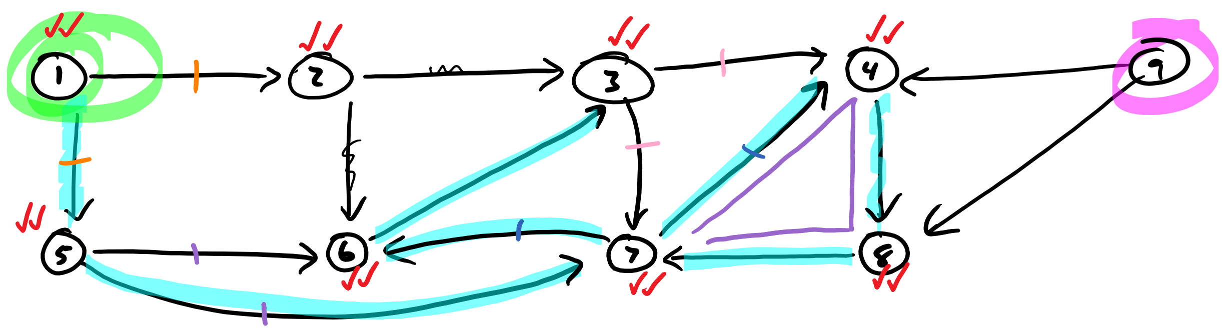
✓✓ finished (recursive call complete)

DFS-VISIT(1)

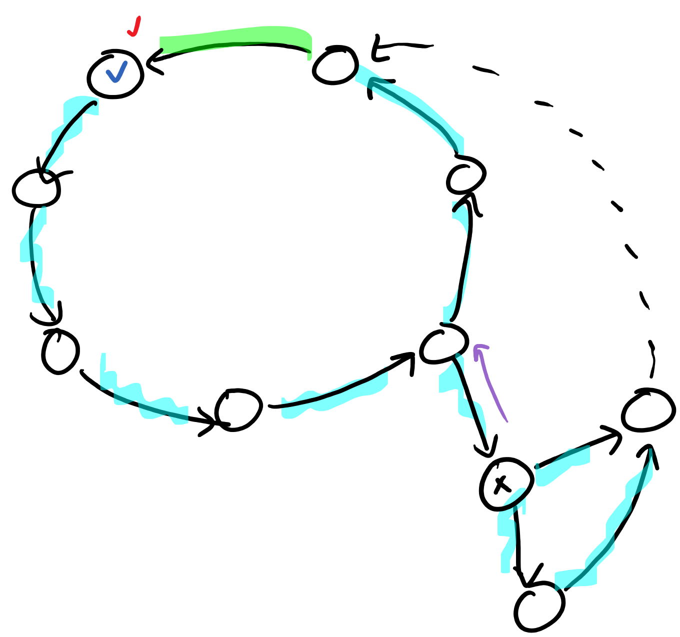
- DFS-VISIT(5)
  - DFS-VISIT(7)
    - DFS-VISIT(4)
      - DFS-VISIT(8)
    - DFS-VISIT(6)
      - DFS-VISIT(3)
- DFS-VISIT(2)

color

1	2	3	. . .	8	9
✓✓	✓✓	✓✓	. . .	✓✓	

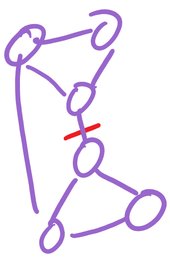


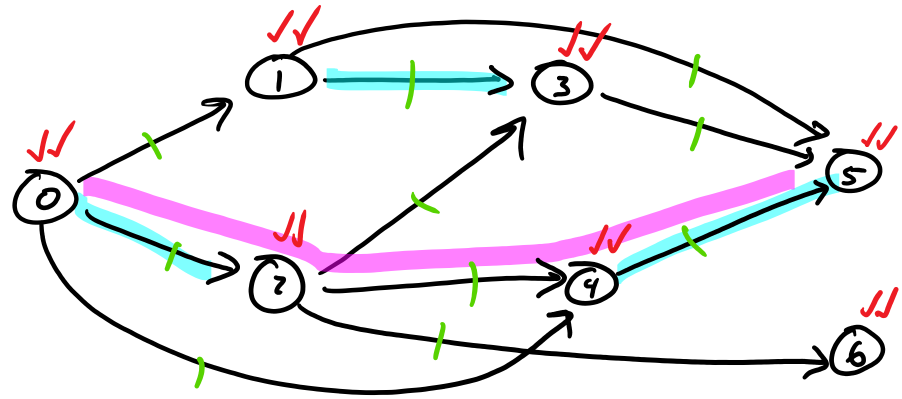
suppose v is 1st vertex on cycle we visit

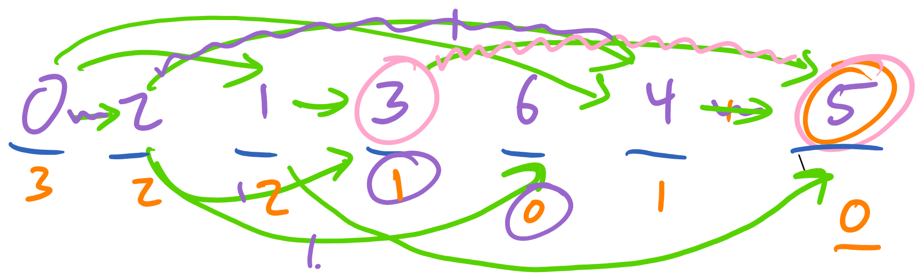


Topological Sort

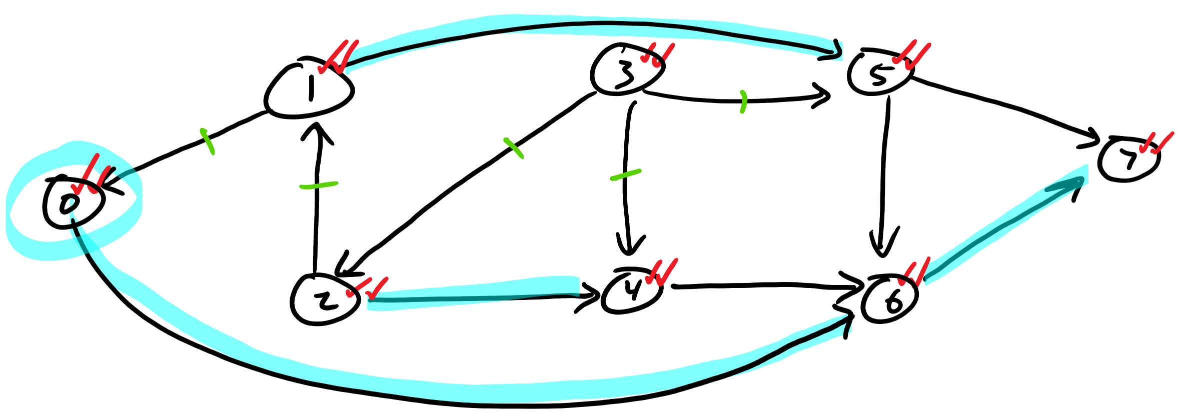
↳ given directed, acyclic graph, order vertices so all edges go →

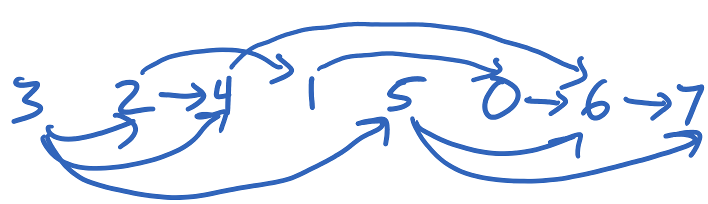






longest path starting at ✓





Dijkstra's Algorithm

PRE: no negative weight edges

POST: d gives total weight of shortest paths, pred gives edges in shortest paths

(∞ to mean unreachable)

```
for each v
  color[v], pred[v], d[v] ← IN_QUEUE, NIL, ∞
d[s] ← 0

Q ← new PriorityQueue(d)
```

(build Q to contain all vertices, w/ d[v] being the initial priority of v)

```
while Q not empty
  u ← dequeue(Q)
  for each outneighbor v of u
    if color[v] = IN_QUEUE and d[v] > d[u] + w(u, v)
      change_priority(Q, v, d[u] + w(u, v))
      d[v] ← d[u] + w(u, v)
      pred[v] ← u
  color[u] ← DONE
```

while Q not empty: one iteration per vertex; dequeue(Q): n calls, $O(\log n)$ per call

for each outneighbor v of u: one iteration per edge

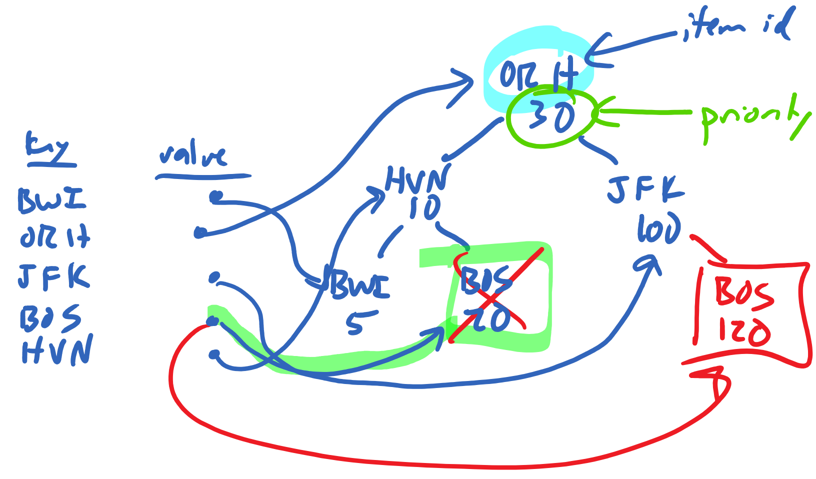
change_priority: $\leq m$ calls, $O(\log n)$ per call

adjacency list: $O(n+m)$ + time for priority queue operations

$O(n \log n) + O(m \log n)$

Priority Queue

	unsorted array	sorted array (by priority)	balanced BST (priority for order)	heaps
enqueue(elt, pri)	$O(1)$	$O(n)$	$O(\log n)$	$O(\log n)$
dequeue()	$O(n)$	$O(1)$ (ring buffer / wraparound)	$O(\log n)$	$O(\log n)$
change-priority(elt, pri)	$O(1)$	$O(n)$	$O(\log n)$	$O(\log n)$
n enqueue/dequeue + m change priority ($n-1 \le m < n^2$)	$O(n^2)$	$O(nm)$	$O(m \log n)$	
$m \in \Theta(n)$	$O(n^2)$	$O(n^2)$	$O(n \log n)$	
$m \in \Theta(n^2)$	$O(n^2)$	$O(n^3)$	$O(n^2 \log n)$	



change-priority(BOS, 120)