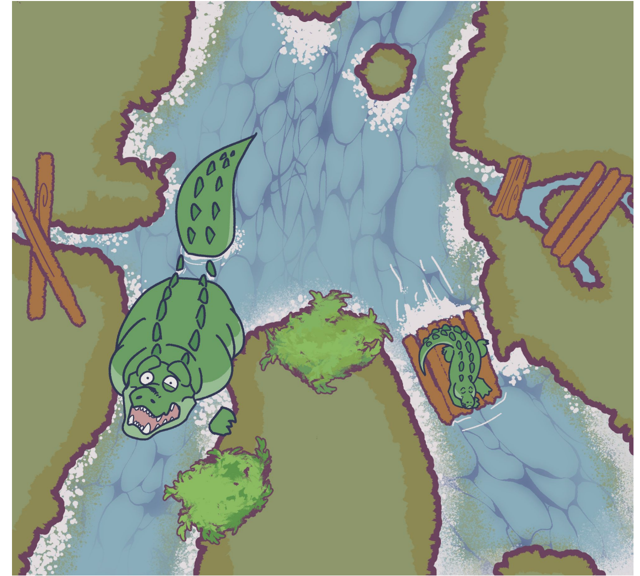
7/15 Lecture Agenda

- Announcements
- Part 4-3: DFS and Topological Sort
- 10 minute break!
- Part 4-4: Kosaraju's Algorithm

Announcements!

- HW3 due Sunday
- HW4 out Sunday, not due until well after midterm (and not required for midterm)
- Pre-HW4 due the day after the midterm (and also not required)
- Next week: no new material! A chance to catch up!
  - **M:** Problem Session 4
  - **W:** Midterm Review
  - **F:** Midterm



In HW4 Problem 6, you'll help large gators navigate networks of sometimes-too-small channels!

7/15 Lecture Agenda

- Announcements
- Part 4-3: DFS and Topological Sort
- 10 minute break!
- Part 4-4: Kosaraju's Algorithm

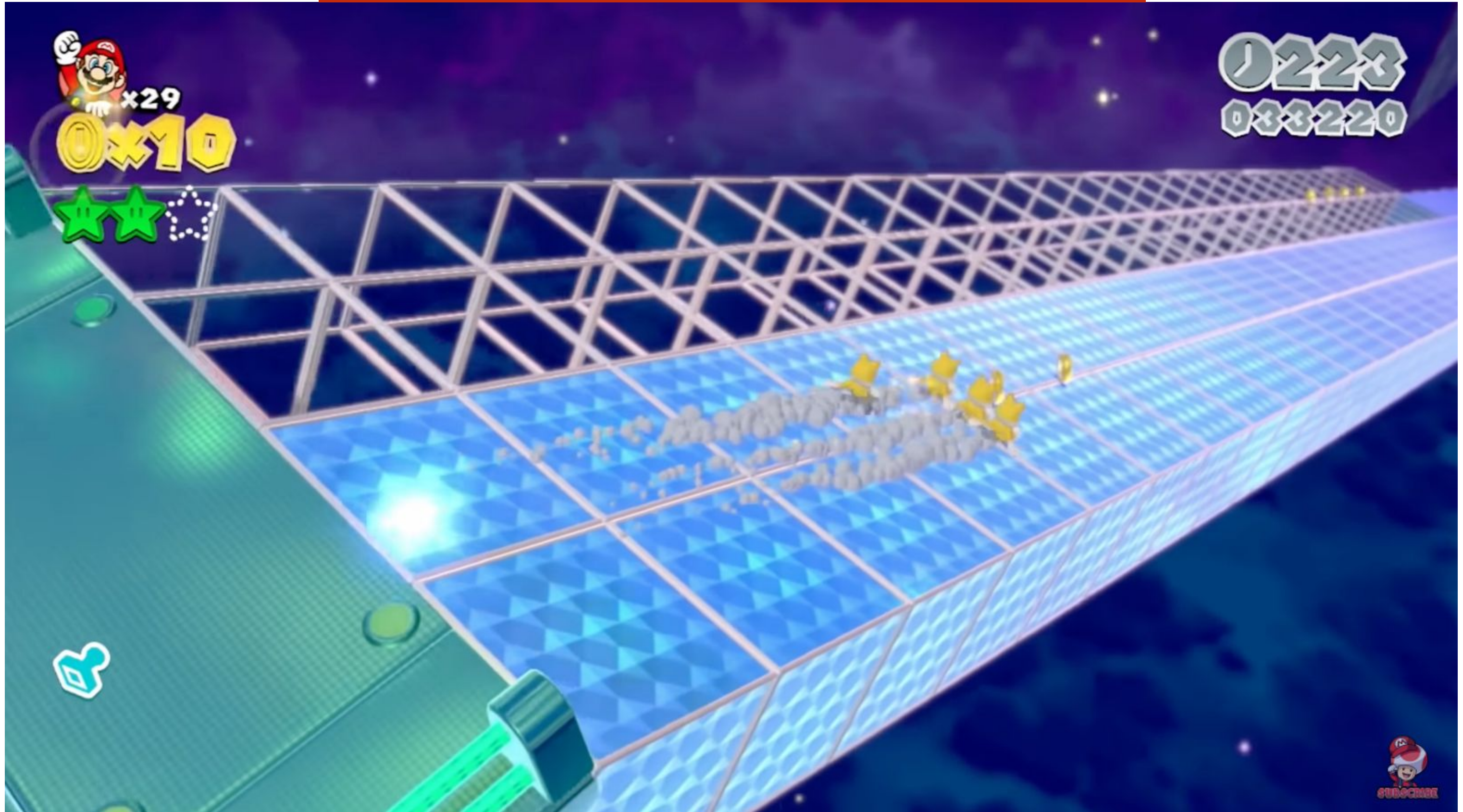
WORLD 4-3

DFS and TopoSort

Divide and Conquer
Sorting & Randomization
Data Structures
Graph Search
Dynamic Programming
Greed & Flow

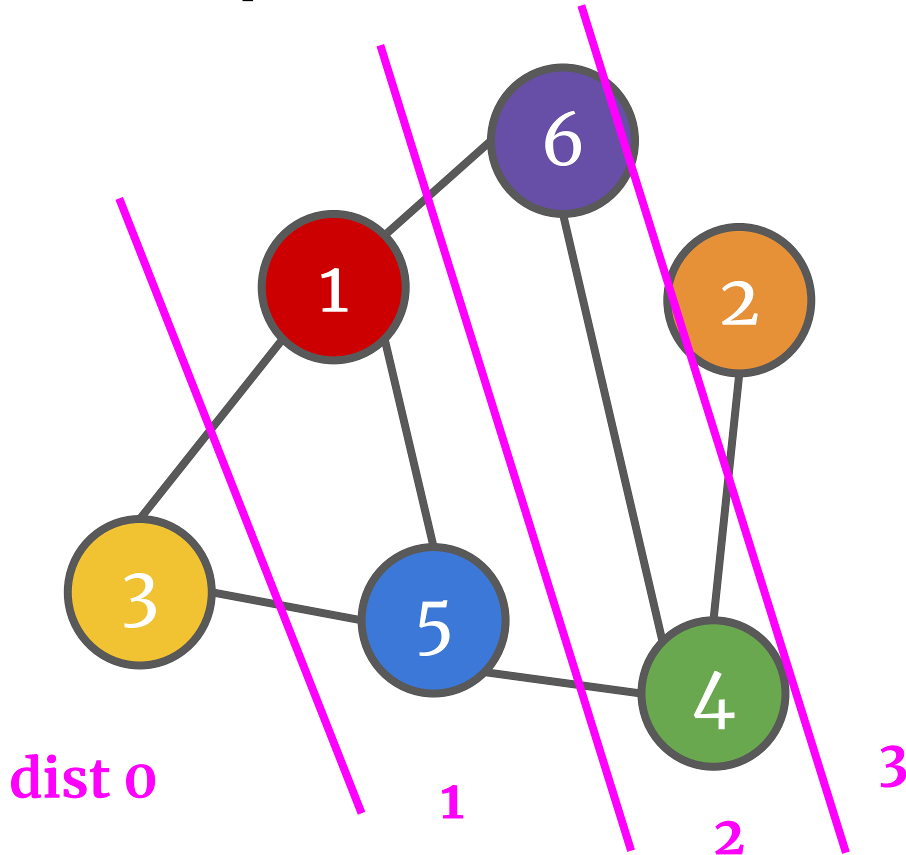
Special Topics

aside: the actual WORLD 4-3 is great



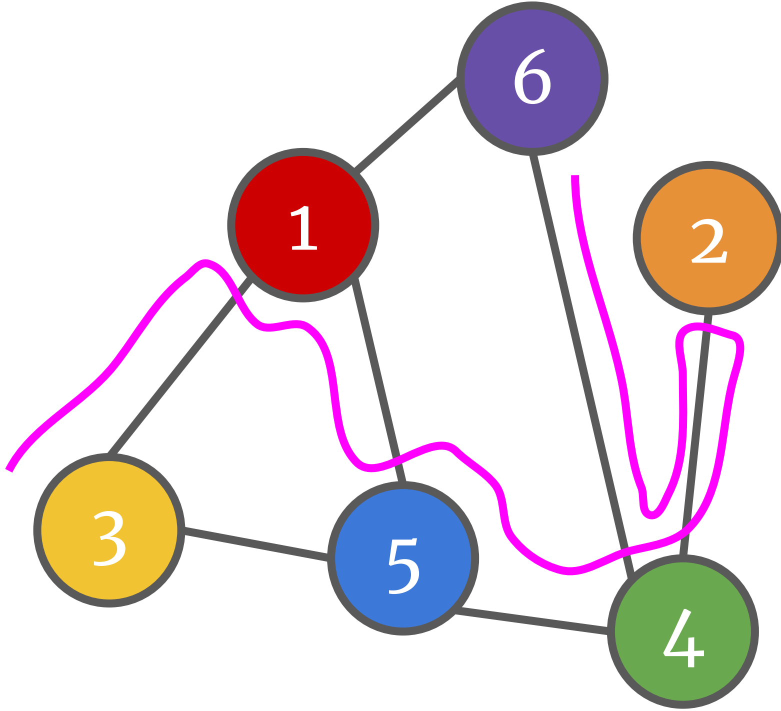
BFS vs. DFS thinking

Recap: BFS



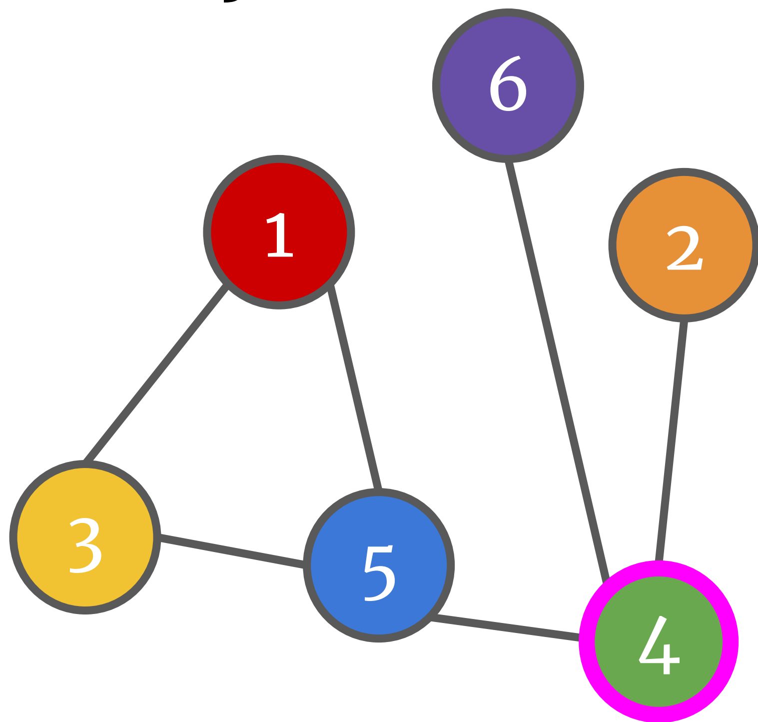
- Queue-based
  - (In this class, you can just think of a queue as being a linked list)
- Visit all nodes 1 step away from the start, then all nodes 2 steps away from the start...

But what about: DFS

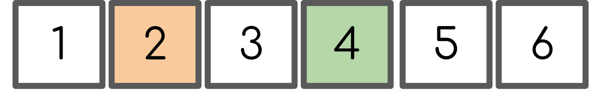


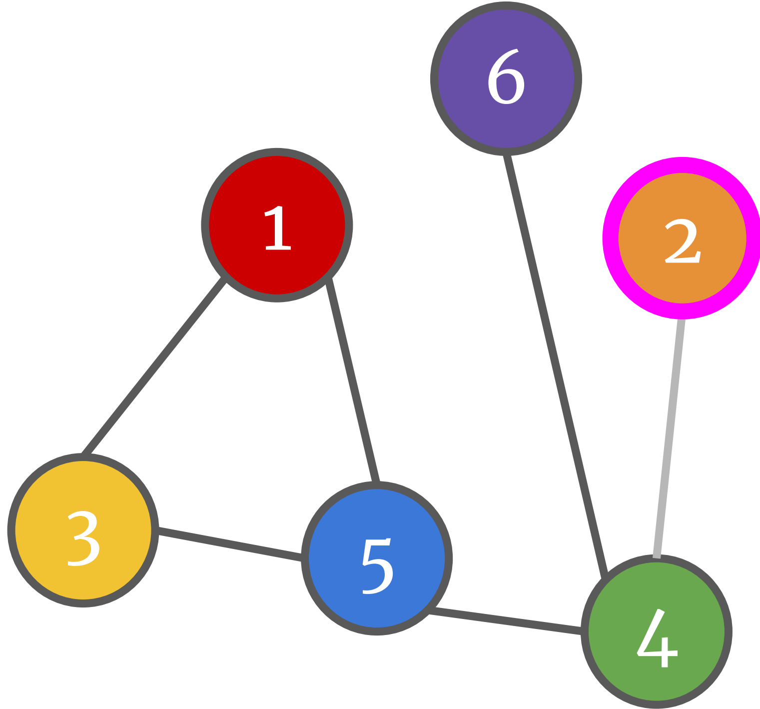
- **Stack**-based
- Kinda like in BFS, keep track of which nodes have been visited.
- Keep visiting an unvisited neighbor (breaking ties by, e.g., node number)
- Backtrack whenever we are out of options.

Let's try!



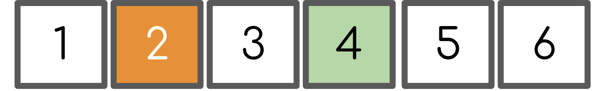
- Start at 4.
- Mark 4 as visited (but not done).
- Check 4's neighbors: 2, 5, 6.
- All are unvisited, so go with the numerically earliest.

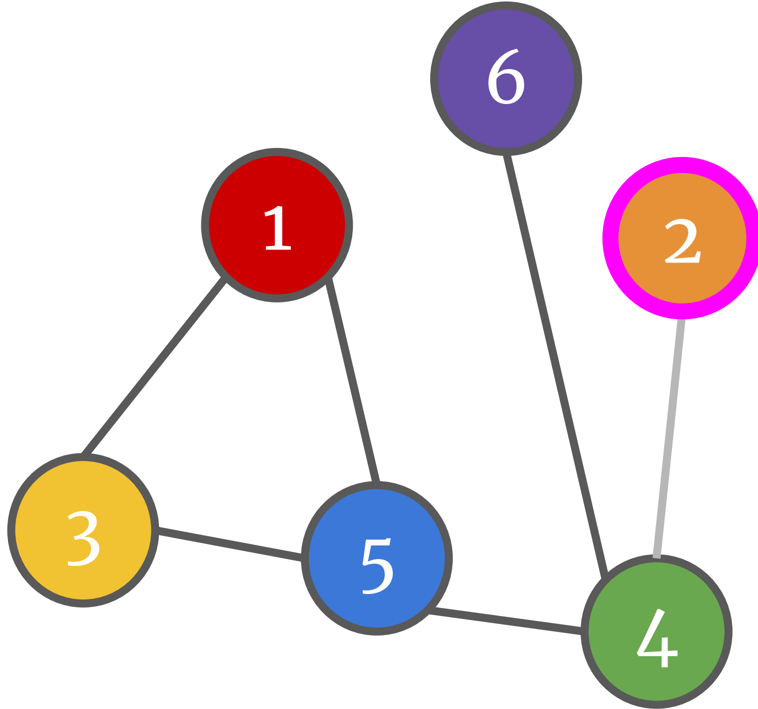




- Now at 2.
- Mark 2 as visited (but not done).
- Check 2's neighbors: 4. But 4 is visited.

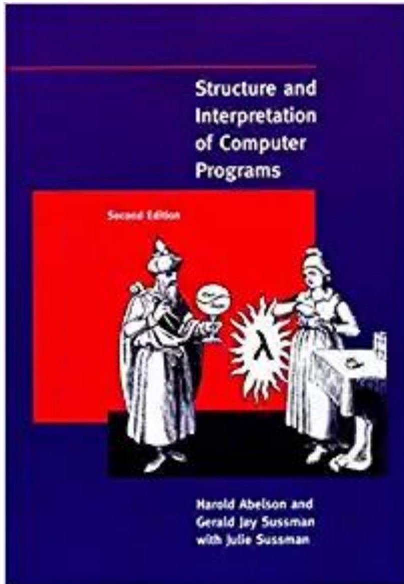
Status:





- Nothing else to do at 2! Mark it as done.
- Backtrack one step to 4. (Imagine we have a piece of string leading back to where we were. In code terms, this could be a function call stack or an explicit stack.)

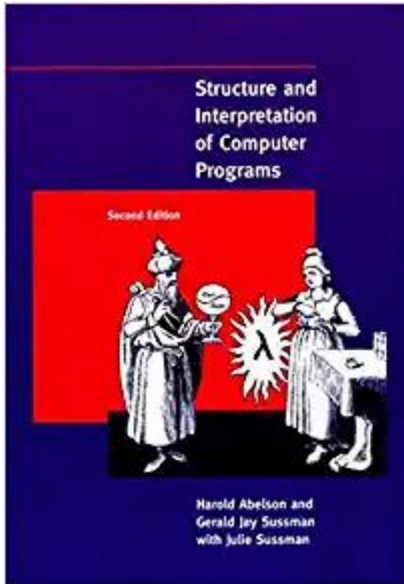
A quick tragic tale from Cal



CS61A (like our 106A)

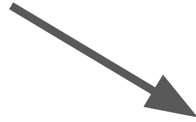
- Recursion is beautiful!
- Do everything with a bunch of recursive calls!

A quick tragic tale from Cal



CS61A (like our 106A)

- Recursion is beautiful!
- Do everything with a bunch of recursive calls!

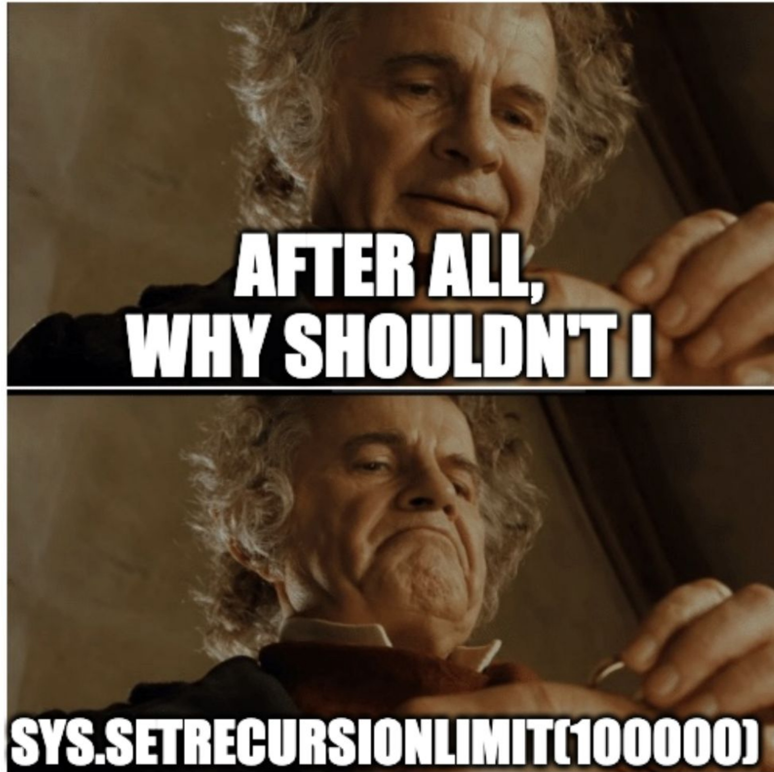


CS61B (like our 106B)

- First assignment was a Photoshop app with floodfill
- Recursive calls cause a stack overflow
- Oh cool so I guess this is the real world

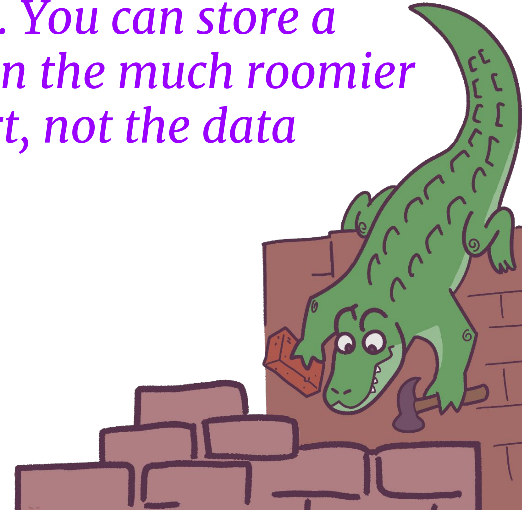
A not-ideal way of handling this

A not-ideal way of handling this

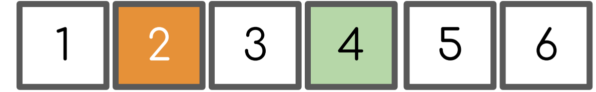


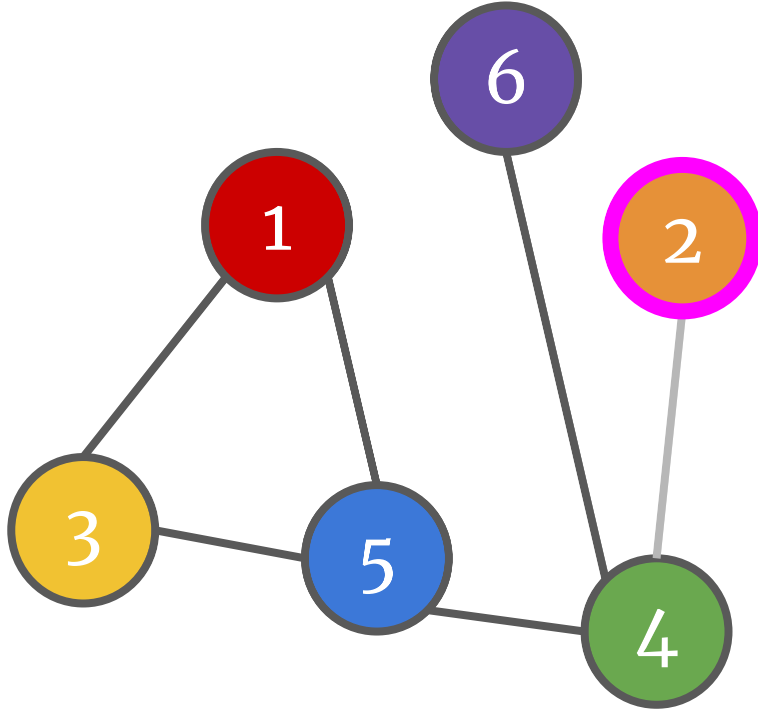
It's usually better to rewrite recursive code to be iterative!

Or at least don't rely too heavily on the stack of function calls. You can store a stack representation in the much roomier heap! (the system part, not the data structure)

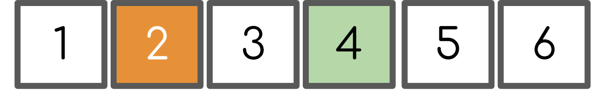


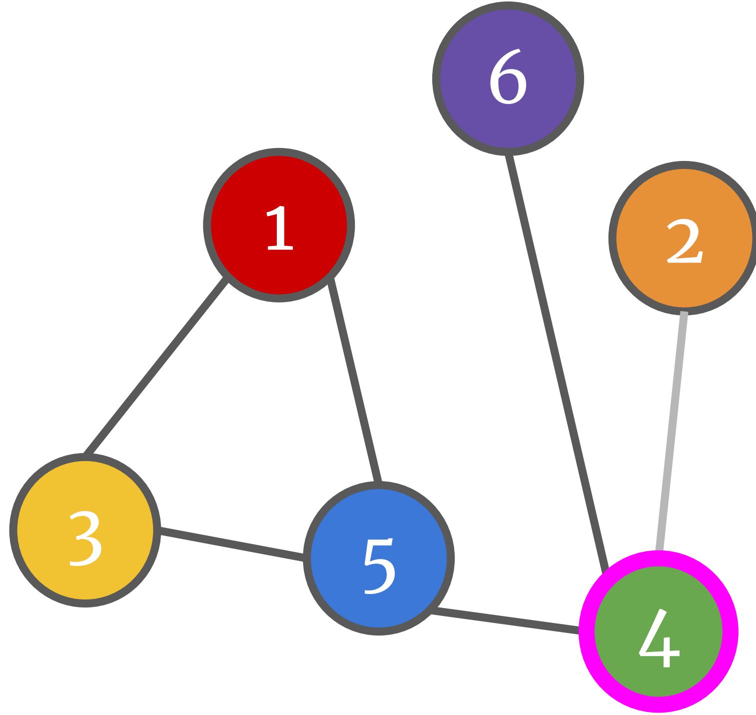
Where were we...



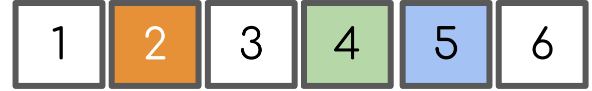


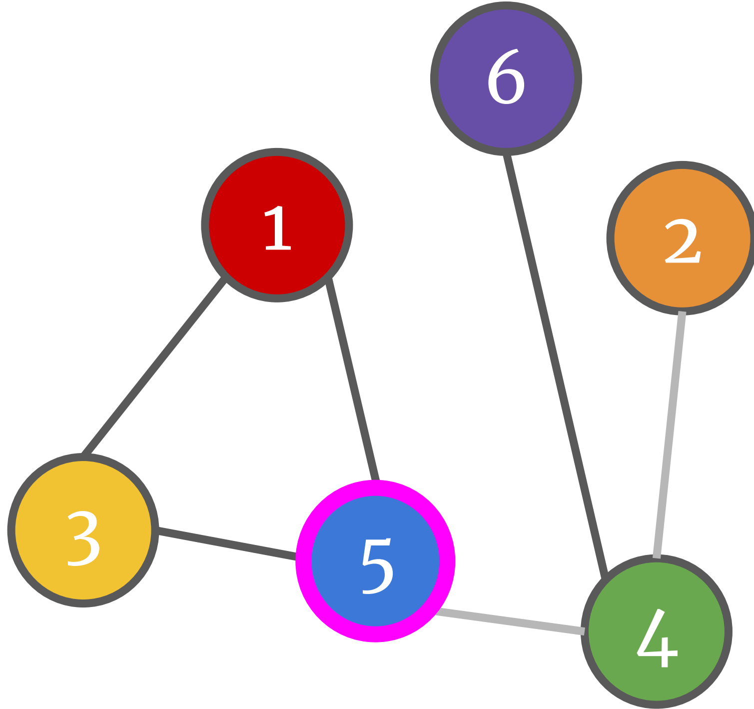
- Nothing else to do at 2! Mark it as done.
- Backtrack one step to 4. (Imagine we have a piece of string leading back to where we were. In code terms, this could be a function call stack or an explicit stack.)



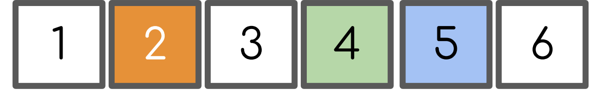


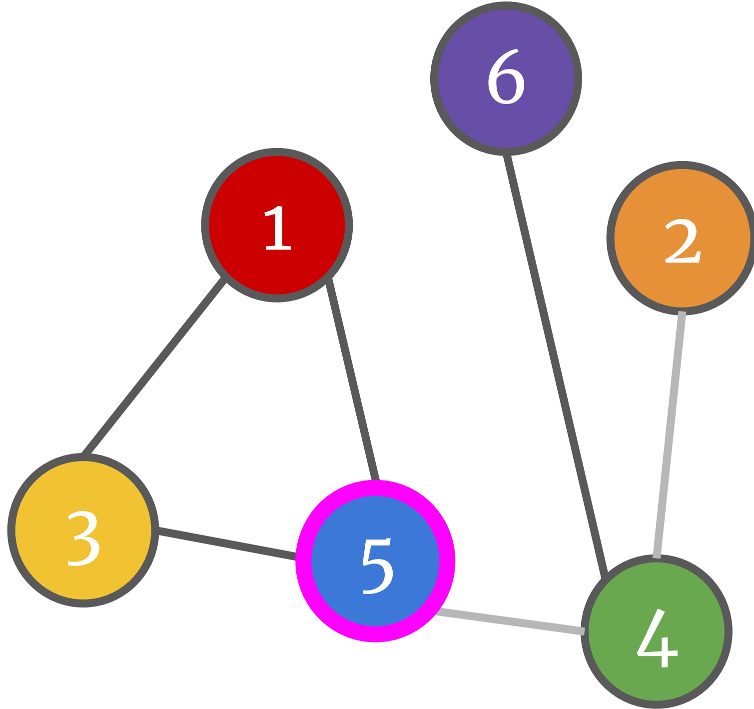
- Now at 4 again.
- Go to the next highest-numbered unvisited neighbor: that's 5!





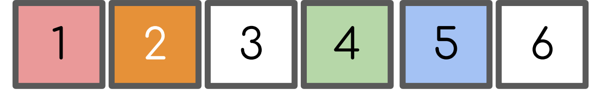
- Now at 5.
- Mark 5 as visited (but not done).
- Check 5's neighbors: 1, 3, 4. Go to the lowest-numbered unvisited one: 1.

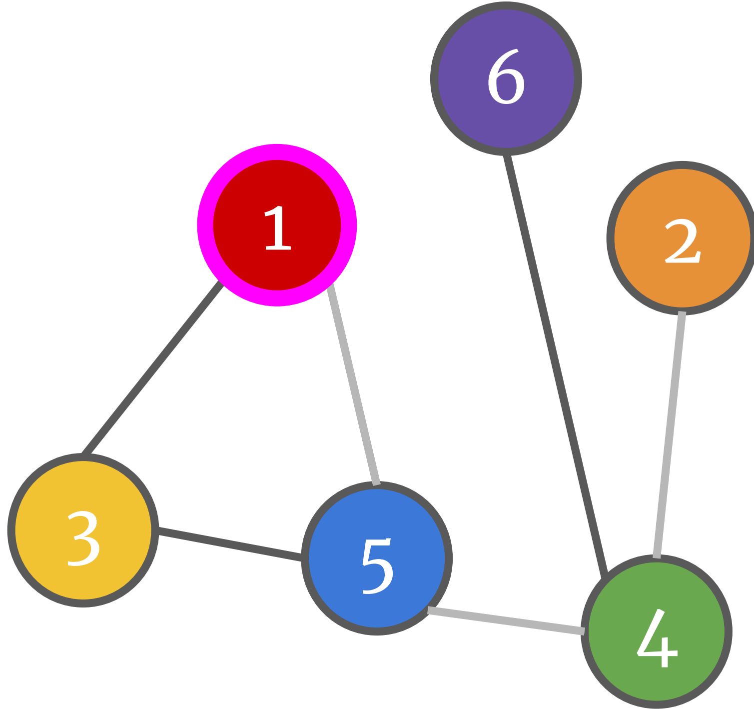




- Now at 5.
- Mark 5 as visited (but not done).
- Check 5's neighbors: 1, 3, 4. Go to the lowest-numbered unvisited one: 1.

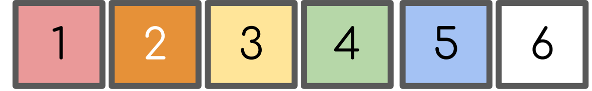
Status:

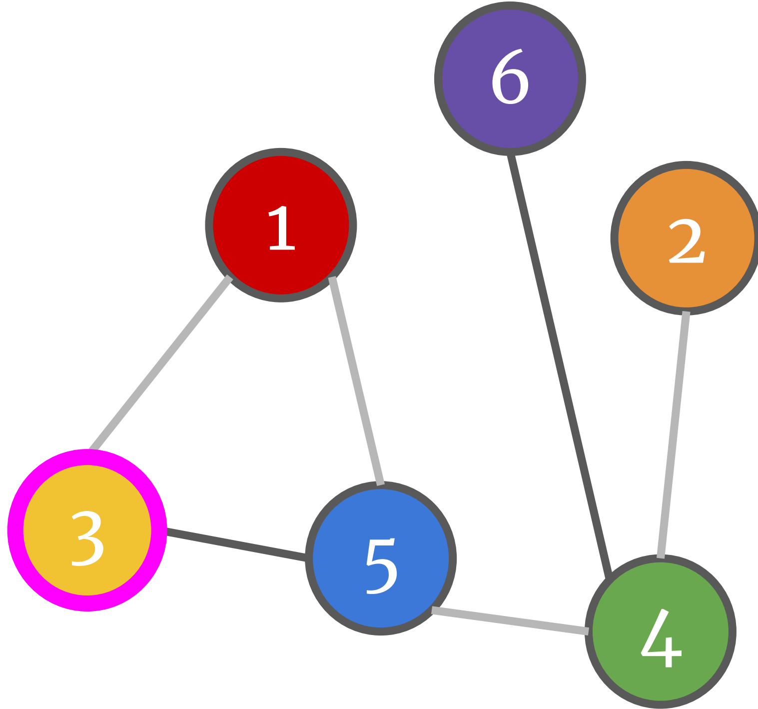




- Now at 1.
- Mark 1 as visited (but not done).
- Check 1's neighbors: 3, 5. Go to the lowest-numbered unvisited one: 3.

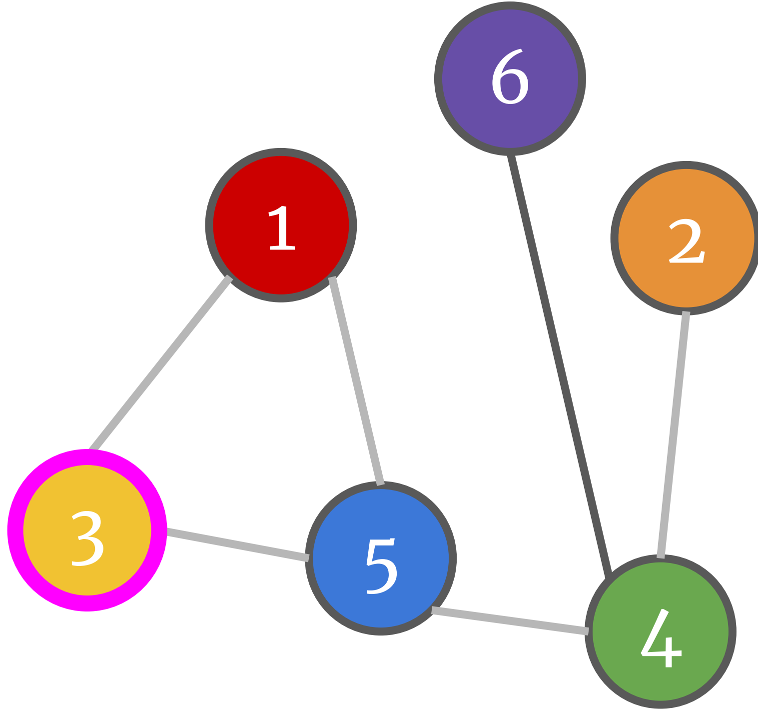
Status:





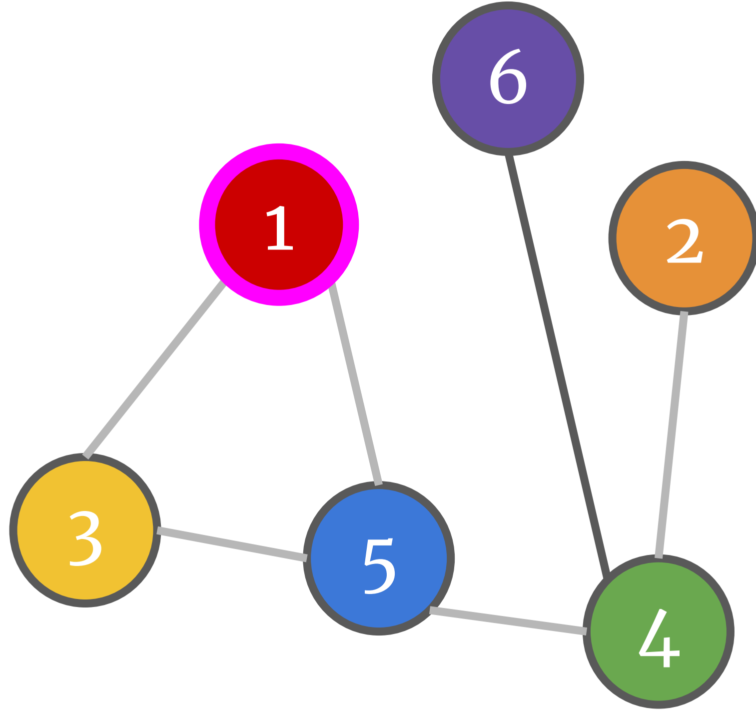
- Now at 3.
- Mark 3 as visited (but not done).
- Check 3's neighbors: 1, 5. But both are visited.

Status:

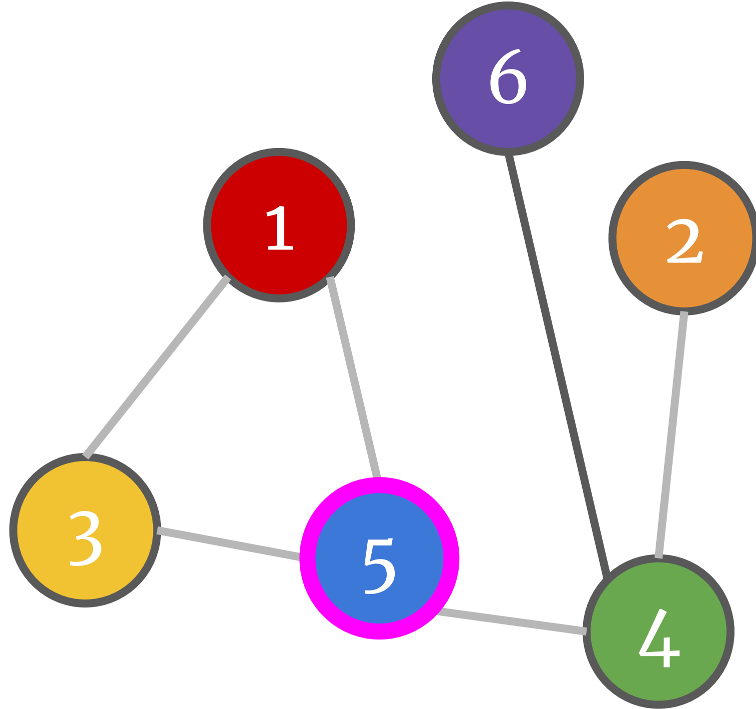


- Nothing else to do at 3! Mark it as done.
- Backtrack one step to 1.

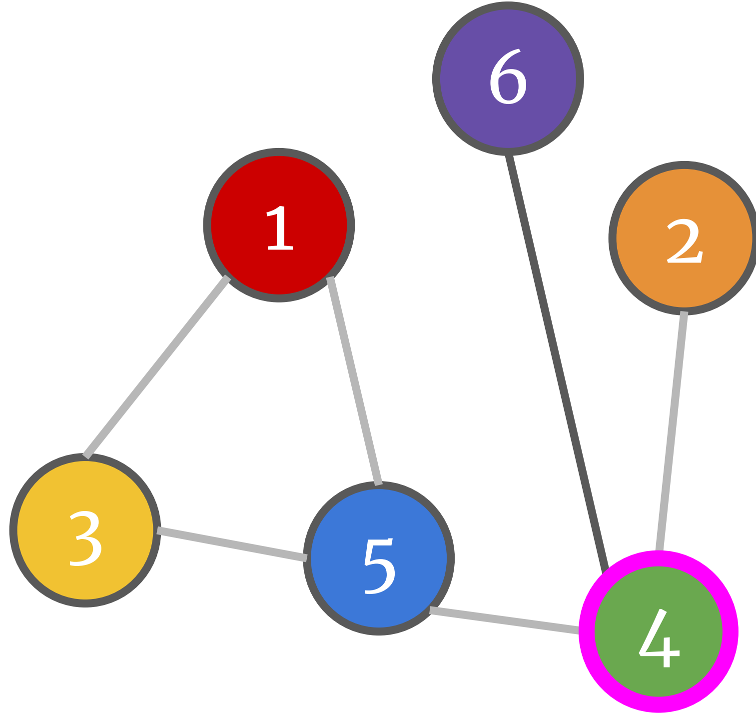
Status:



- Nothing else to do at 1! Mark it as done.
- Backtrack one step to 5.

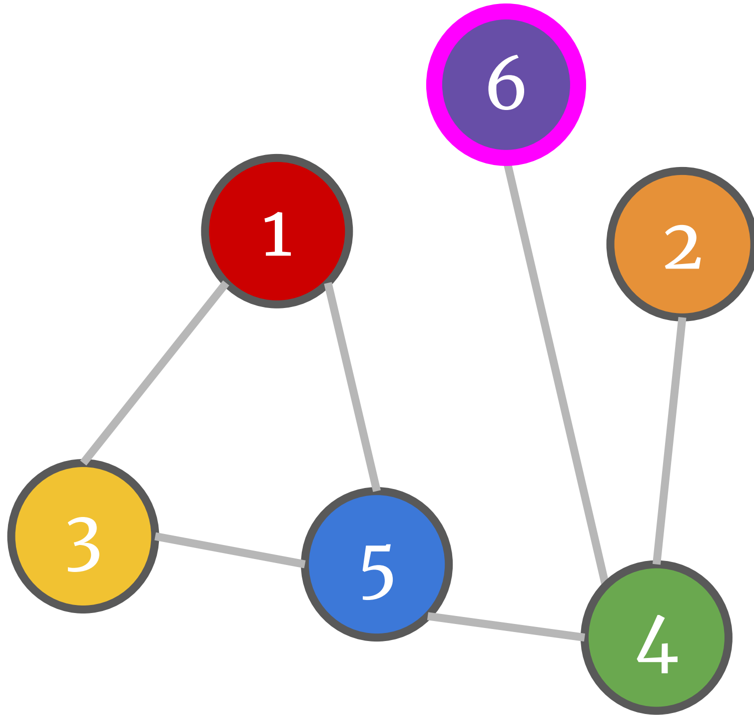


- Nothing else to do at 5! Mark it as done.
- Backtrack one step to 4.

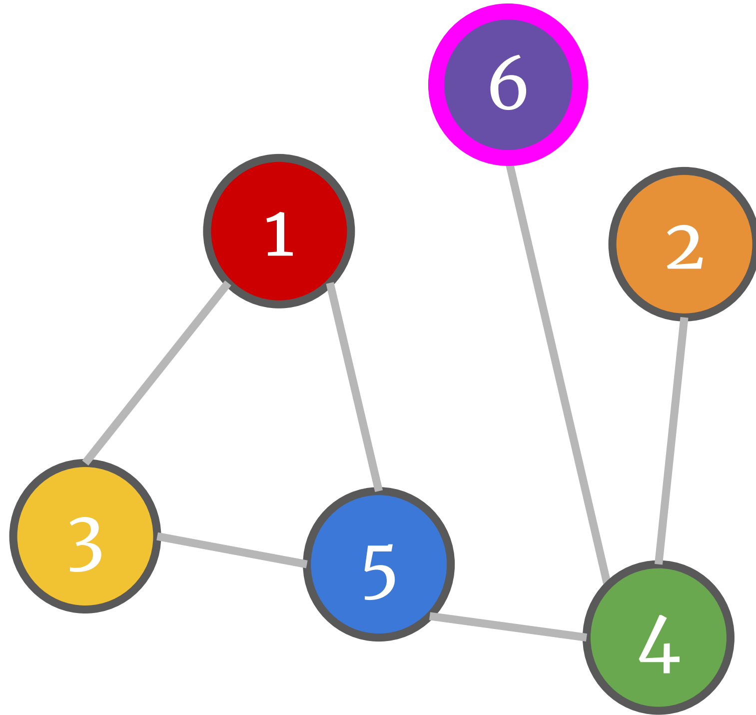


- Now at 4 (again).
- Visit the lowest-numbered unvisited neighbor: that's 6.

Status:

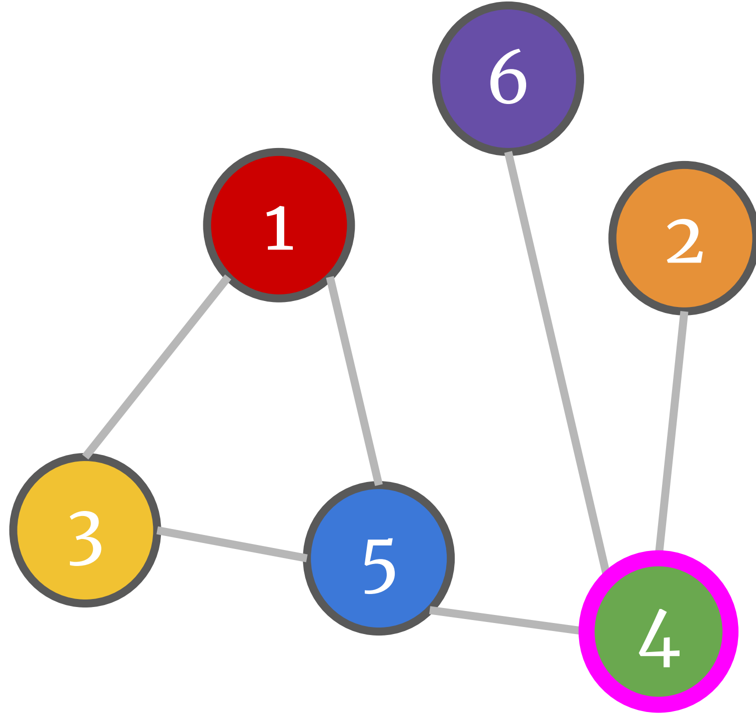


- Now at 6.
- Mark 6 as visited (but not done).
- Check 6's neighbors: 4. But 4 is visited.



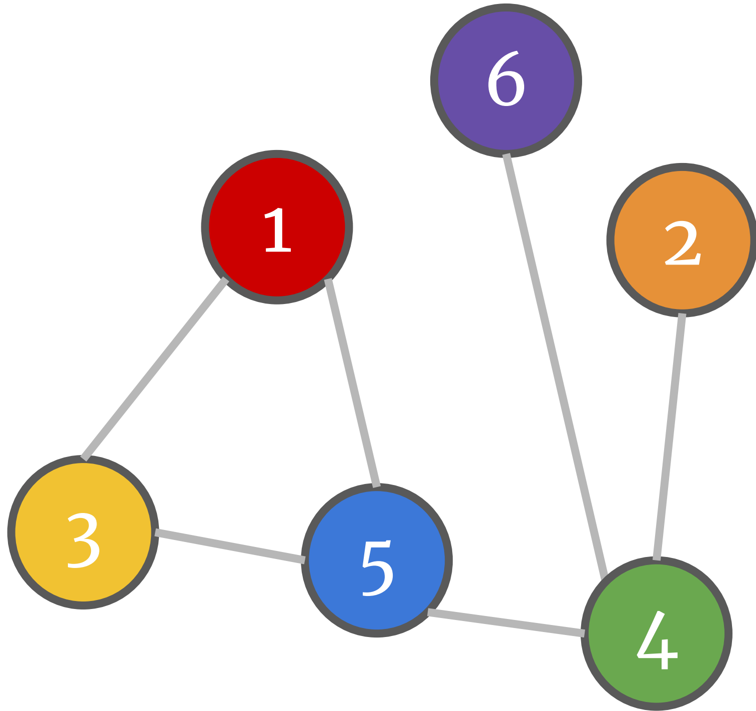
- Nothing else to do at 6, so mark it done.
- Backtrack 1 step to 4.

Status:



- Now at 4 (again!)
- Nothing else to do at 4, so mark it as done.
- Nowhere to backtrack to, so we are finished!

Running time



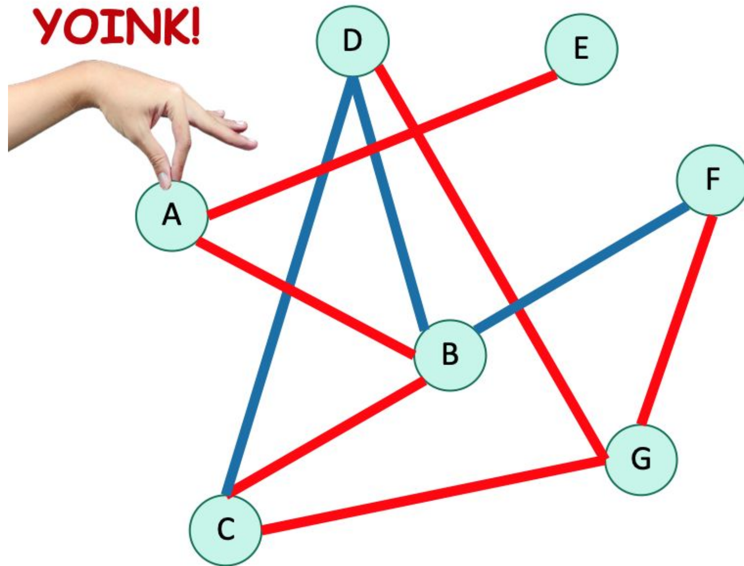
Like in BFS, we considered each edge at most twice – once on the way "out", and once on the way "back".

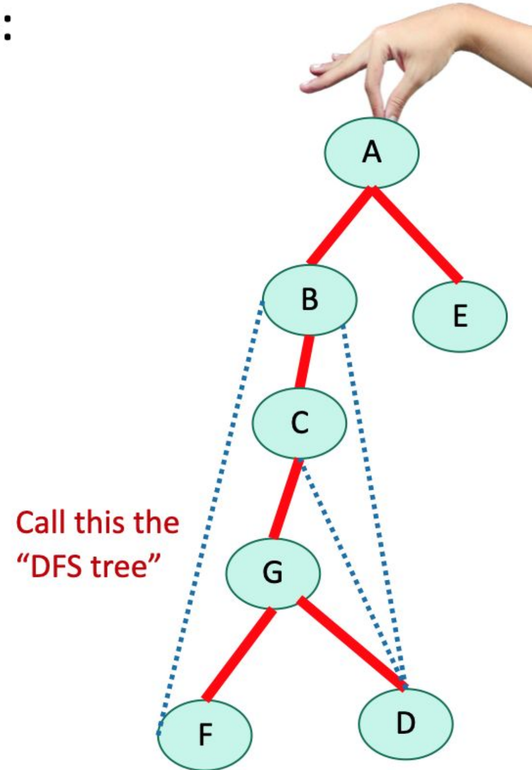
Also like in BFS, we had to look at all the neighbors of each node. (Once we have done this once, we can advance through the list instead of looking again)

So: **O(*n* + *m*)**, like BFS.

Why is it called depth-first?

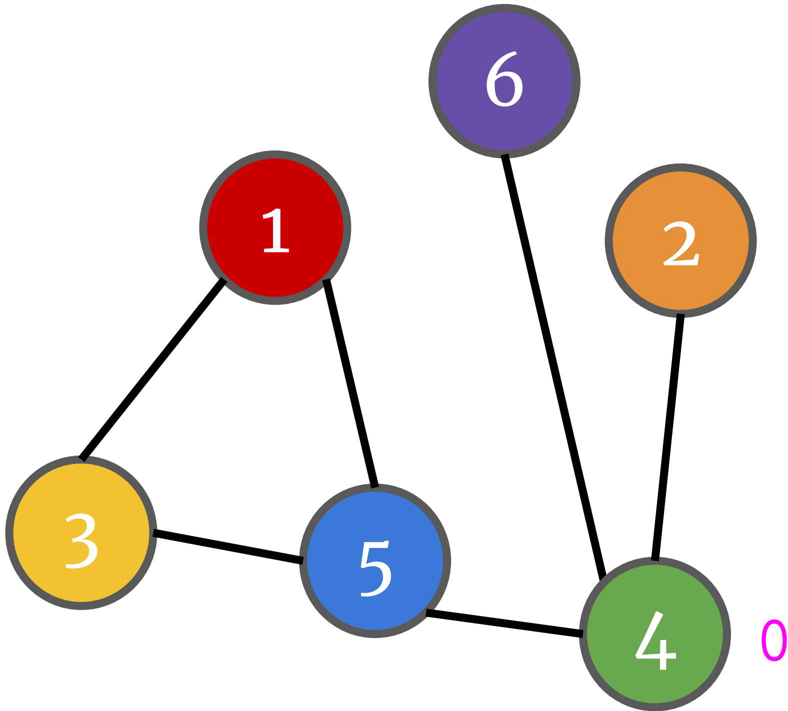
- We are implicitly building a tree:





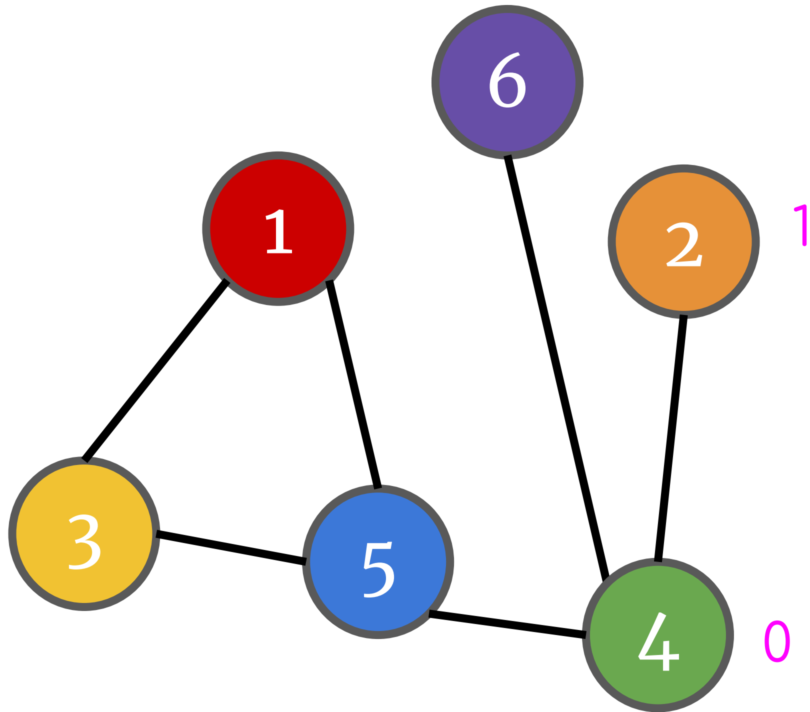
- First, we go as deep as we can.

Now what if we want to find distances?



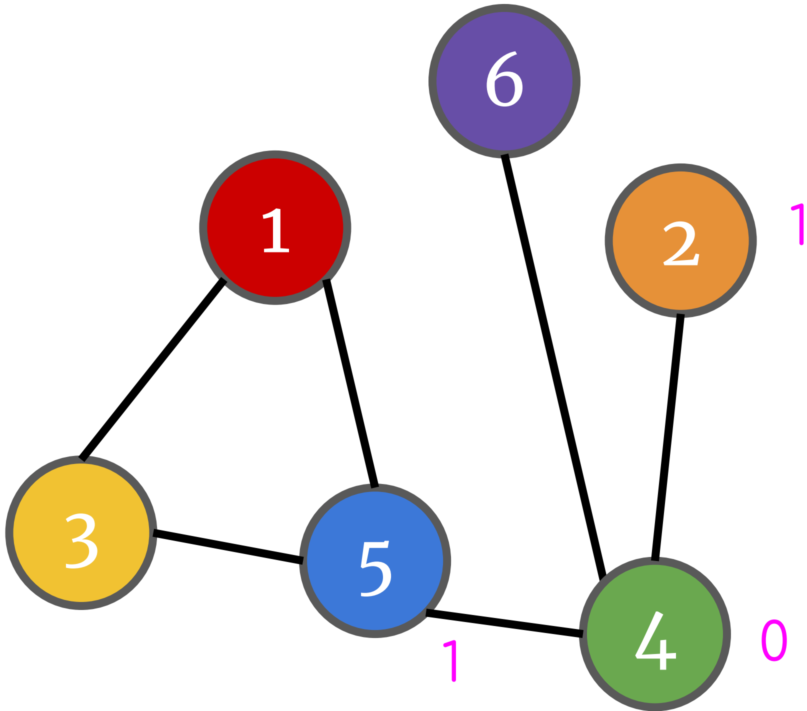
Sure, let's try it!

Now what if we want to find distances?



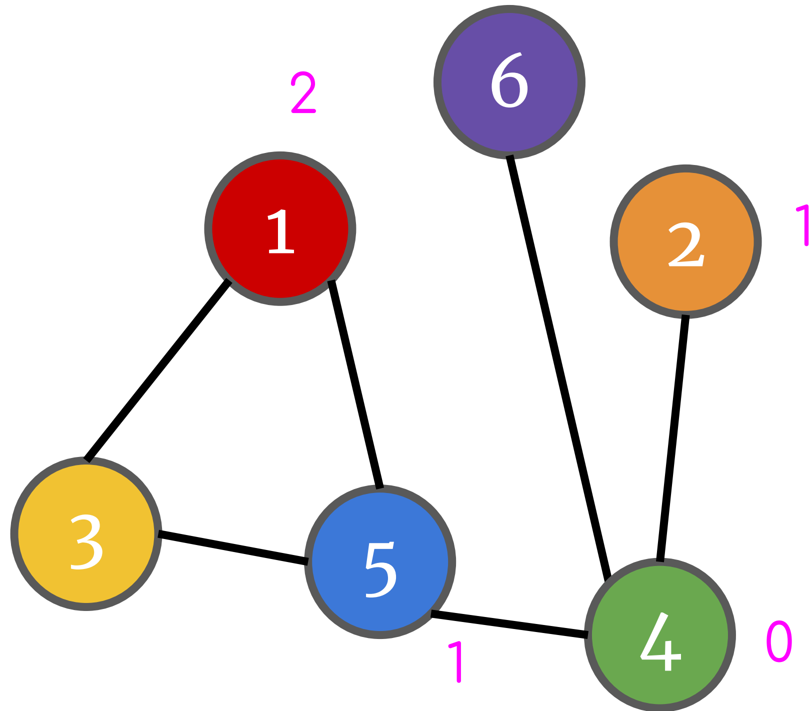
Sure, let's try it!

Now what if we want to find distances?



Sure, let's try it!

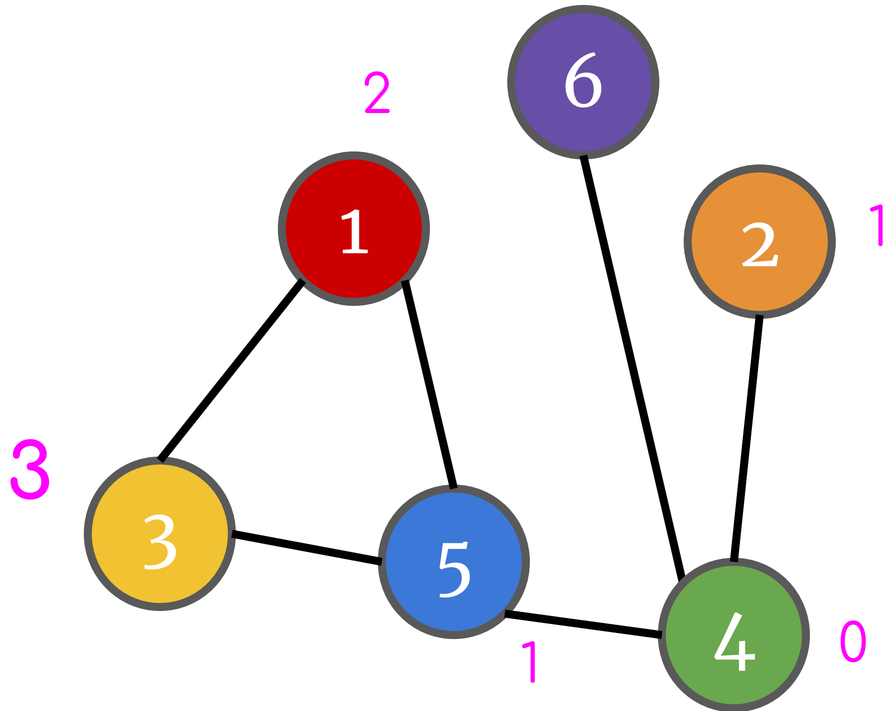
Now what if we want to find distances?



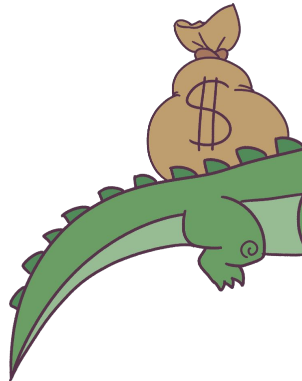
Sure, let's try it!

This is looking great!
Call the investors.

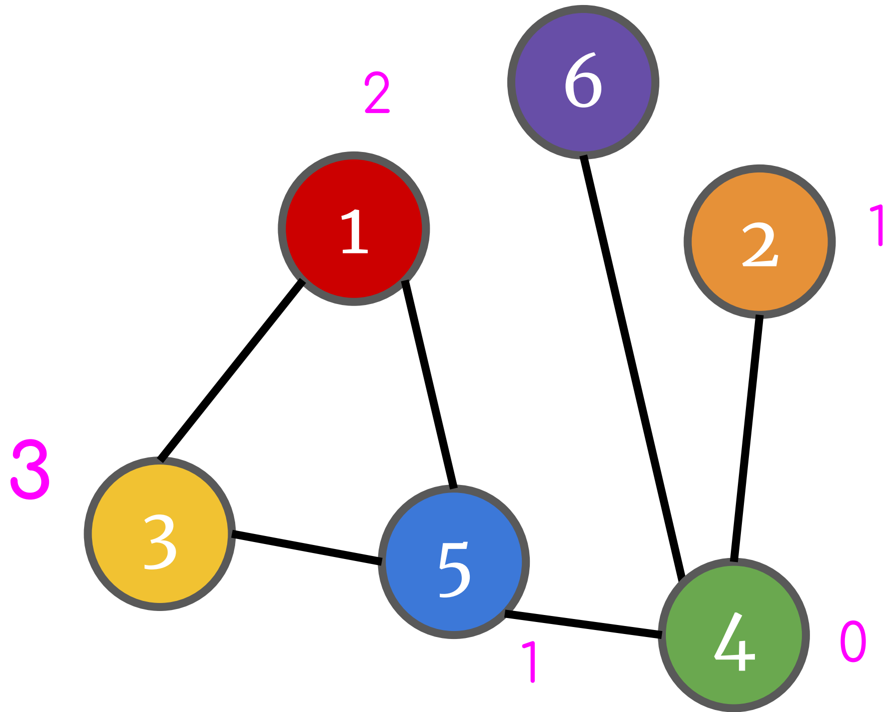
Uh oh - this is wrong!



Keep the investors busy and call my private jet!



What happened?



DFS charges ahead! It might never take the true shortest path to (one or more) nodes.

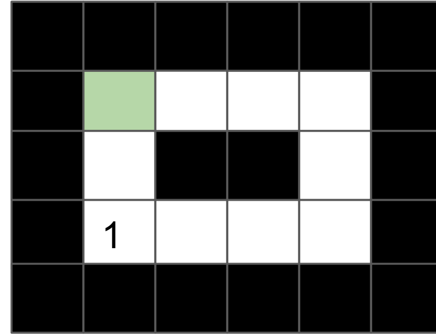
It is not suitable for finding shortest paths.

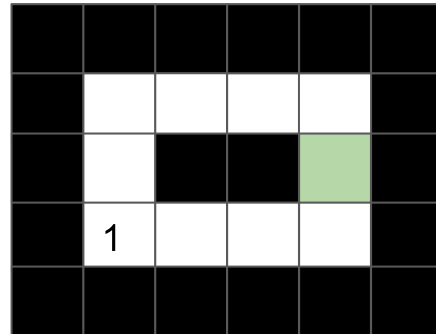
DFS looks broken! Why not just BFS?

- DFS can traverse / find all nodes and connected components, but so can BFS!

- DFS can solve mazes, but so can BFS!
  - though BFS will do better on some mazes than DFS, and vice versa...

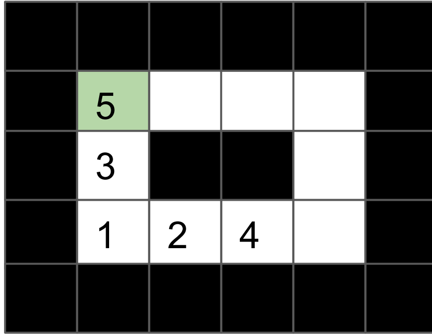
Suppose nodes to the right beat nodes above in ties.
Which method is worse on each of these mazes?

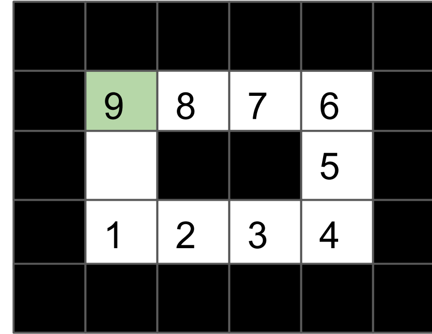




BFS

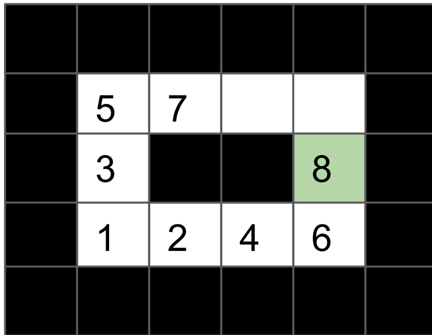
DFS

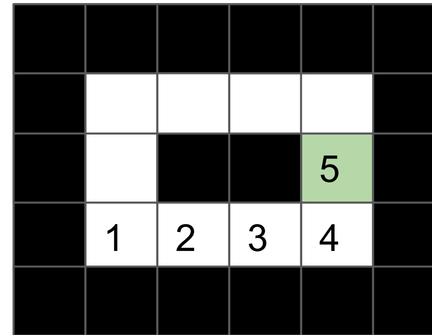




DFS is worse

BFS is worse





Strengths of DFS

- We often *want* to traverse trees in a depth-first way. For example, an inorder traversal of a binary search tree is a DFS with some extra printing.

- Some algorithms, like topological sort (coming in the second half!), rely on DFS and its stack-based operation.

- DFS is also one way to solve a classic problem called 2-SAT (possibly coming on HW4)

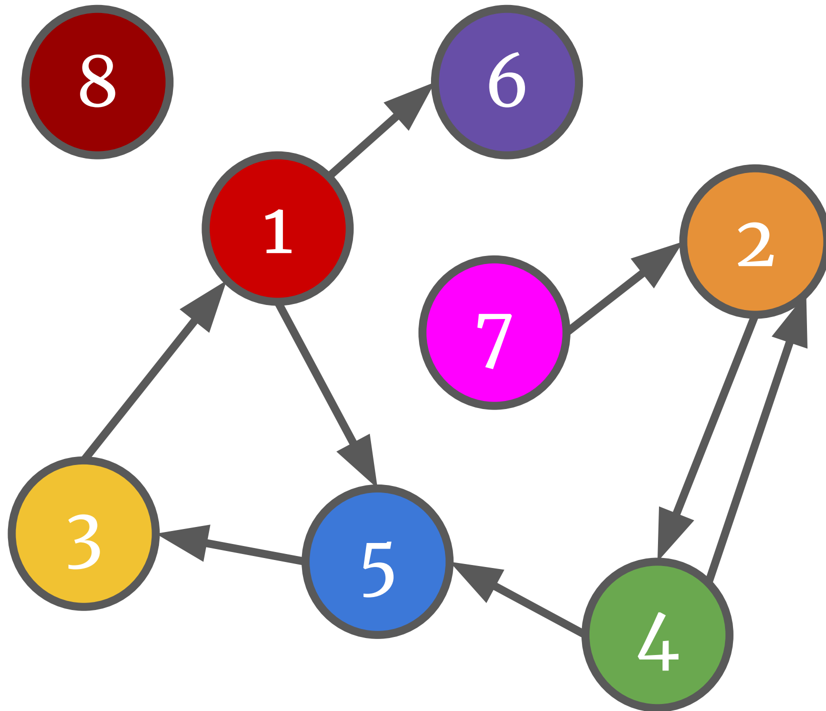
It's Time For...

One Direction

It's Time For...

Directed Edges

New phenomena!



Node 7 is a **source** since it has no incoming edges.

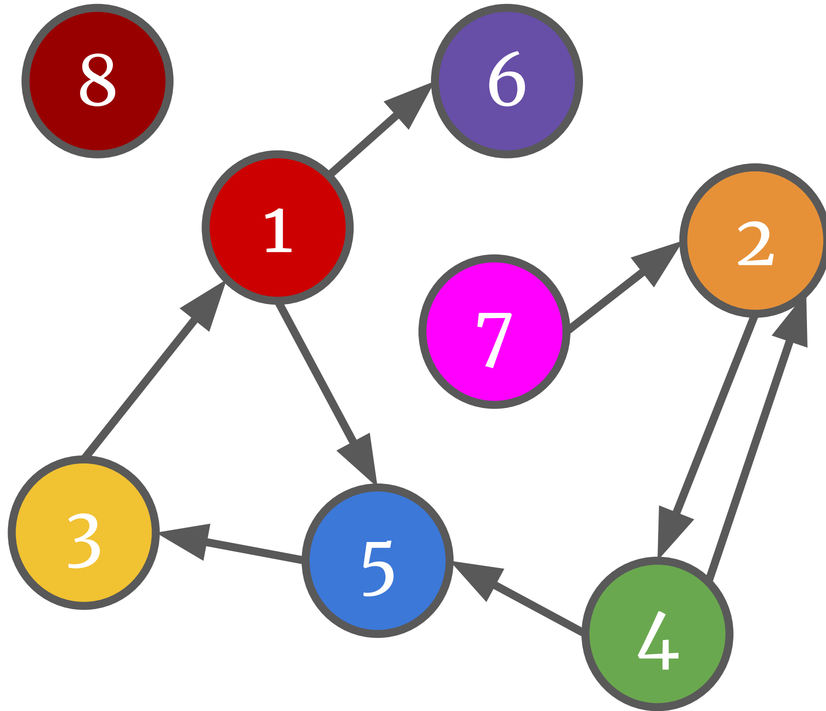
- Note: does not imply it can reach every other node.

Node 6 is a **sink** since it has no outgoing edges.

- Note: does not imply every other node can reach it.

(Node 8 is trivially a source and a sink.)

Similar representations



1: [5, 6]

2: [4]

3: [1]

4: [2, 5]

5: [3]

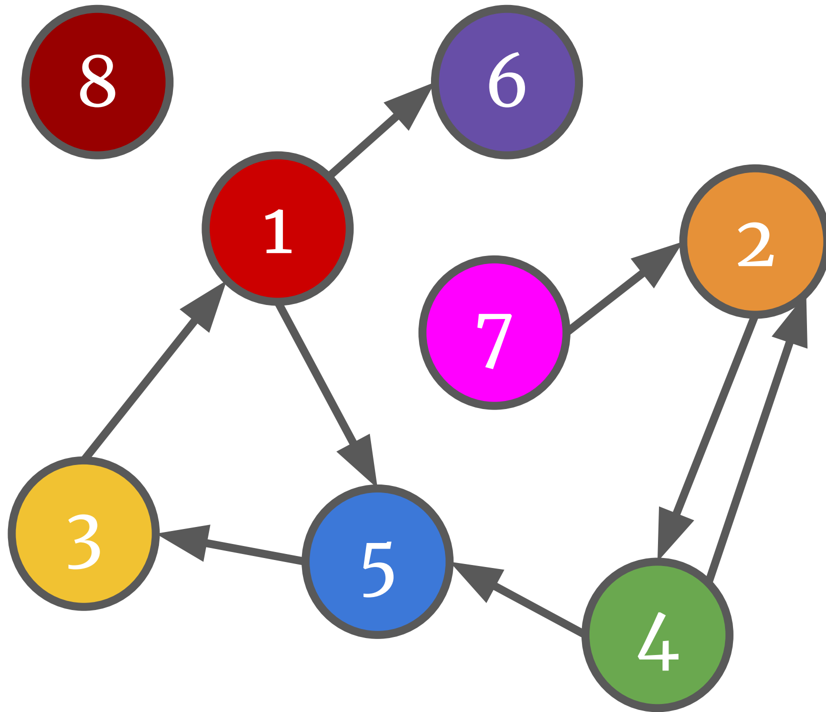
6: []

7: [2]

8: []

The adjacency list is not symmetric now!

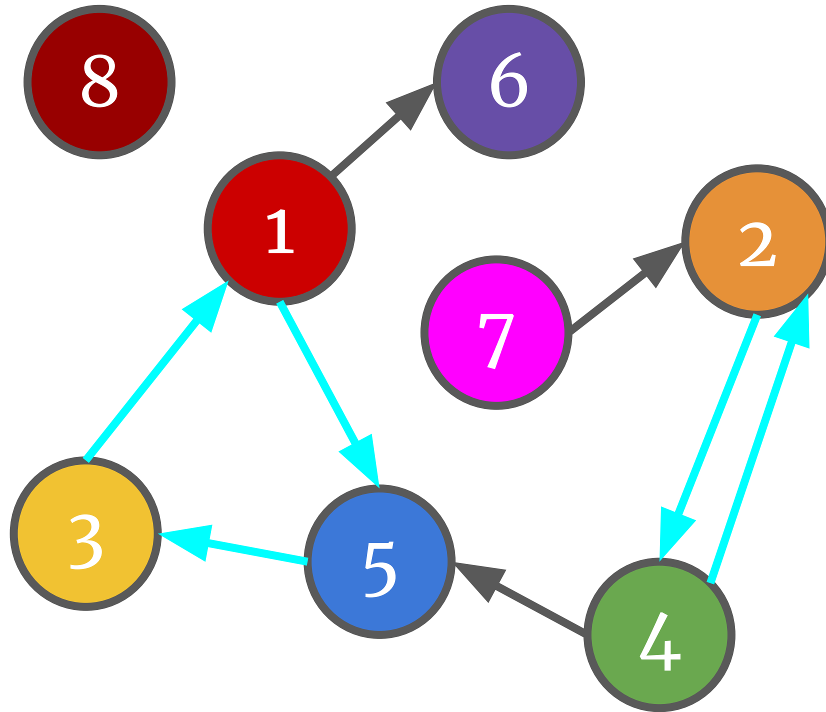
Similar representations



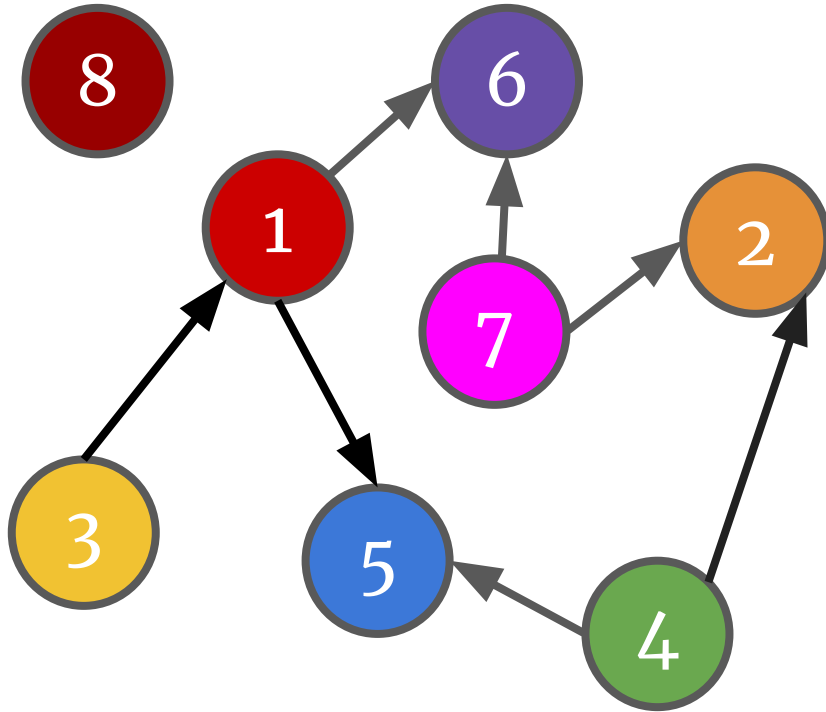
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

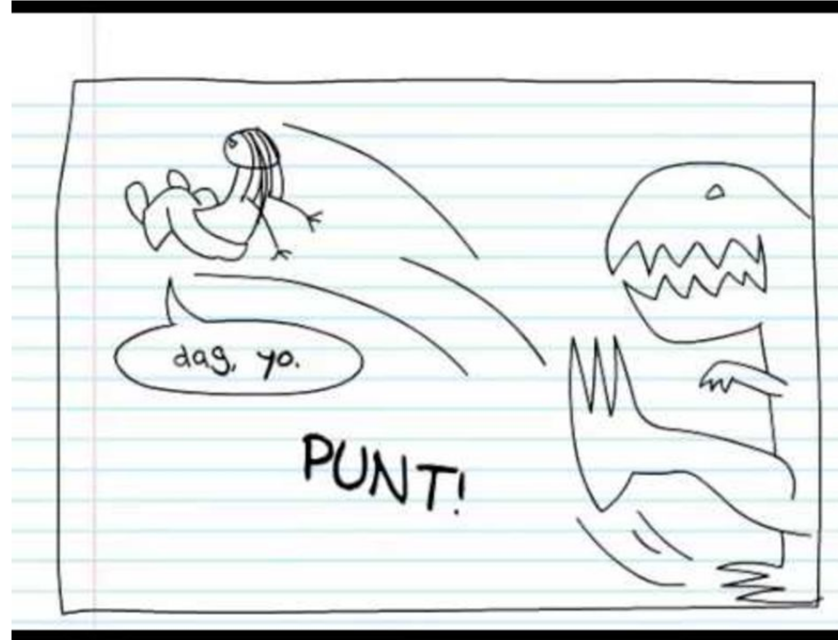
The adjacency matrix is not symmetric now!

Now cycles are directed

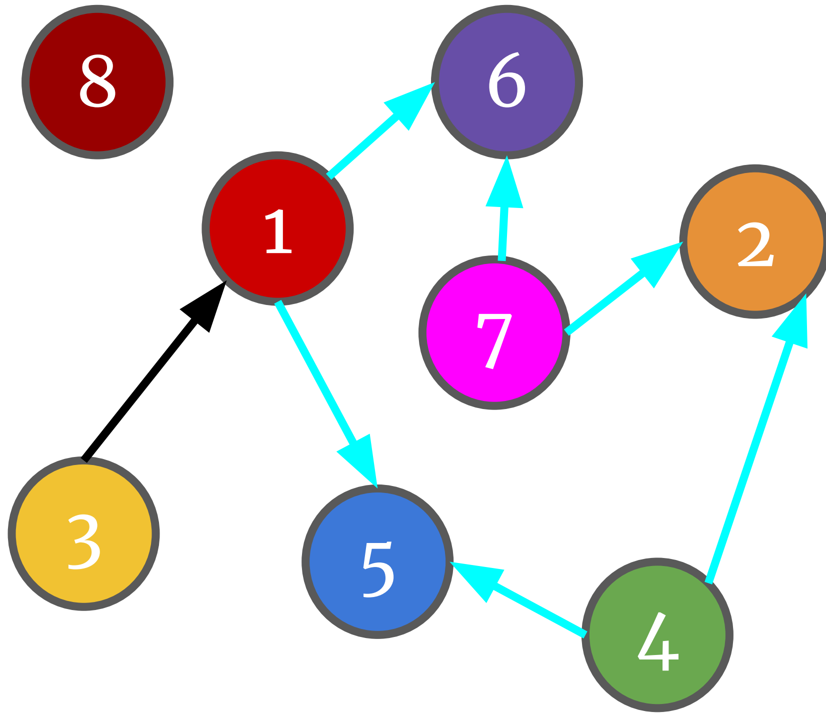


A dag is a directed acyclic graph.



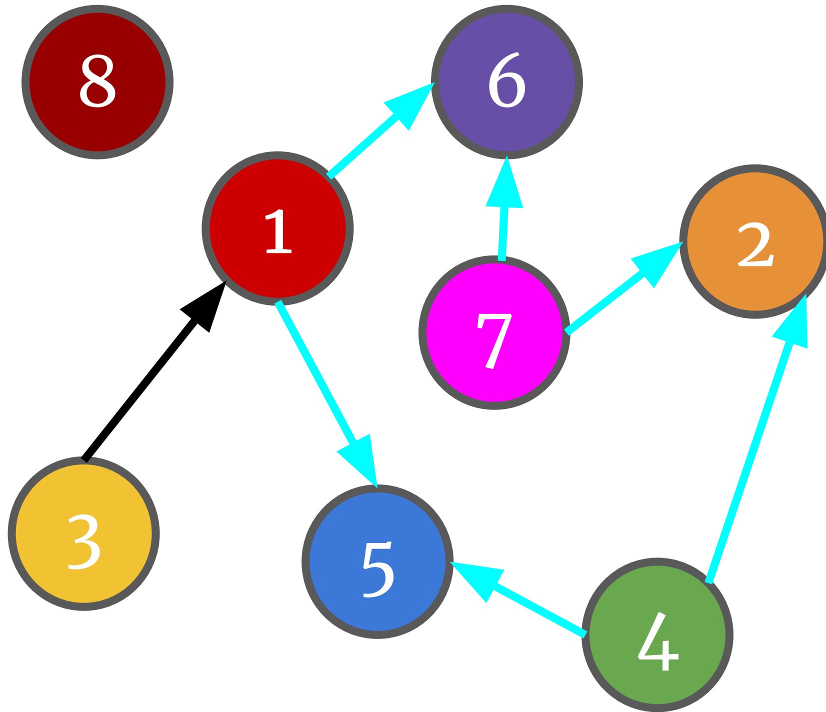


A dag is a directed acyclic graph.



Wait, isn't this a cycle?

A dag is a directed acyclic graph.



Wait, isn't this a cycle?

Nope, it's not a directed cycle, so it doesn't count as one in directed graph land!

File dependency graph

cxf-tools-wsdlto org.apache.cxf 2.4.0-SNAPSHOT
cxf-java2ws-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt org.apache.cxf 2.4.0-SNAPSHOT
cxf-parent org.apache.cxf 2.4.0-SNAPSHOT
cxf-archetypes org.apache.cxf 2.4.0-SNAPSHOT
cxf-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools org.apache.cxf 2.4.0-SNAPSHOT
cxf-maven-plugins org.apache.cxf 2.4.0-SNAPSHOT
cxf-jaxws-javafirst org.apache.cxf.archetype 2.4.0-SNAPSHOT
cxf-integration org.apache.cxf 2.4.0-SNAPSHOT
cxf org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-javato org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-frontend-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-java2ws org.apache.cxf 2.4.0-SNAPSHOT
cxf-corbatools-maven-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management-web org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-test org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxrs org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-security org.apache.cxf 2.4.0-SNAPSHOT

cxf-codegen-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-misctools org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-xmlbeans org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-aegis org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-js org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jca org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-sdo org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-object org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-policy org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-wsdlto-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-testsupport org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-jetty org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-osgi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-rm org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-wsdl-validator-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-simple org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-addr org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-validator org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jms org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-xml org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-soap org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-local org.apache.cxf 2.4.0-SNAPSHOT

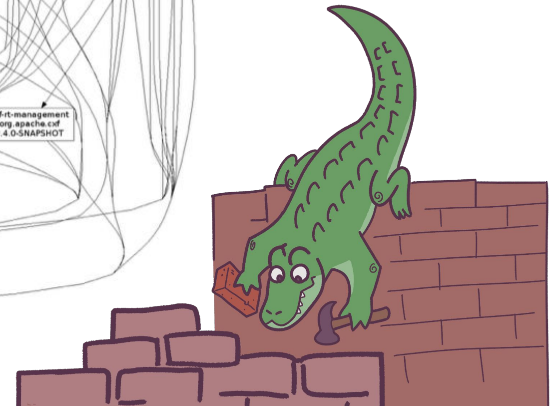
cxf-tools-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-coloc org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jbi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management org.apache.cxf 2.4.0-SNAPSHOT

cxf-wstx-msv-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-xerces-xsd-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-testutils org.apache.cxf 2.4.0-SNAPSHOT

cxf-api org.apache.cxf 2.4.0-SNAPSHOT

cxf-common-schemas org.apache.cxf 2.4.0-SNAPSHOT
cxf-common-utilities org.apache.cxf 2.4.0-SNAPSHOT

Apache CXF



File dependency graph

This had better be a DAG!!!

cxf-tools-wsdlto org.apache.cxf 2.4.0-SNAPSHOT
cxf-java2ws-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt org.apache.cxf 2.4.0-SNAPSHOT
cxf-parent org.apache.cxf 2.4.0-SNAPSHOT
cxf-archetypes org.apache.cxf 2.4.0-SNAPSHOT
cxf-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools org.apache.cxf 2.4.0-SNAPSHOT
cxf-maven-plugins org.apache.cxf 2.4.0-SNAPSHOT
cxf-jaxws-javafirst org.apache.cxf.archetype 2.4.0-SNAPSHOT
cxf-integration org.apache.cxf 2.4.0-SNAPSHOT
cxf org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-javato org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-frontend-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-java2ws org.apache.cxf 2.4.0-SNAPSHOT
cxf-corbatools-maven-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management-web org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-test org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxrs org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-security org.apache.cxf 2.4.0-SNAPSHOT

cxf-codegen-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-misctools org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-xmlbeans org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-aegis org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-js org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jca org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-sdo org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-object org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-policy org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-wsdlto-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-testsupport org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-jetty org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-osgi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-rm org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-wsdl-validator-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-simple org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-addr org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-validator org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jms org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-xml org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-soap org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-local org.apache.cxf 2.4.0-SNAPSHOT

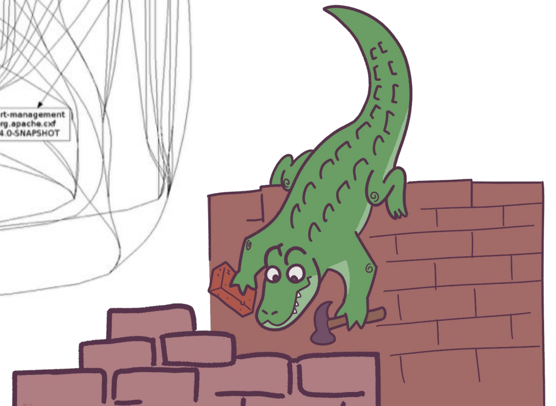
cxf-tools-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-coloc org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jbi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management org.apache.cxf 2.4.0-SNAPSHOT

cxf-wstx-msv-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-xerces-xsd-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-testutils org.apache.cxf 2.4.0-SNAPSHOT

cxf-api org.apache.cxf 2.4.0-SNAPSHOT

cxf-common-schemas org.apache.cxf 2.4.0-SNAPSHOT
cxf-common-utilities org.apache.cxf 2.4.0-SNAPSHOT

Apache CXF



File dependency graph

What order do we install these in?

cxf-tools-wsdlto org.apache.cxf 2.4.0-SNAPSHOT
cxf-java2ws-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt org.apache.cxf 2.4.0-SNAPSHOT
cxf-parent org.apache.cxf 2.4.0-SNAPSHOT
cxf-archetypes org.apache.cxf 2.4.0-SNAPSHOT
cxf-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools org.apache.cxf 2.4.0-SNAPSHOT
cxf-maven-plugins org.apache.cxf 2.4.0-SNAPSHOT
cxf-jaxws-javafirst org.apache.cxf.archetype 2.4.0-SNAPSHOT
cxf-integration org.apache.cxf 2.4.0-SNAPSHOT
cxf org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-javato org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-frontend-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-java2ws org.apache.cxf 2.4.0-SNAPSHOT
cxf-corbatools-maven-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management-web org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-test org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxrs org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-security org.apache.cxf 2.4.0-SNAPSHOT

cxf-codegen-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-misctools org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-xmlbeans org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-aegis org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-js org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jca org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-sdo org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-object org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-policy org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-wsdlto-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-testsupport org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-jetty org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-osgi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-rm org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-wsdl-validator-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-simple org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-addr org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-validator org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jms org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-xml org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-soap org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-local org.apache.cxf 2.4.0-SNAPSHOT

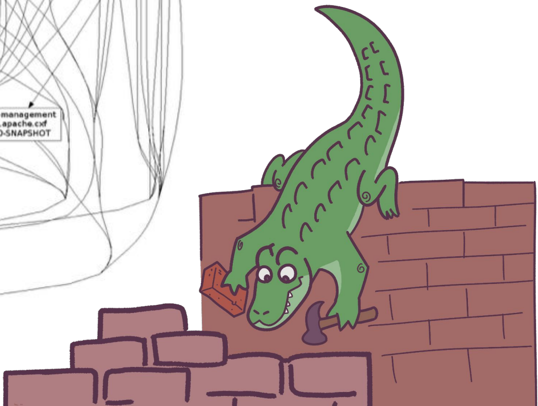
cxf-tools-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-coloc org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jbi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management org.apache.cxf 2.4.0-SNAPSHOT

cxf-wstx-msv-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-xerces-xsd-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-testutils org.apache.cxf 2.4.0-SNAPSHOT

cxf-api org.apache.cxf 2.4.0-SNAPSHOT

cxf-common-schemas org.apache.cxf 2.4.0-SNAPSHOT
cxf-common-utilities org.apache.cxf 2.4.0-SNAPSHOT

Apache CXF



File dependency graph

cxf-tools-wsdlto org.apache.cxf 2.4.0-SNAPSHOT
cxf-java2ws-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt org.apache.cxf 2.4.0-SNAPSHOT
cxf-parent org.apache.cxf 2.4.0-SNAPSHOT
cxf-archetypes org.apache.cxf 2.4.0-SNAPSHOT
cxf-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools org.apache.cxf 2.4.0-SNAPSHOT
cxf-maven-plugins org.apache.cxf 2.4.0-SNAPSHOT
cxf-jaxws-javafirst org.apache.cxf.archetype 2.4.0-SNAPSHOT
cxf-integration org.apache.cxf 2.4.0-SNAPSHOT
cxf org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-javato org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-frontend-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-java2ws org.apache.cxf 2.4.0-SNAPSHOT
cxf-corbatools-maven-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management-web org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-test org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-javascript org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxrs org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-security org.apache.cxf 2.4.0-SNAPSHOT

cxf-codegen-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-misctools org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-xmlbeans org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-aegis org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-js org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jca org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-sdo org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-object org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-corba org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-policy org.apache.cxf 2.4.0-SNAPSHOT
cxf-integration-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-tools-wsdlto-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-testsupport org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-jetty org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http-osgi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-jaxws org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-rm org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-jbi org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-wsdlto-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-wsdl-validator-plugin org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-http org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-frontend-simple org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-ws-addr org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-validator org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jms org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-xml org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-soap org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-local org.apache.cxf 2.4.0-SNAPSHOT

cxf-tools-common org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-bindings-coloc org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-databinding-jaxb org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-transports-jbi org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-management org.apache.cxf 2.4.0-SNAPSHOT

cxf-wstx-msv-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-xerces-xsd-validation org.apache.cxf 2.4.0-SNAPSHOT
cxf-rt-core org.apache.cxf 2.4.0-SNAPSHOT
cxf-testutils org.apache.cxf 2.4.0-SNAPSHOT

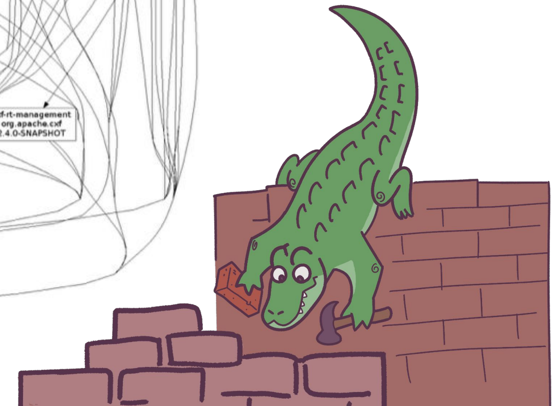
cxf-api org.apache.cxf 2.4.0-SNAPSHOT

cxf-common-schemas org.apache.cxf 2.4.0-SNAPSHOT
cxf-common-utilities org.apache.cxf 2.4.0-SNAPSHOT

Apache CXF

What order do we install these in?

Answer: use a package manager

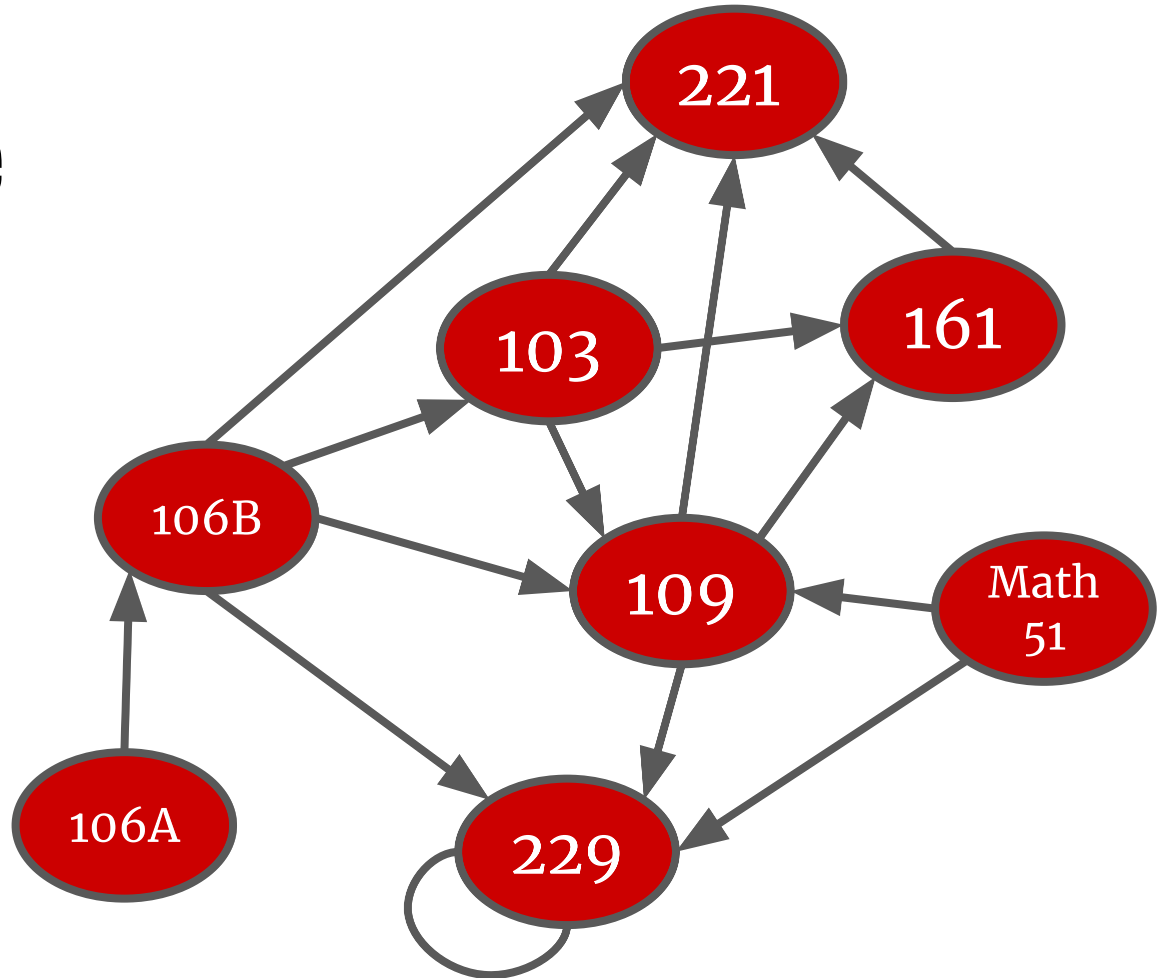


Indy's schedule advice

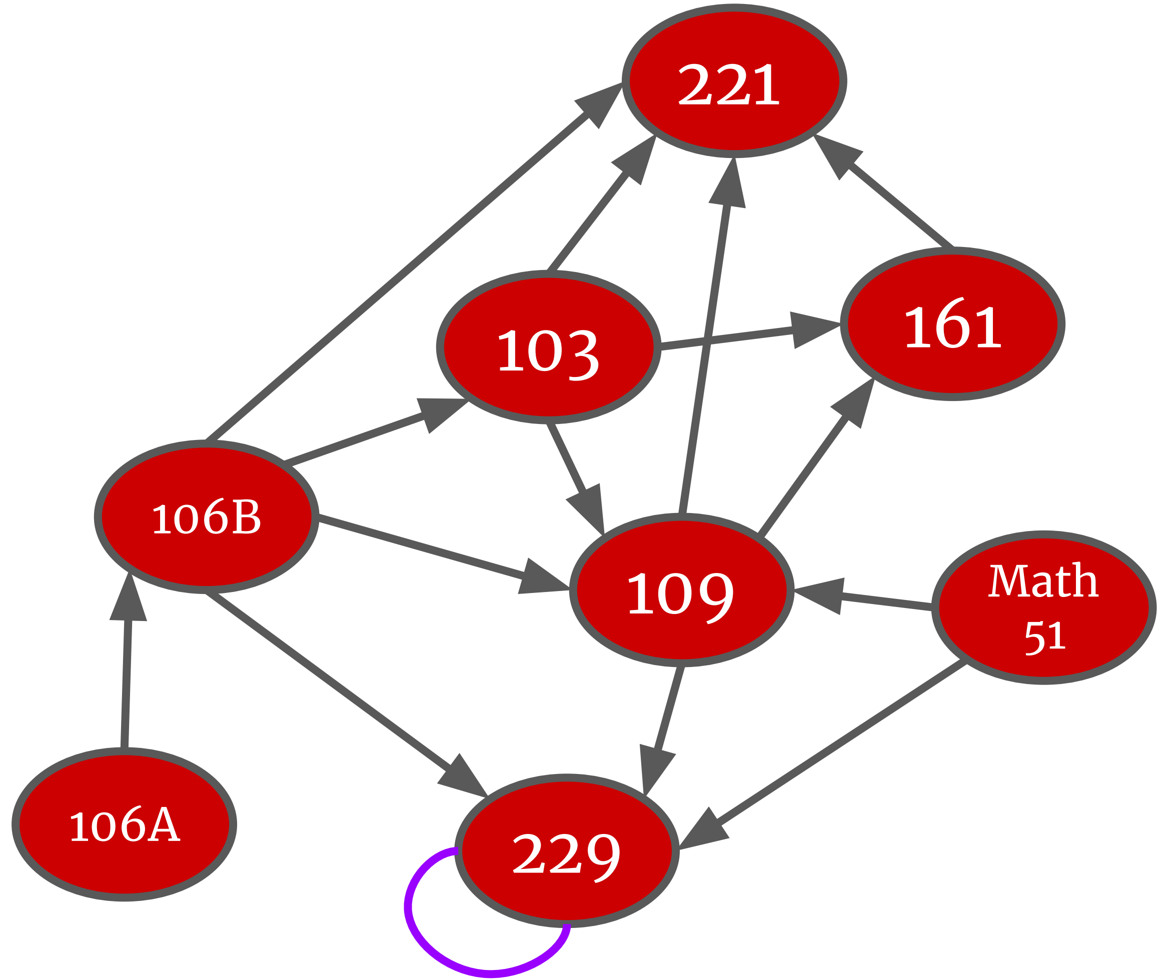
- Take CS229 ASAP
- Side hustle
- Drop out
- FIRE (Financial Independence, Retire Early)
- Possibly realize – say, while on the toilet one day in your mansion in Atherton – that your life is empty

Let's pretend people take the prereqs seriously.

In what order should we take these classes?

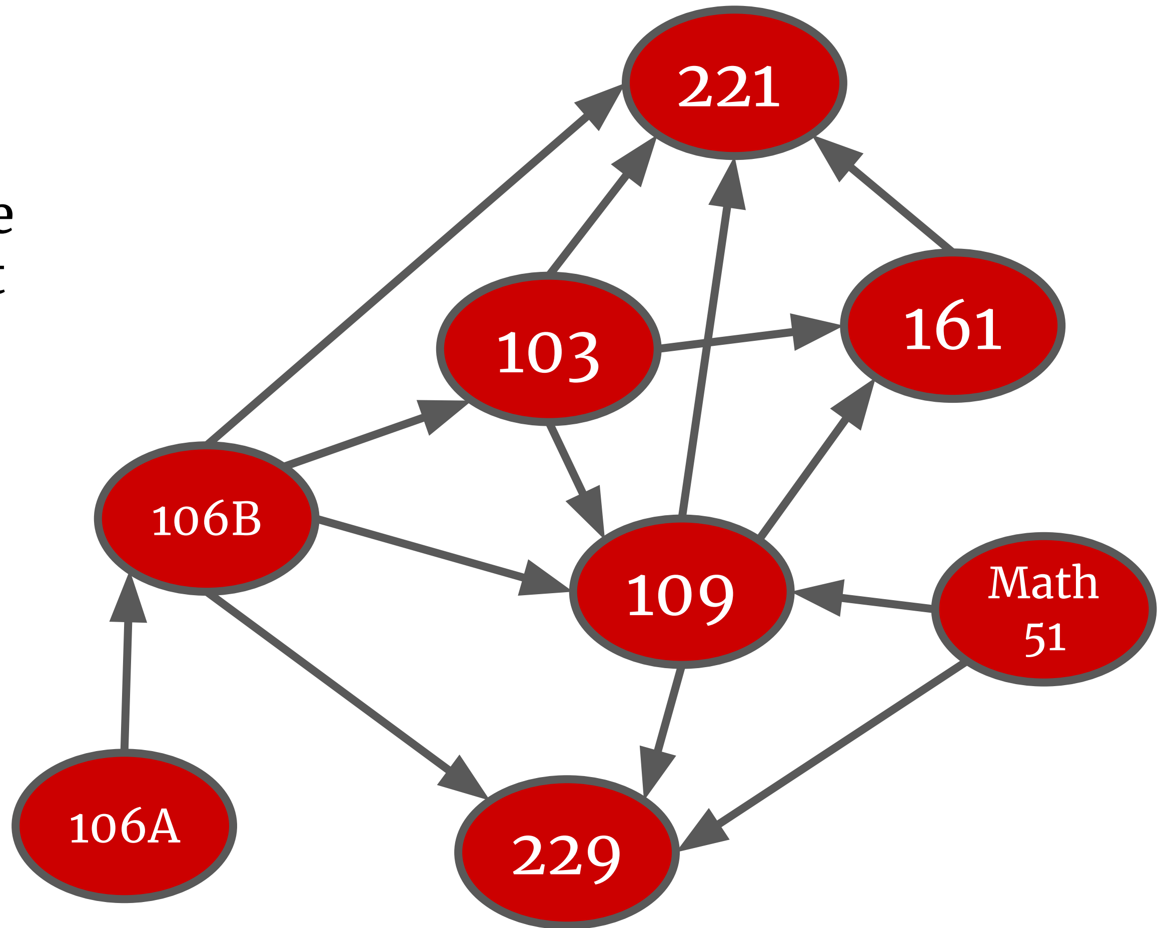


CS229 is not actually a prereq for itself, but it can feel that way!



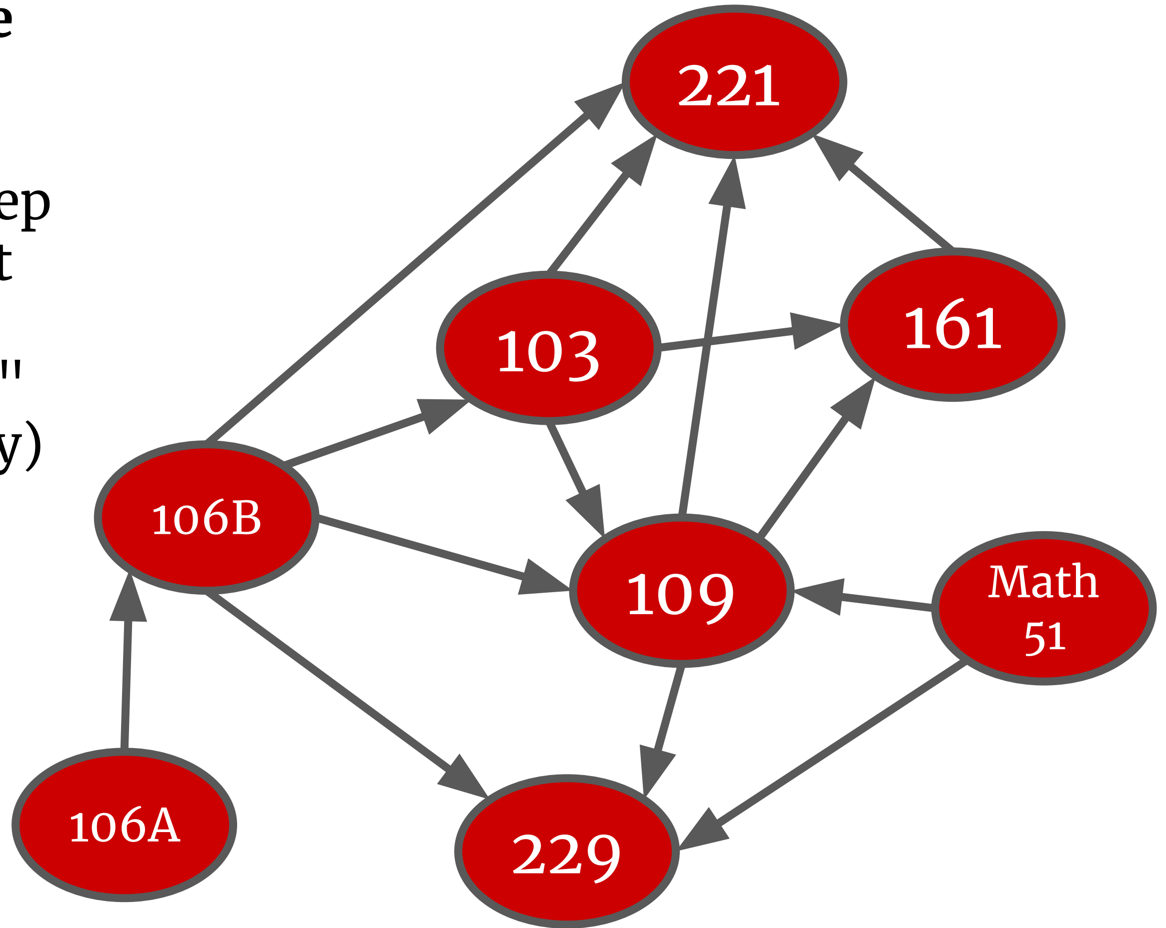
Hopefully this is actually a DAG!

- I think there's like a 10% chance that there is a cycle among the AI classes at Stanford...



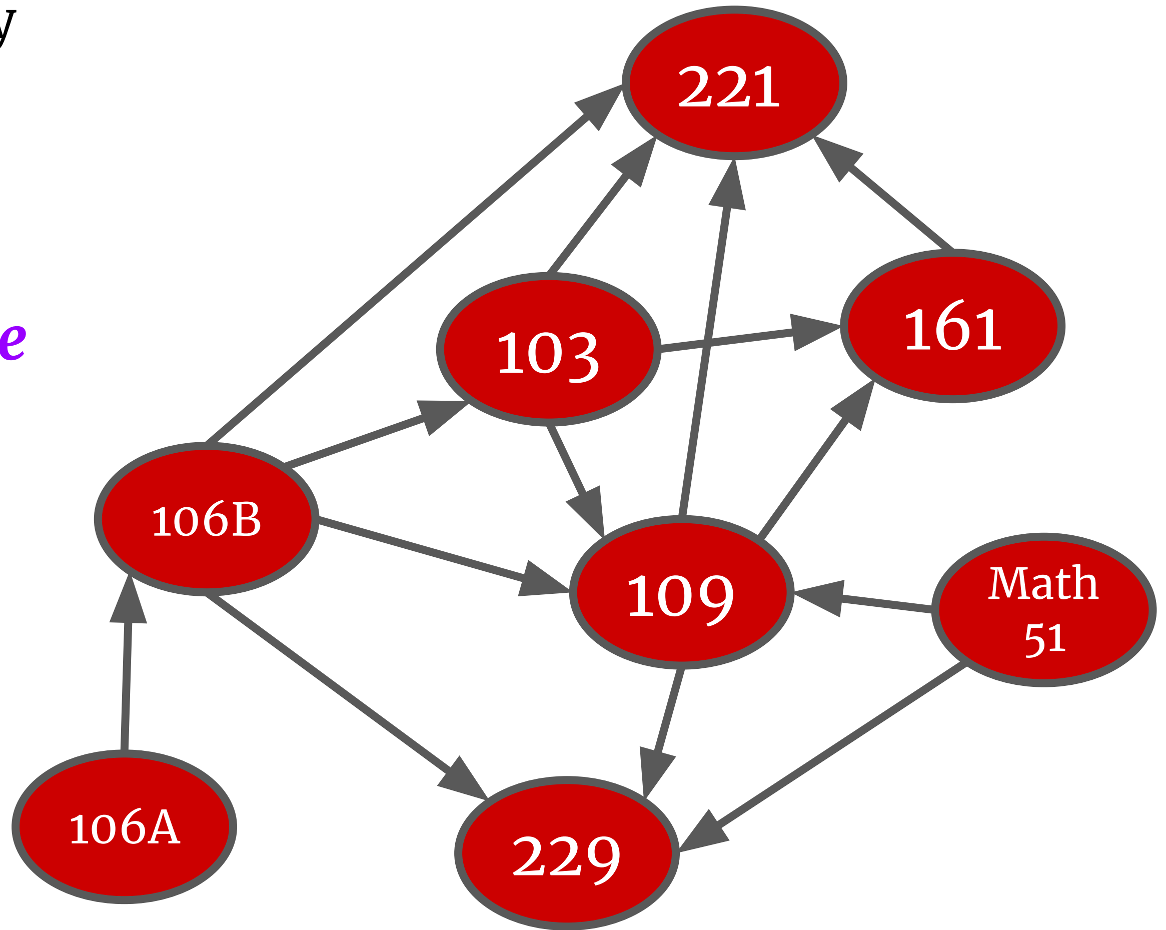
What order should we take the classes in?

Let's do a DFS, but keep track of when we start and finish each node. ("Visited" and "Done" in our old terminology)

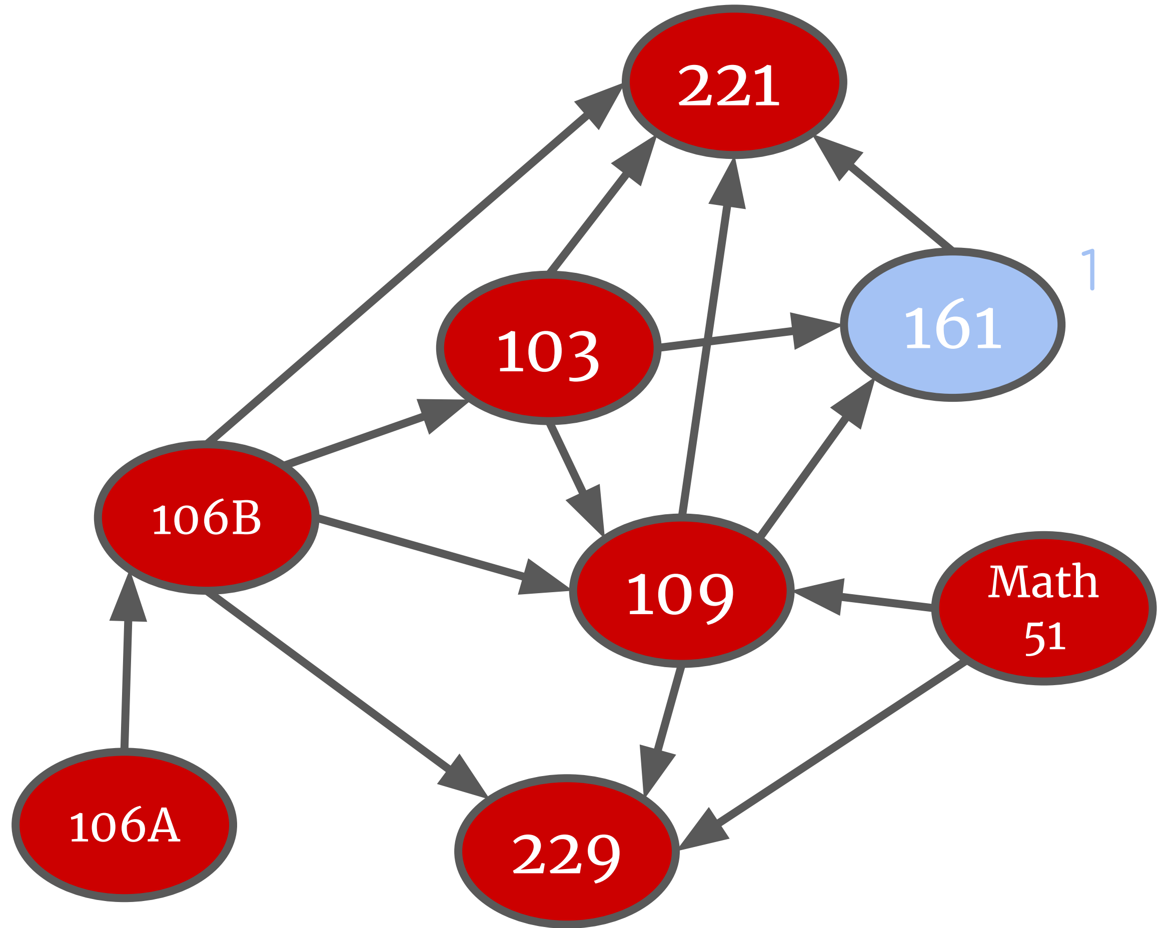


We'll pick an arbitrary start point for our search.

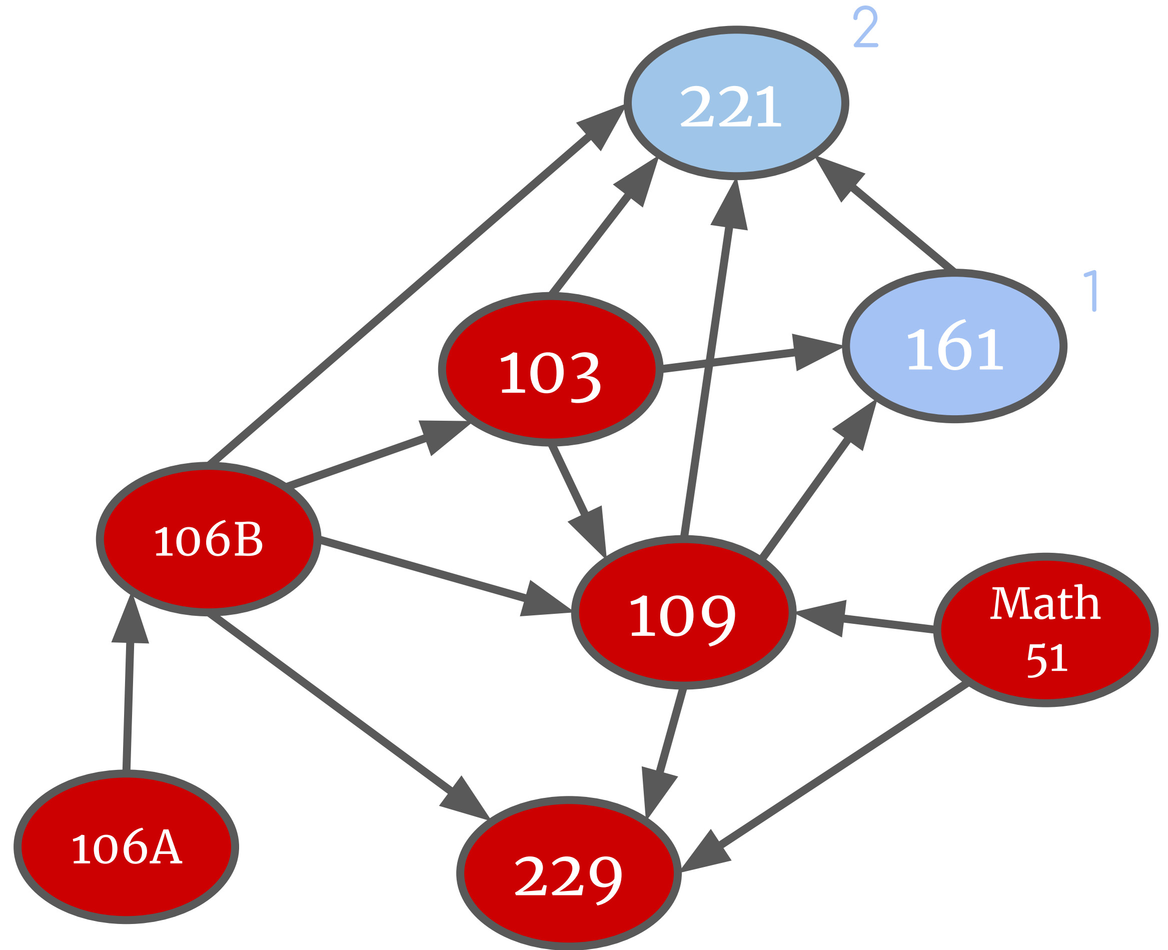
(That does NOT necessarily mean we are taking this class first!)



light blue = started

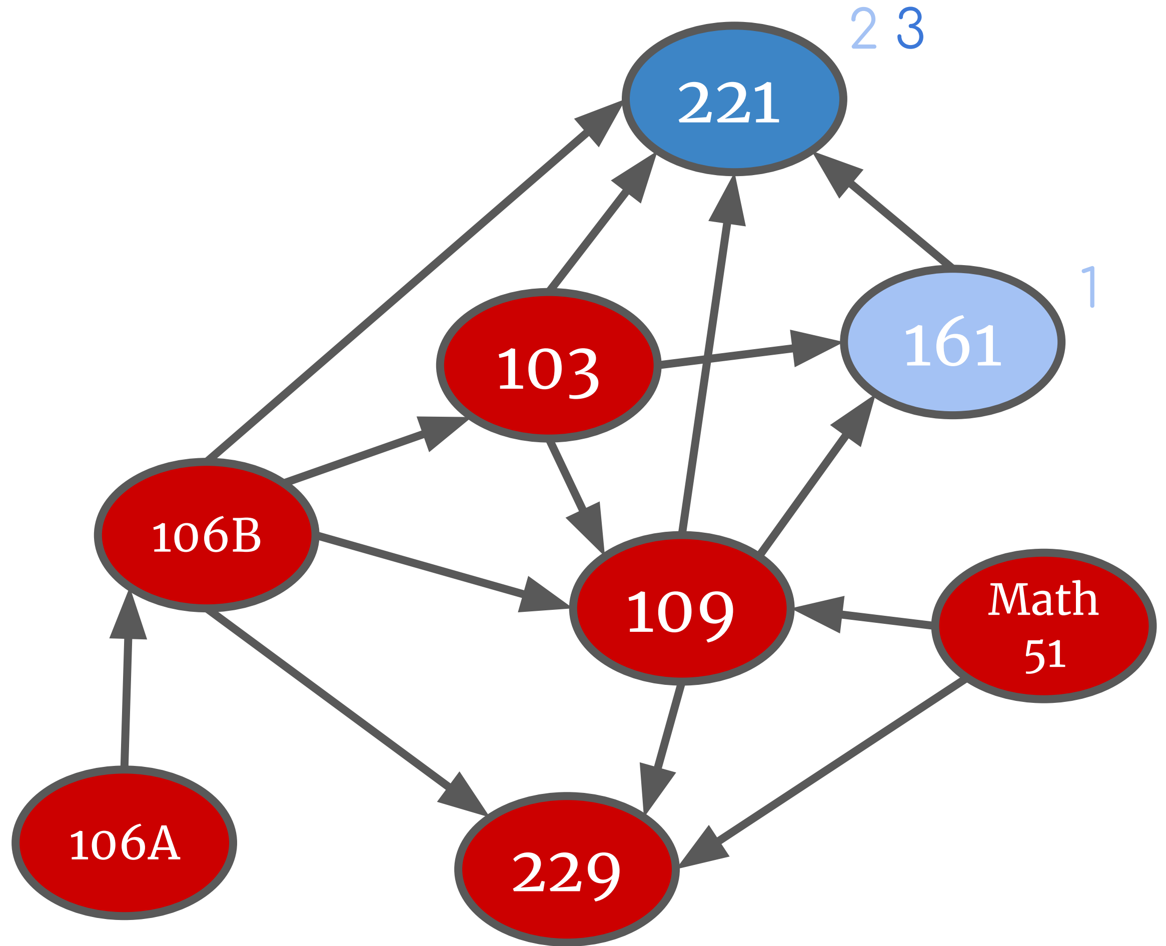


light blue = started



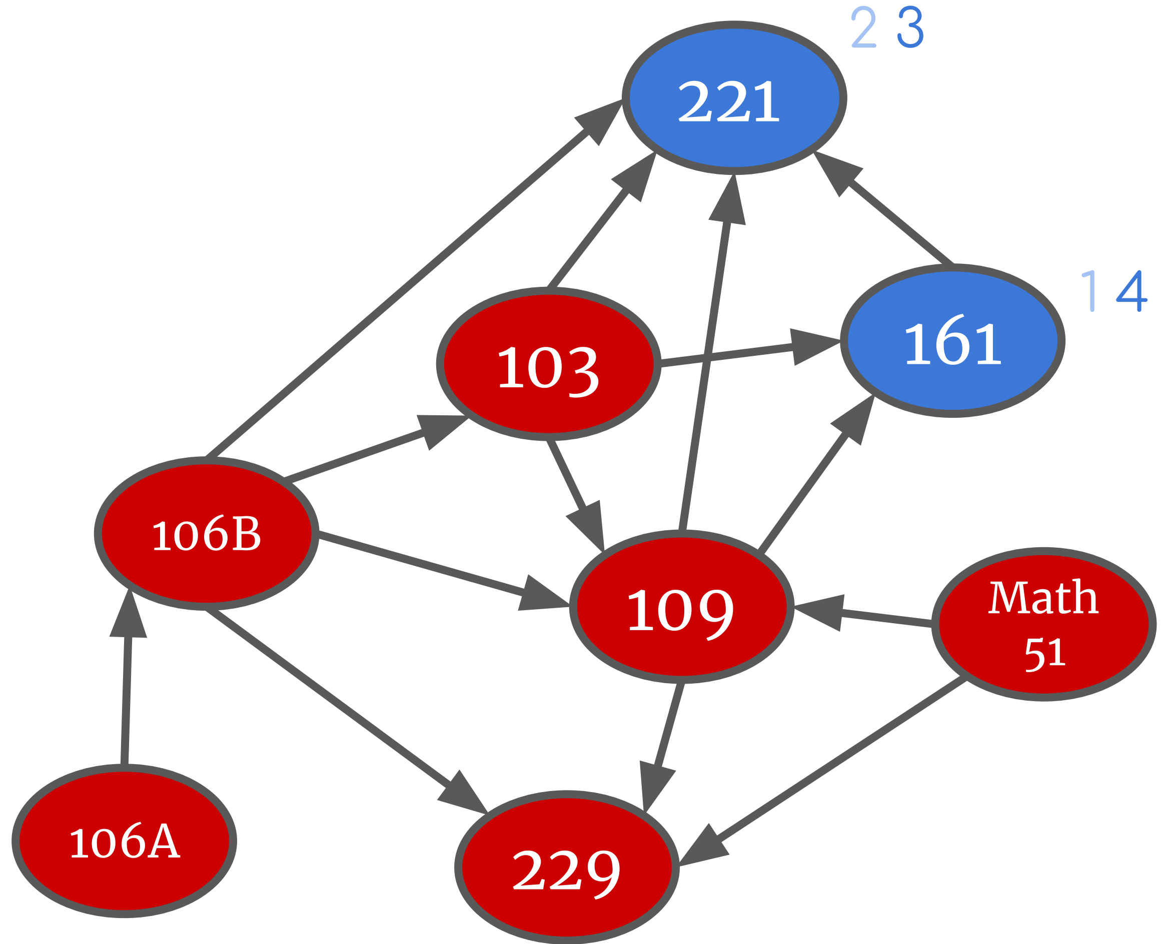
light blue = started

dark blue = done



light blue = started

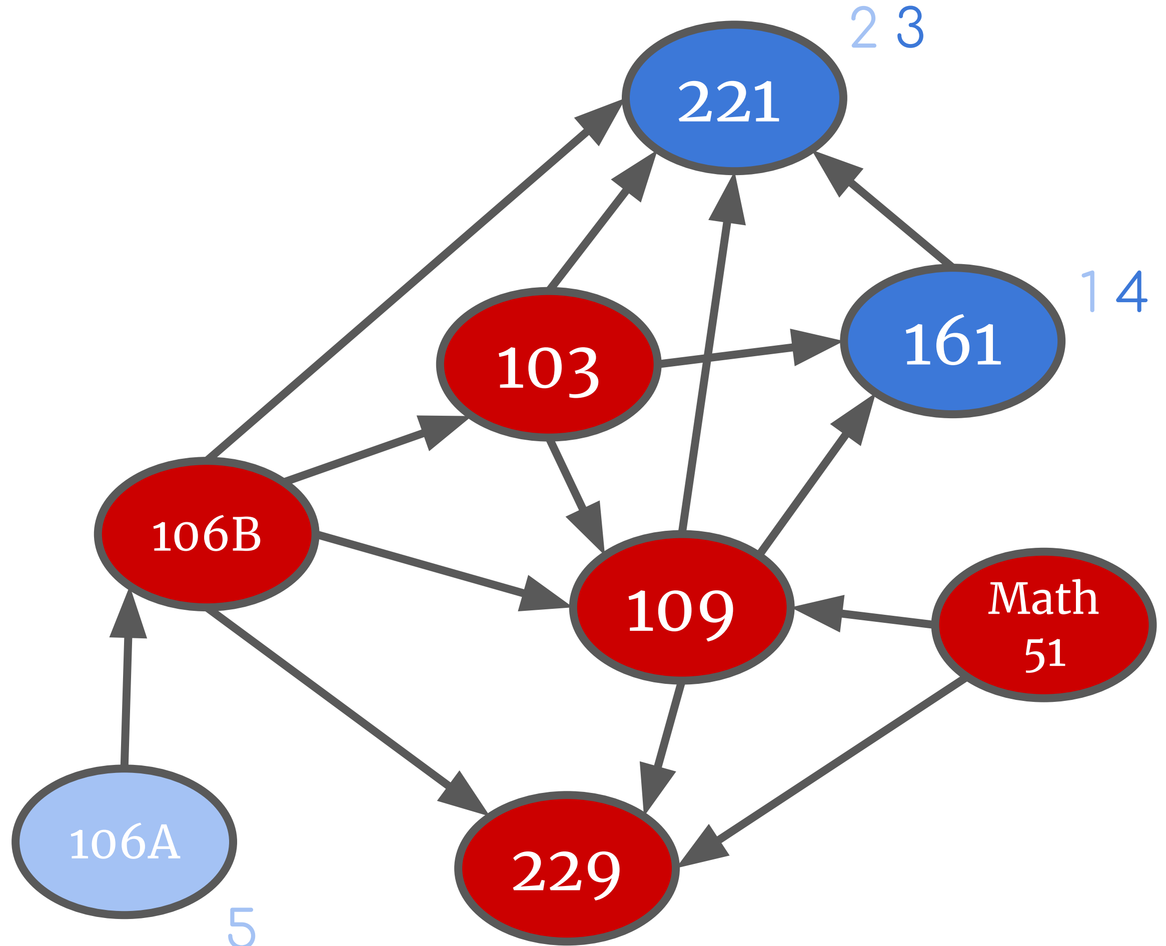
dark blue = done



light blue = started

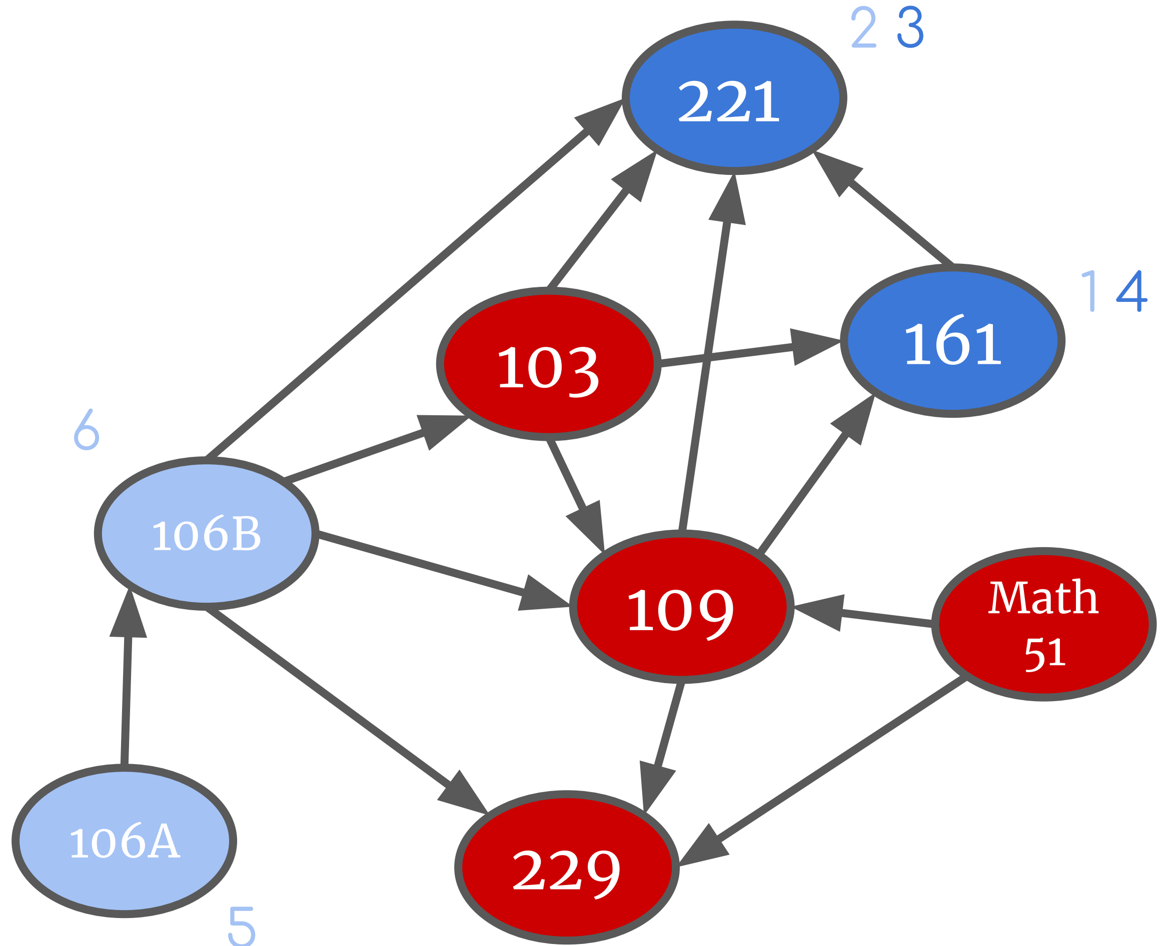
dark blue = done

Restart somewhere arbitrary that isn't done. (We might have a tiebreaker rule, or maybe not)



light blue = started

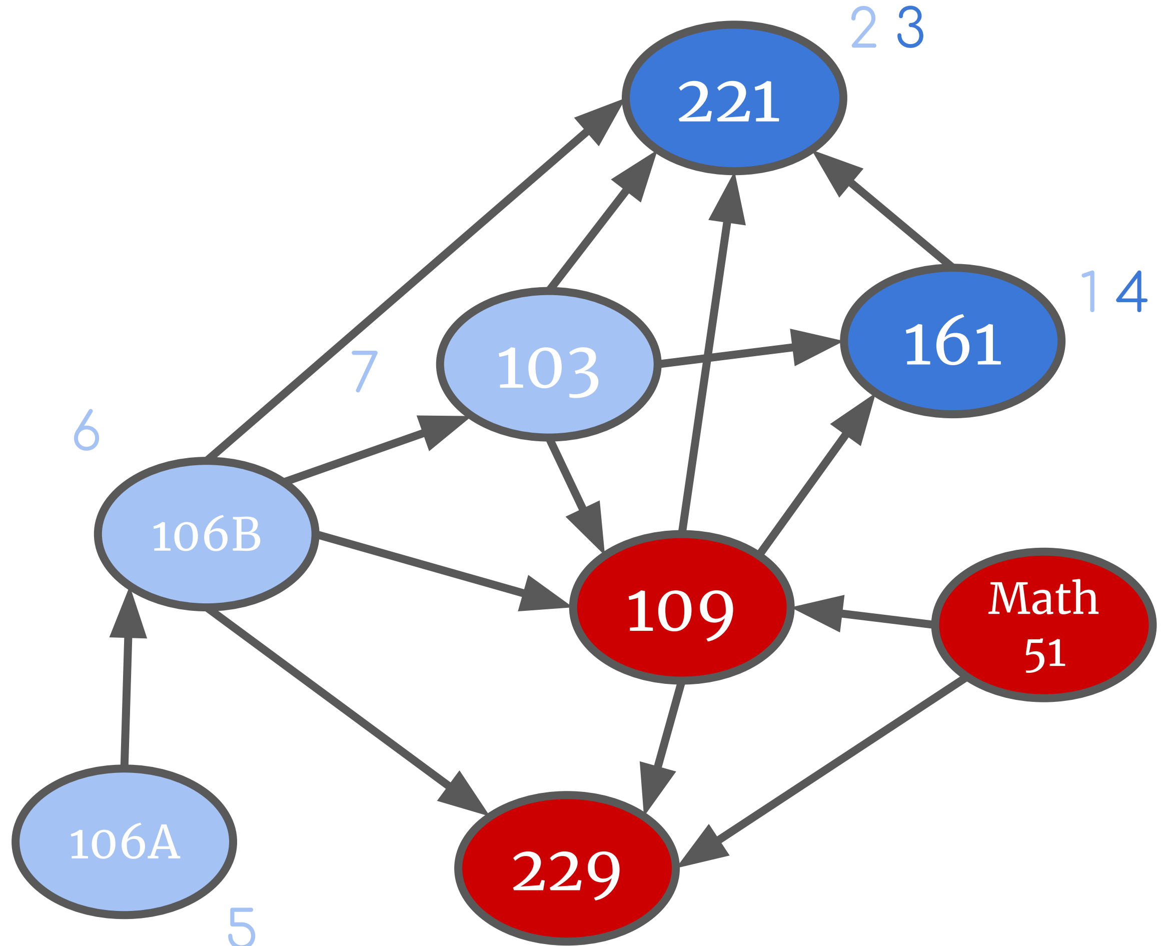
dark blue = done



light blue = started

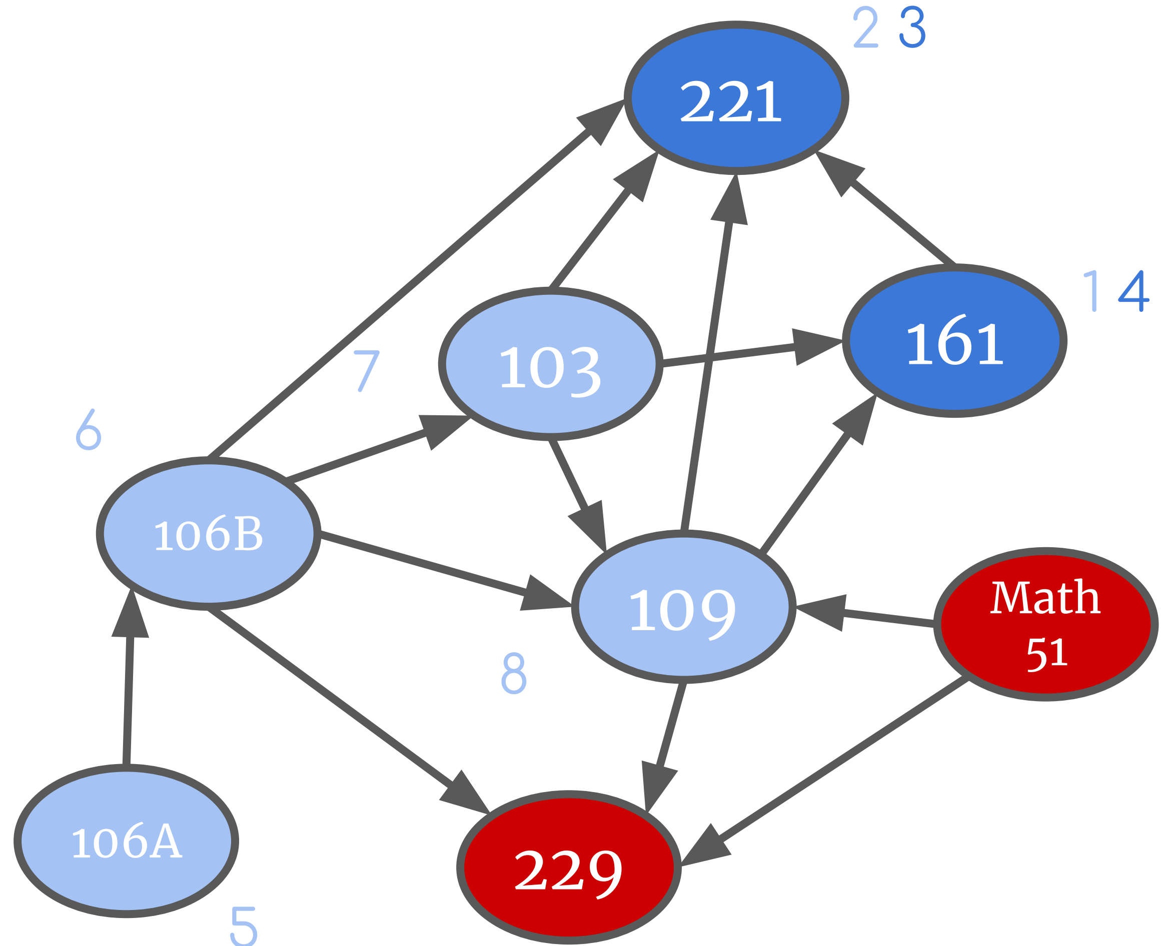
dark blue = done

Let's say we break ties by course number (though it turns out not to matter)



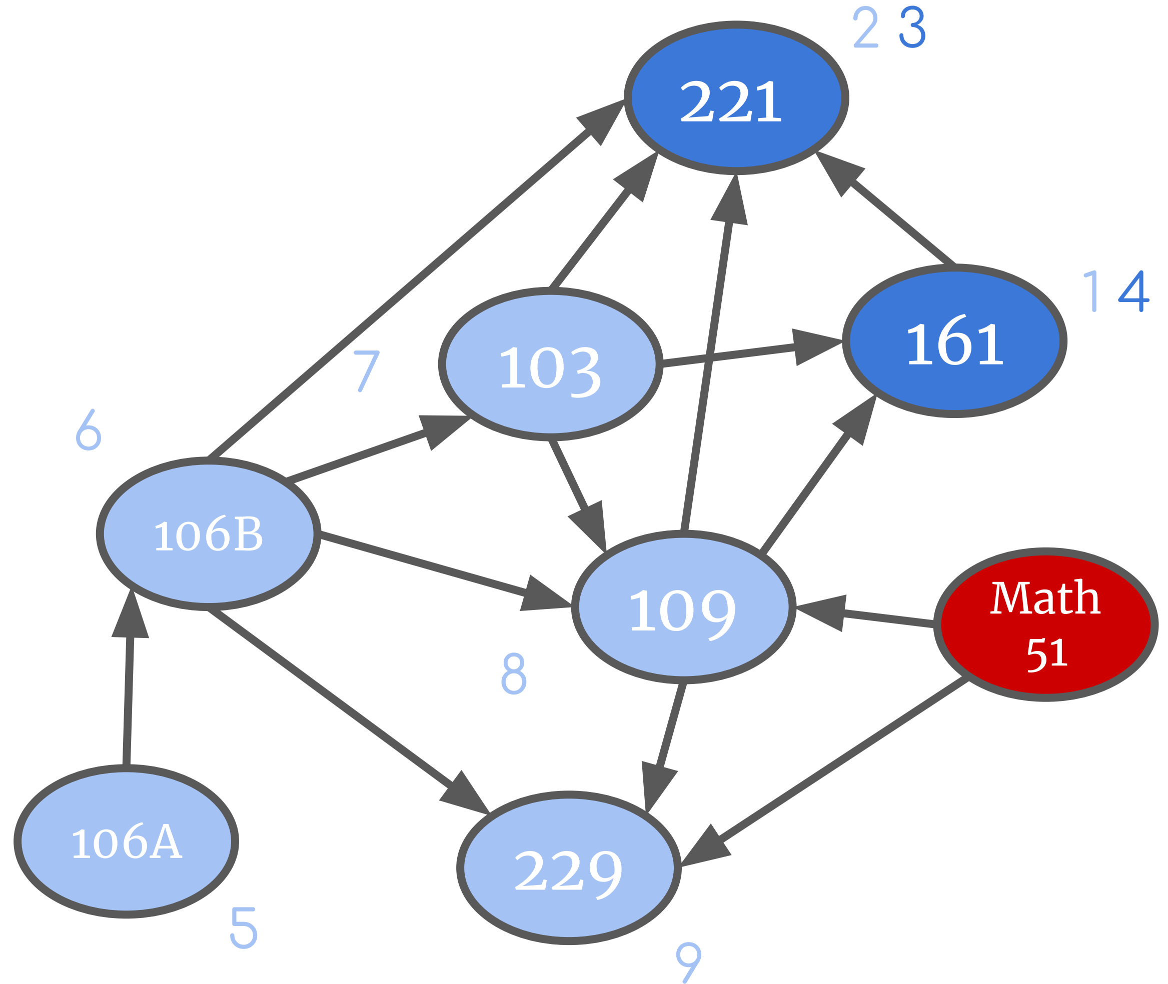
light blue = started

dark blue = done



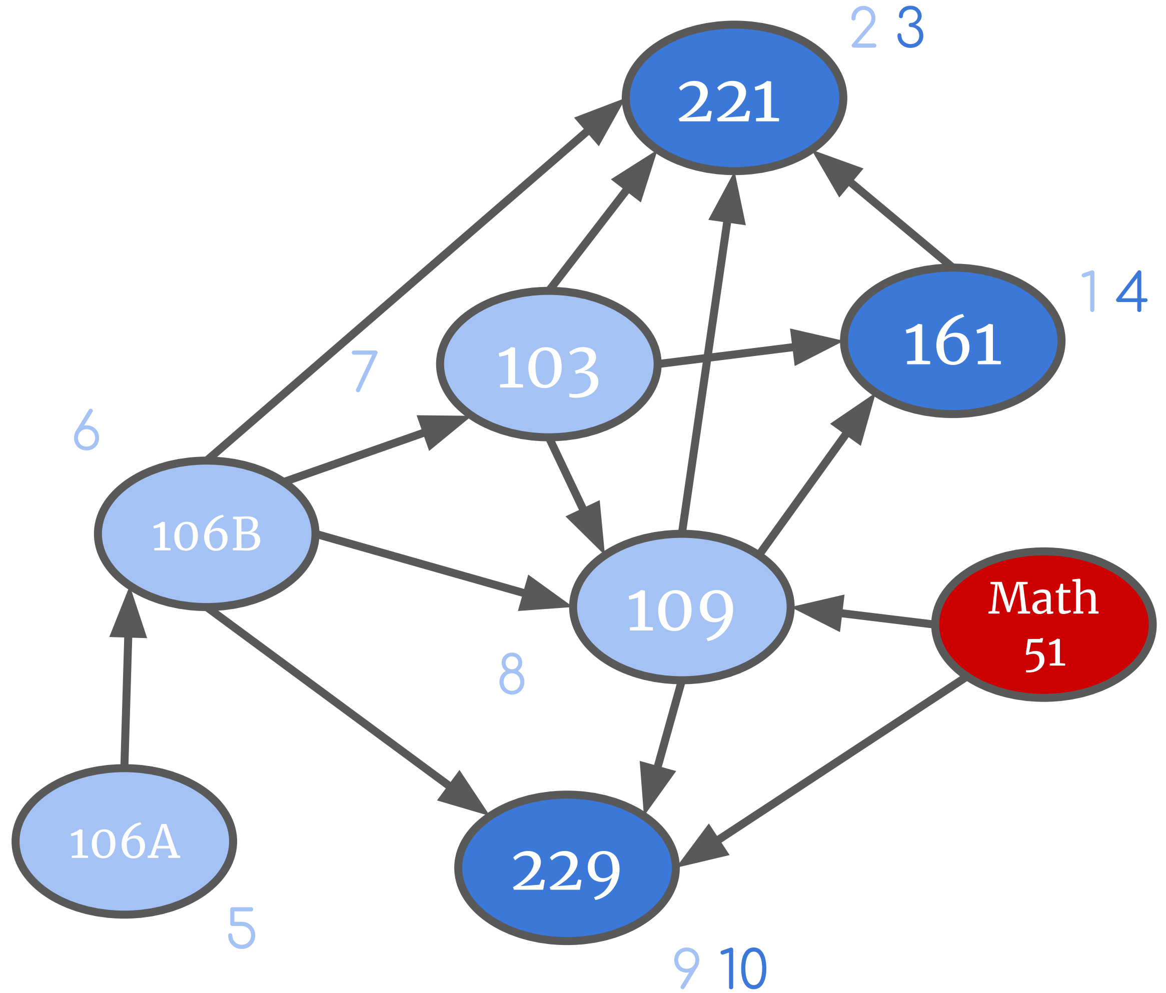
light blue = started

dark blue = done



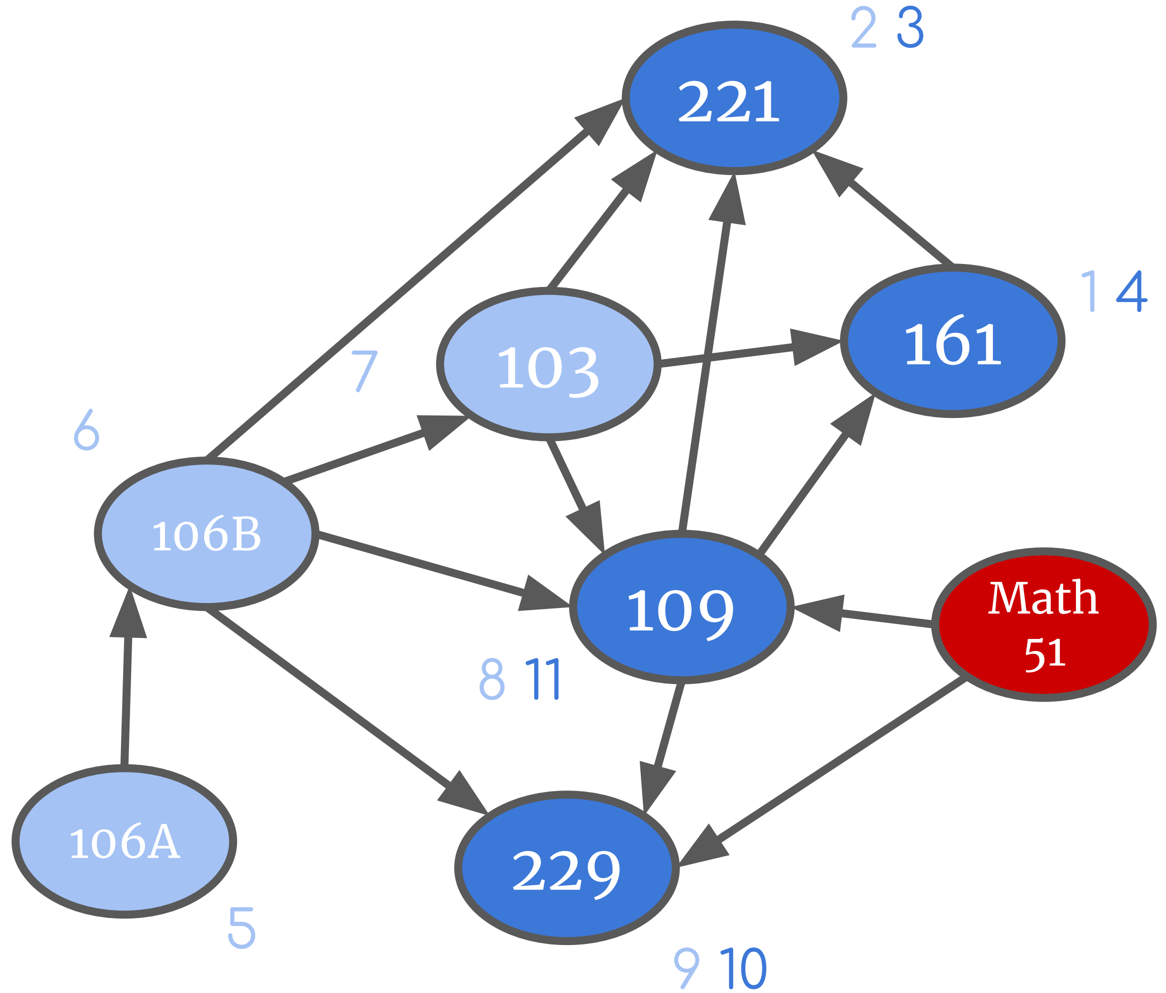
light blue = started

dark blue = done



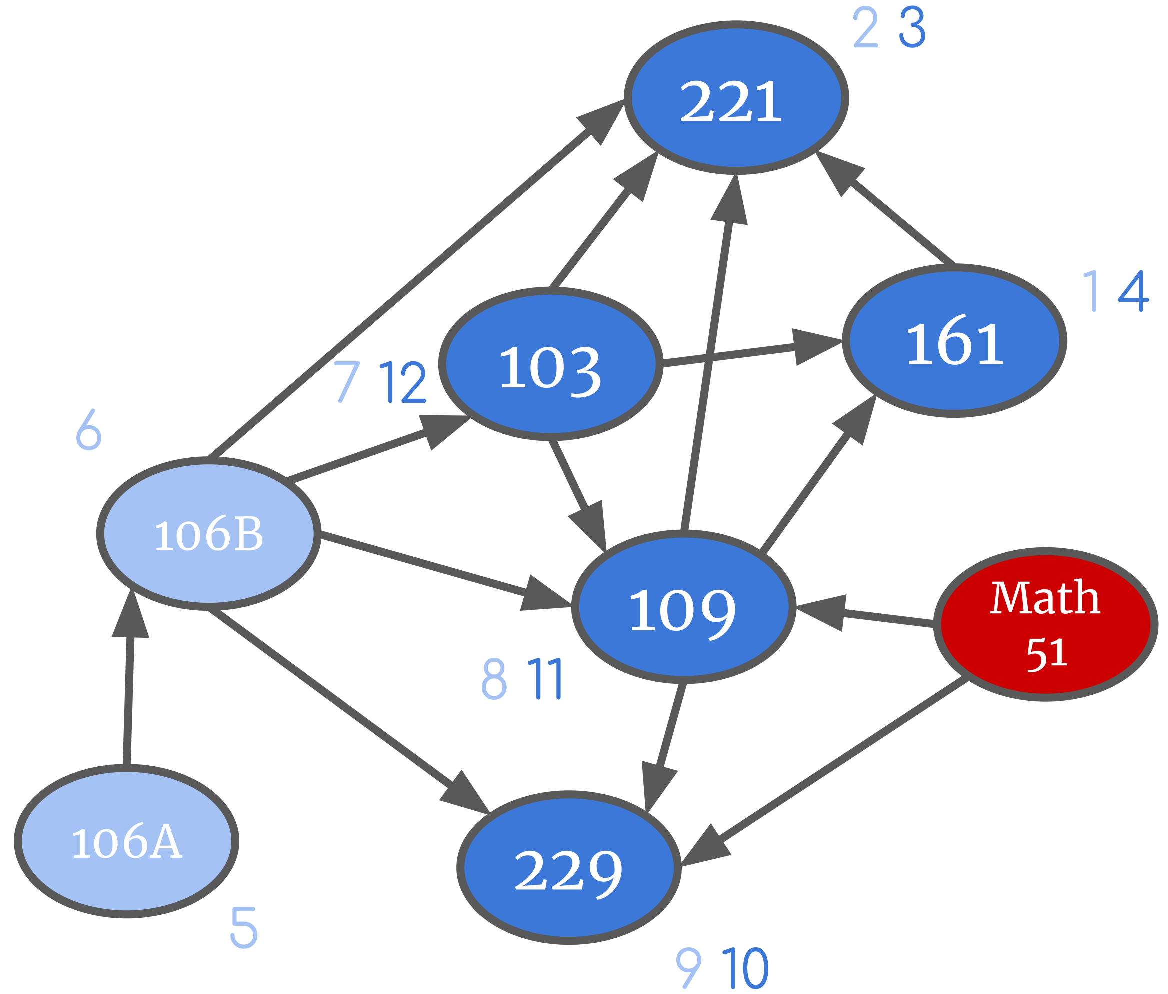
light blue = started

dark blue = done



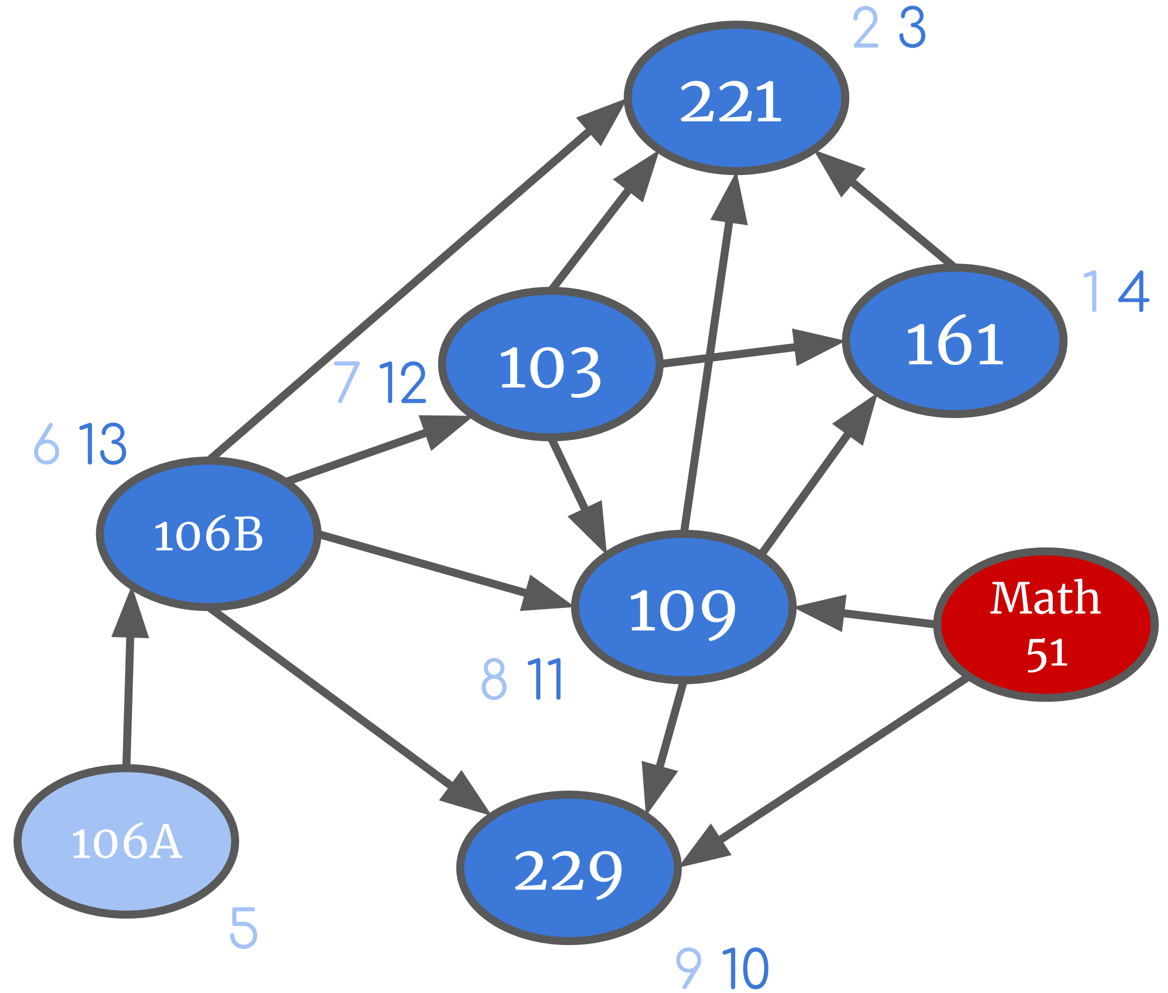
light blue = started

dark blue = done



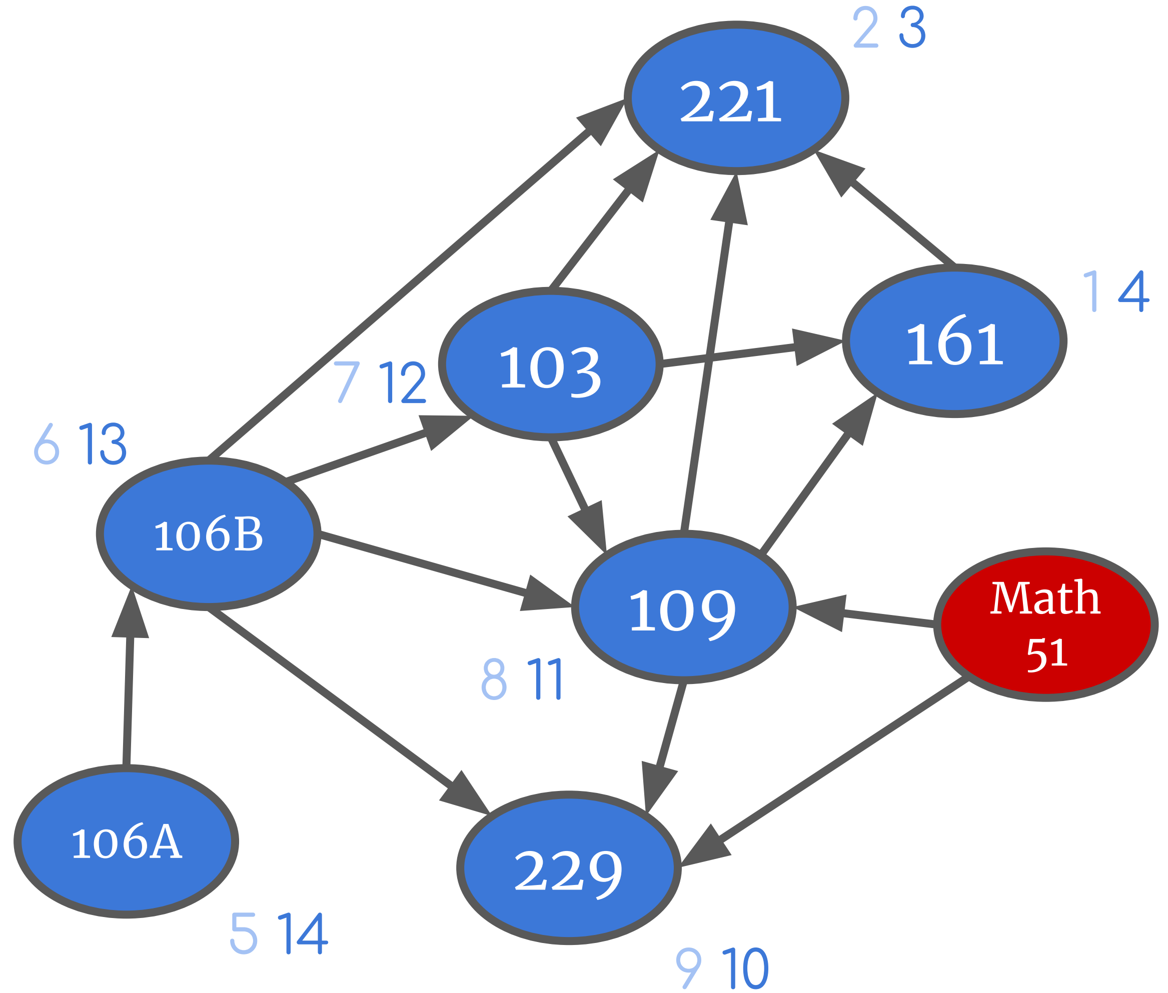
light blue = started

dark blue = done



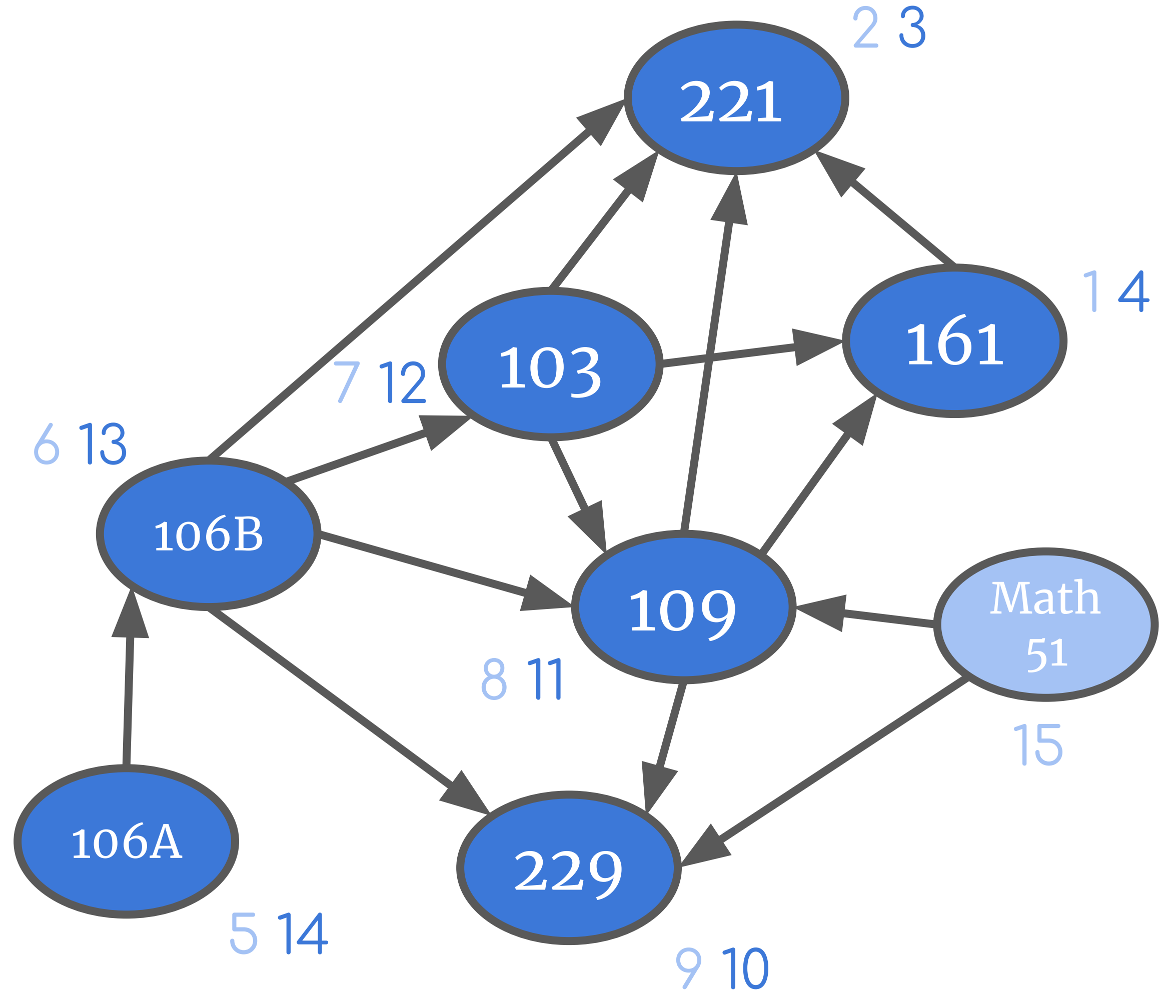
light blue = started

dark blue = done



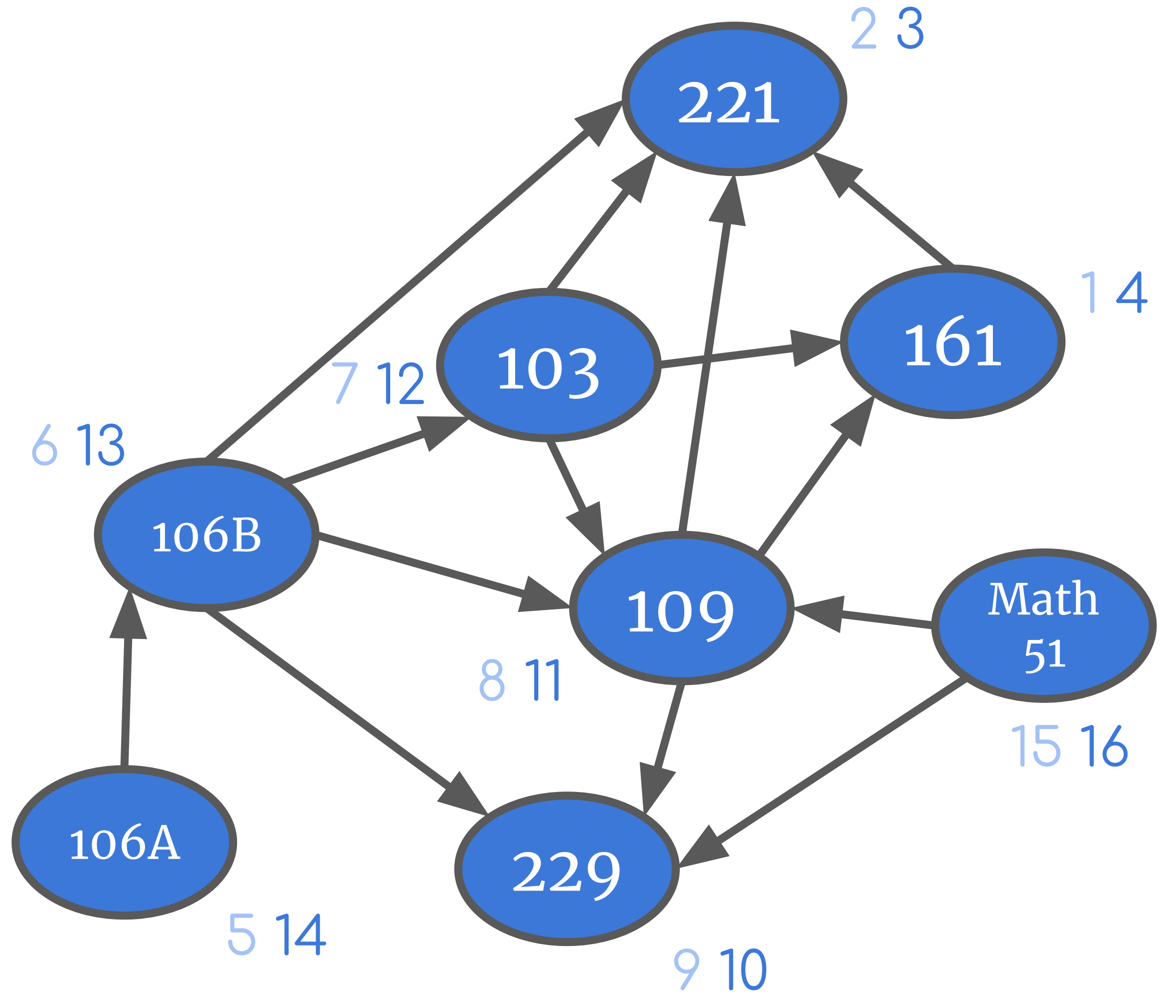
light blue = started

dark blue = done

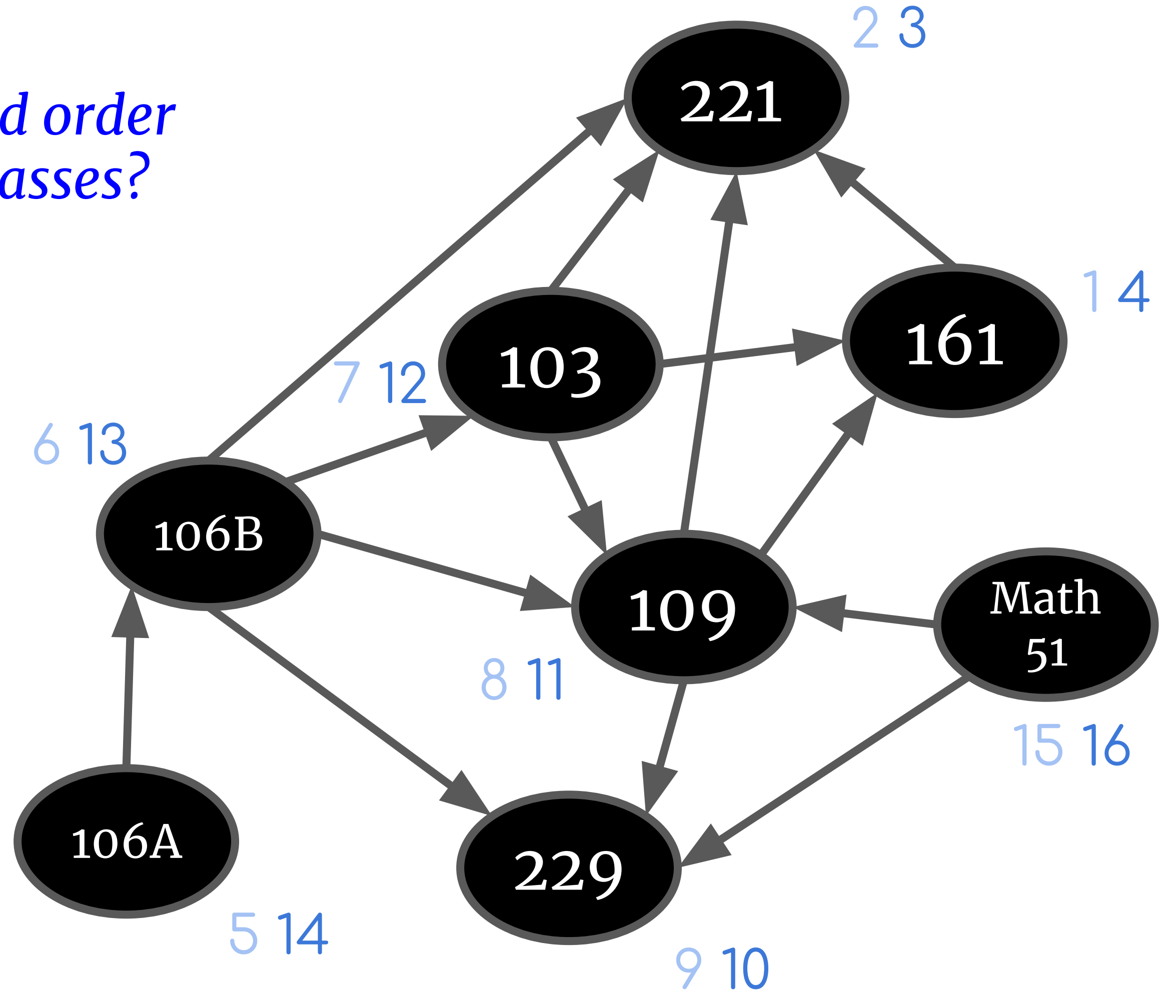


light blue = started

dark blue = done

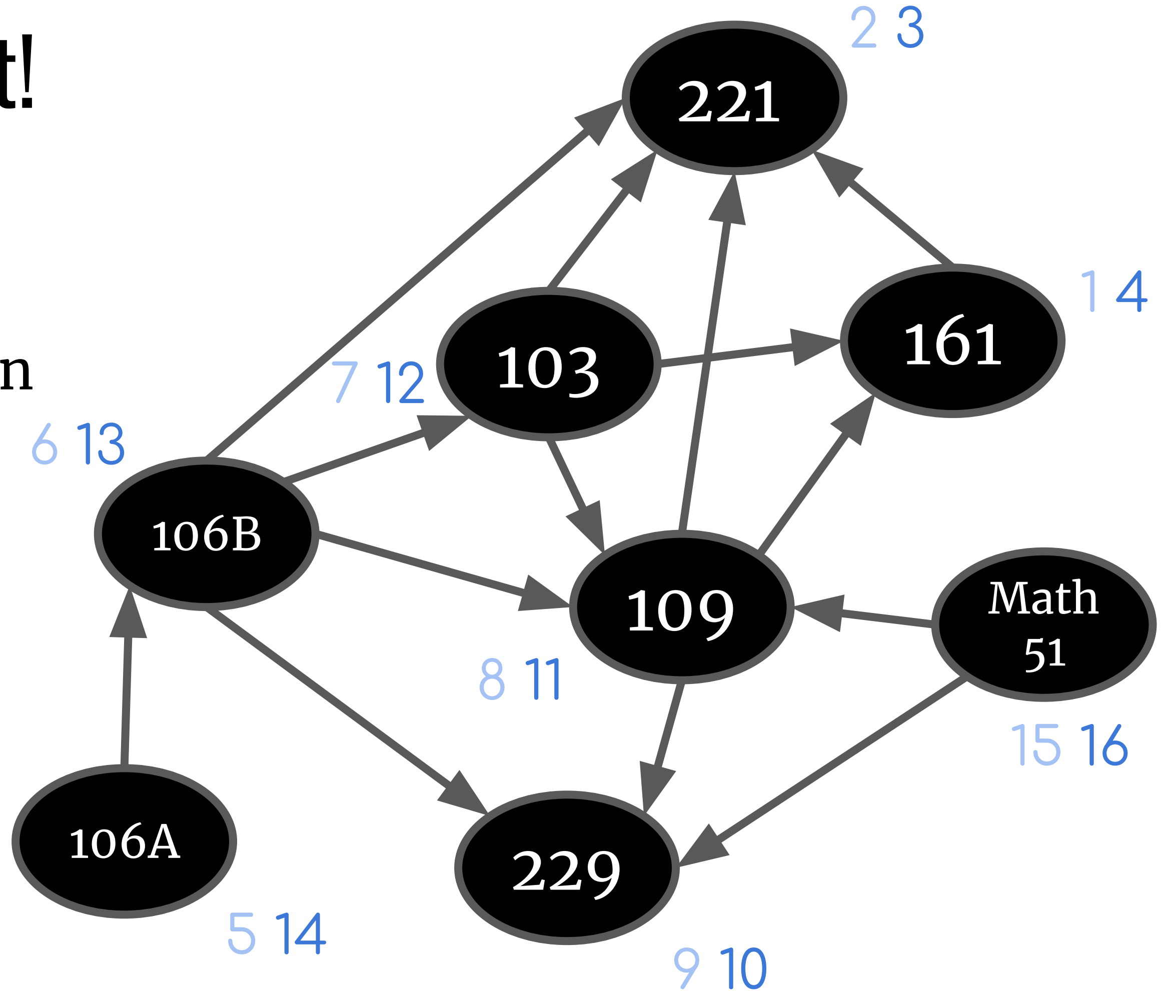


How can we use these numbers to find a valid order in which to take the classes?



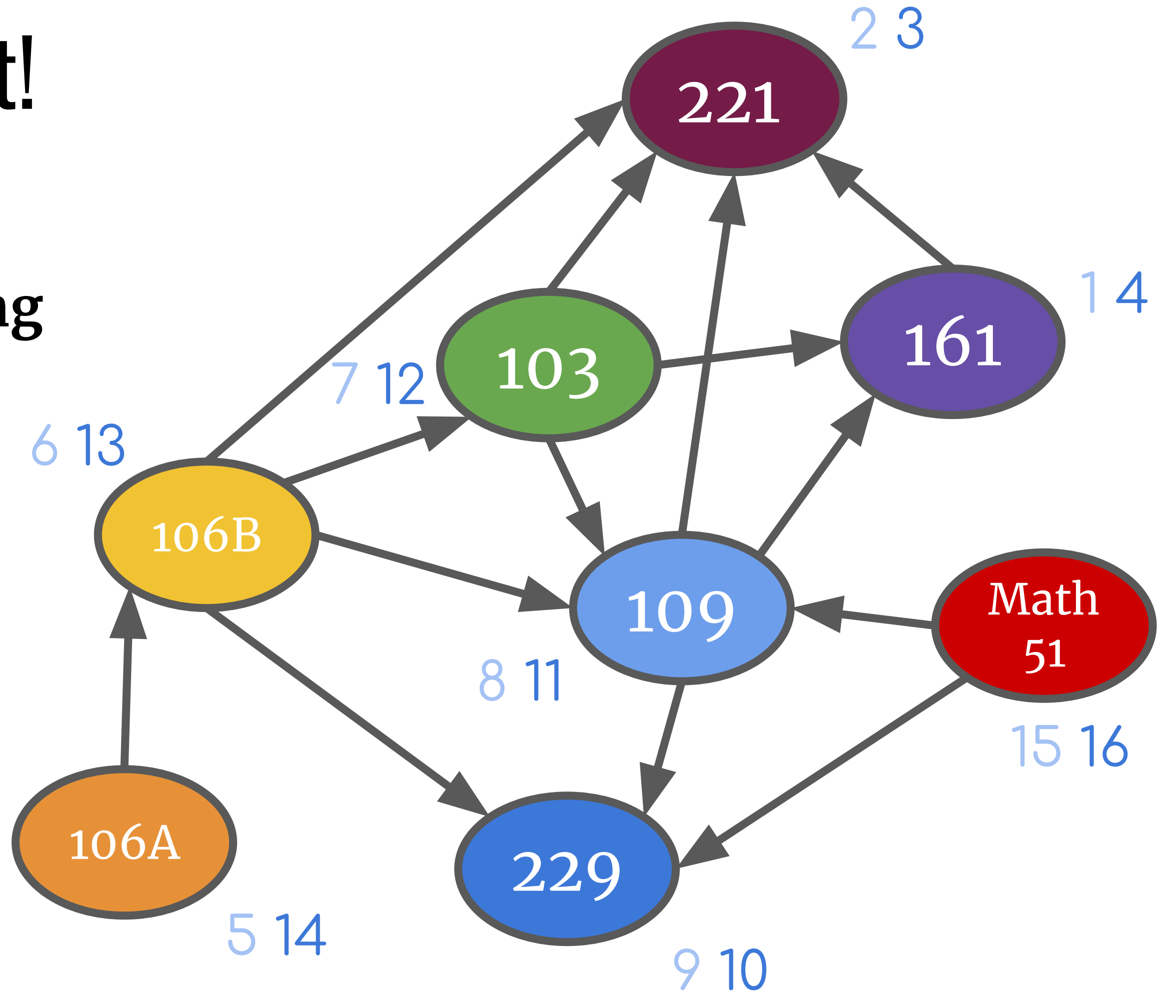
Topological sort!

- First, do a DFS.
- Then take the **finishing times in reverse order.**



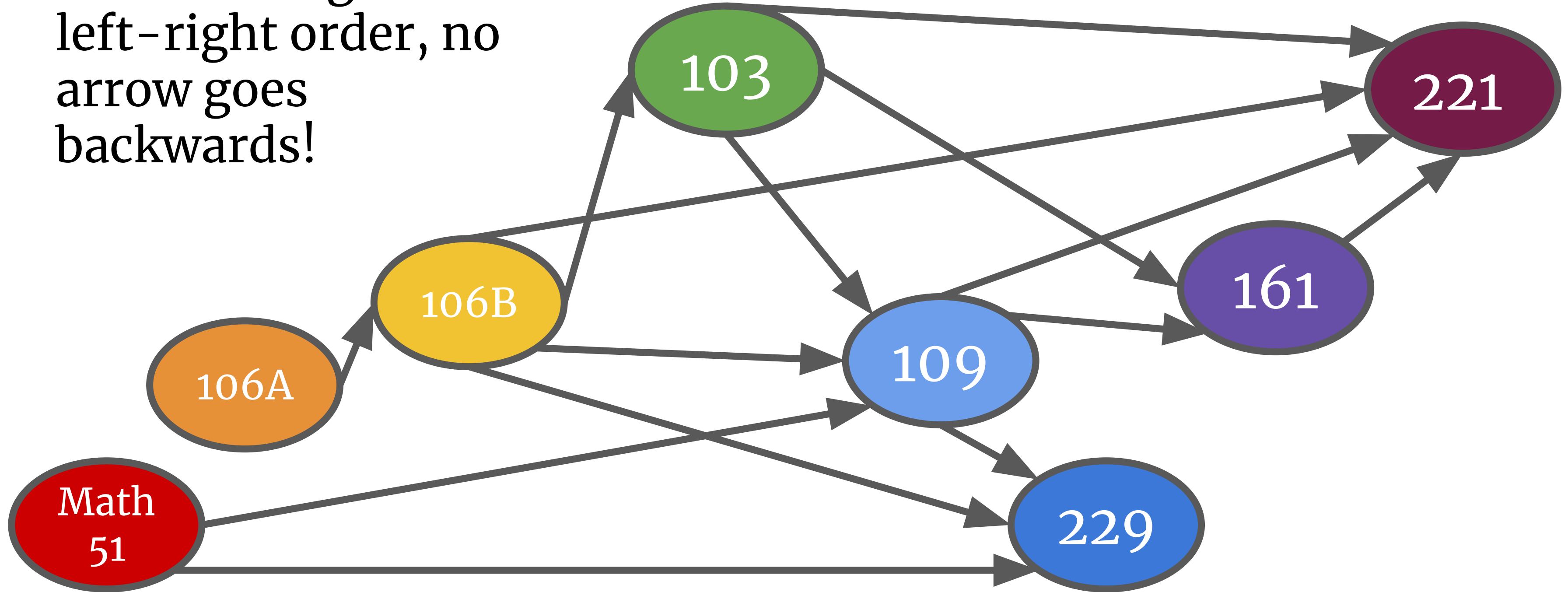
Topological sort!

- First, do a DFS.
- Take the **finishing times in reverse order.**



Topological sort!

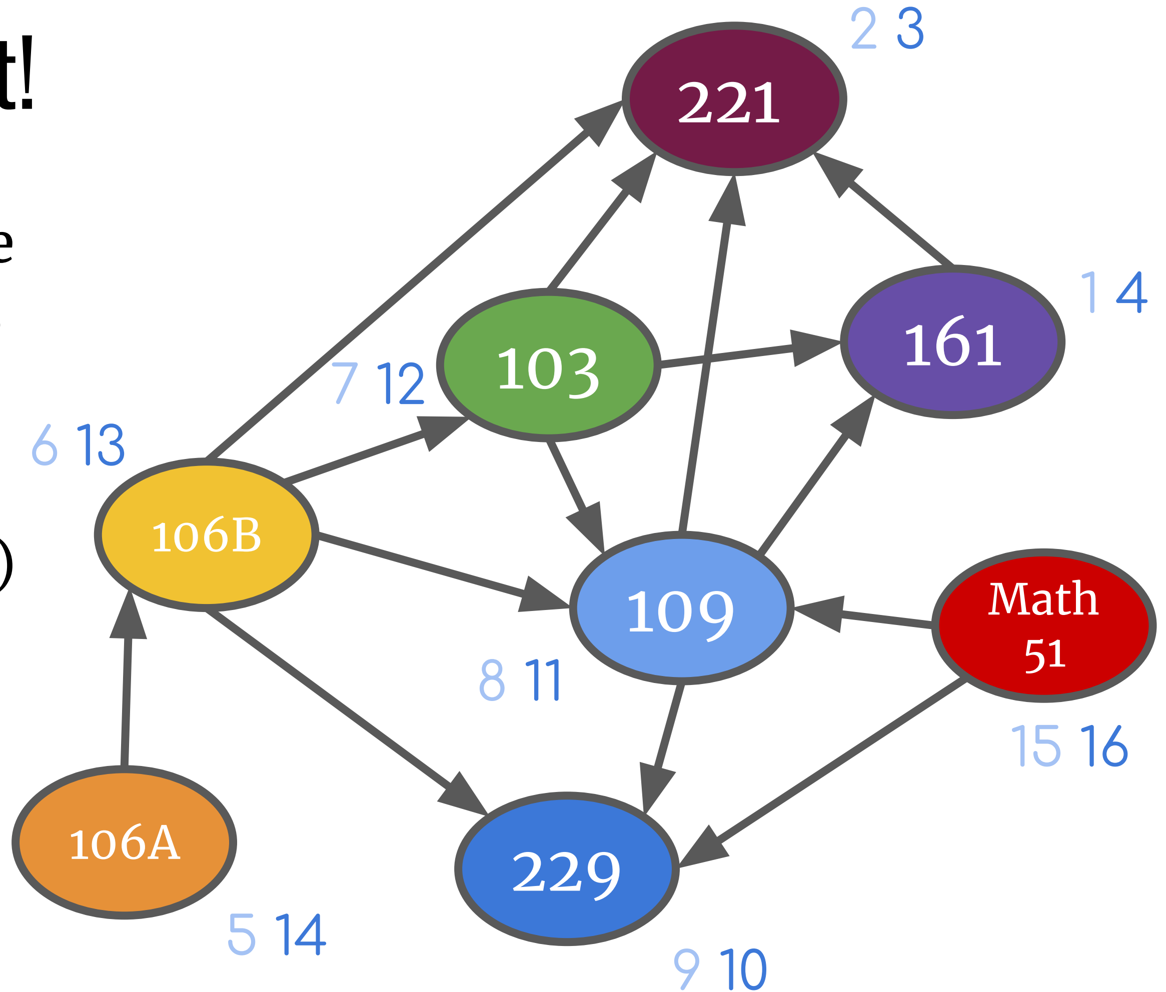
When arranged in this left-right order, no arrow goes backwards!



Topological sort!

There might be more than one topological sort of a DAG.

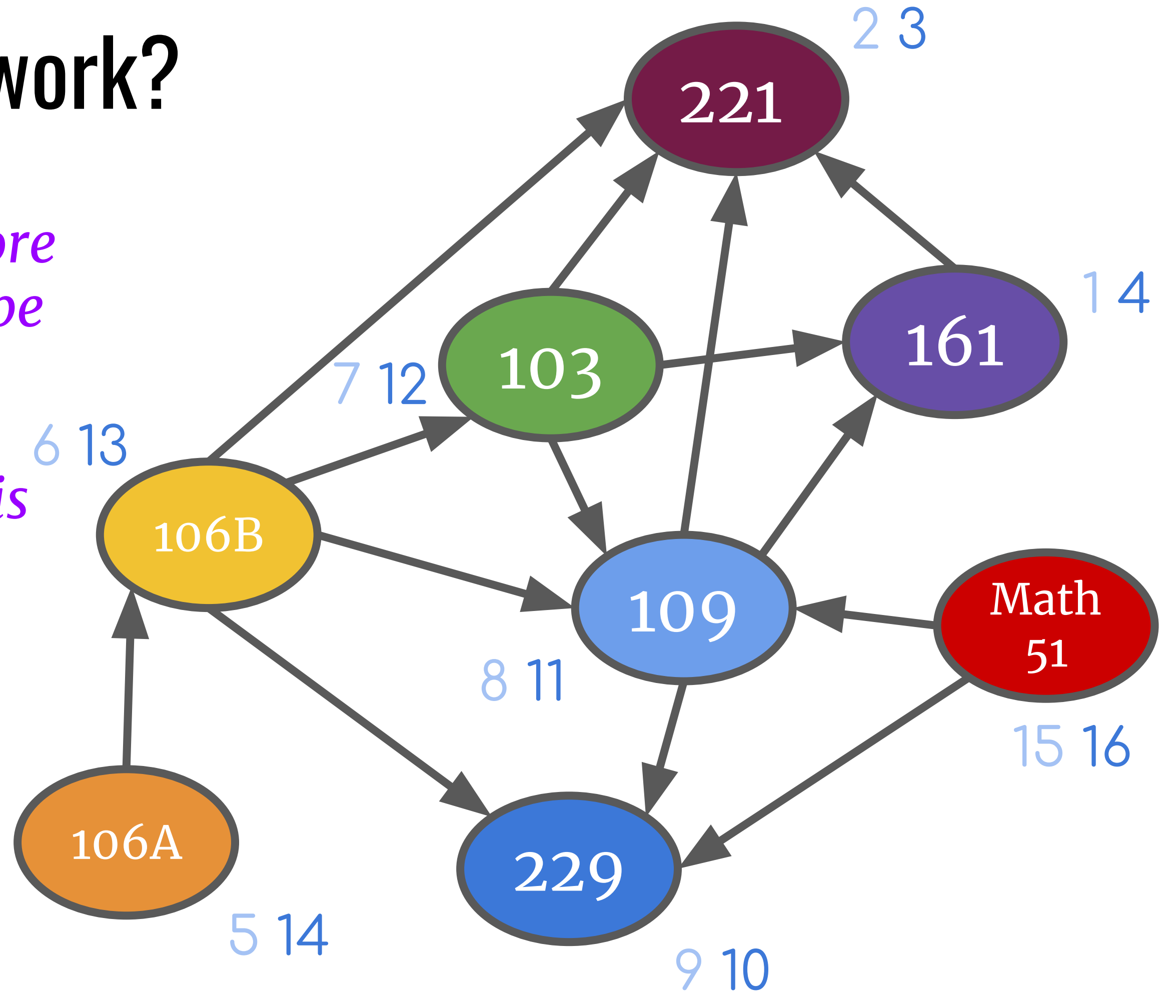
(e.g., we could take 106A before Math 51)



Why does this work?

If node A finishes before node B, there cannot be a path from A to B.

(We don't say a node is done until we've ***fully*** *explored* ***all*** *its descendants.)*

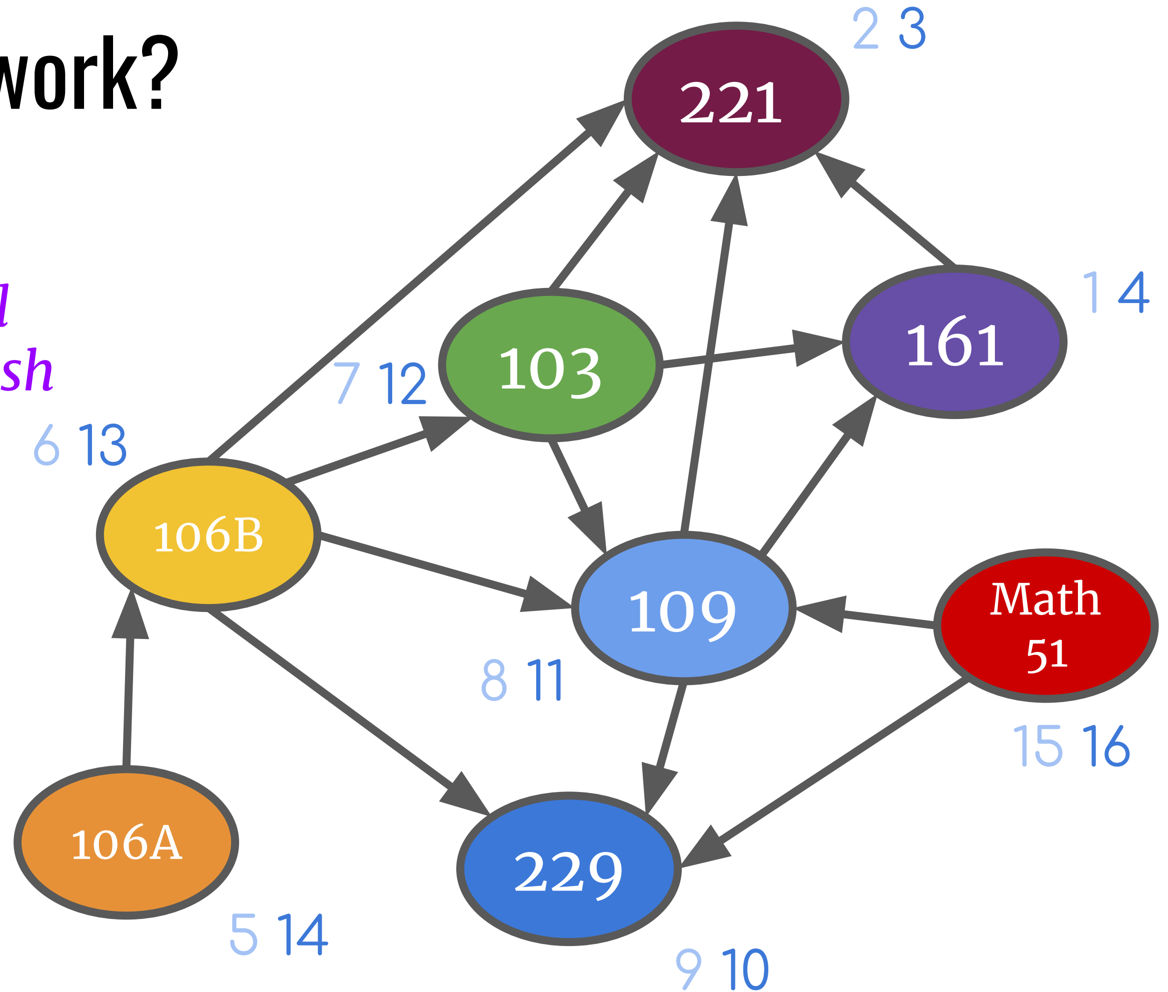


Why does this work?

Suppose we had 221 before 109 in our final order (of reversed finish times).

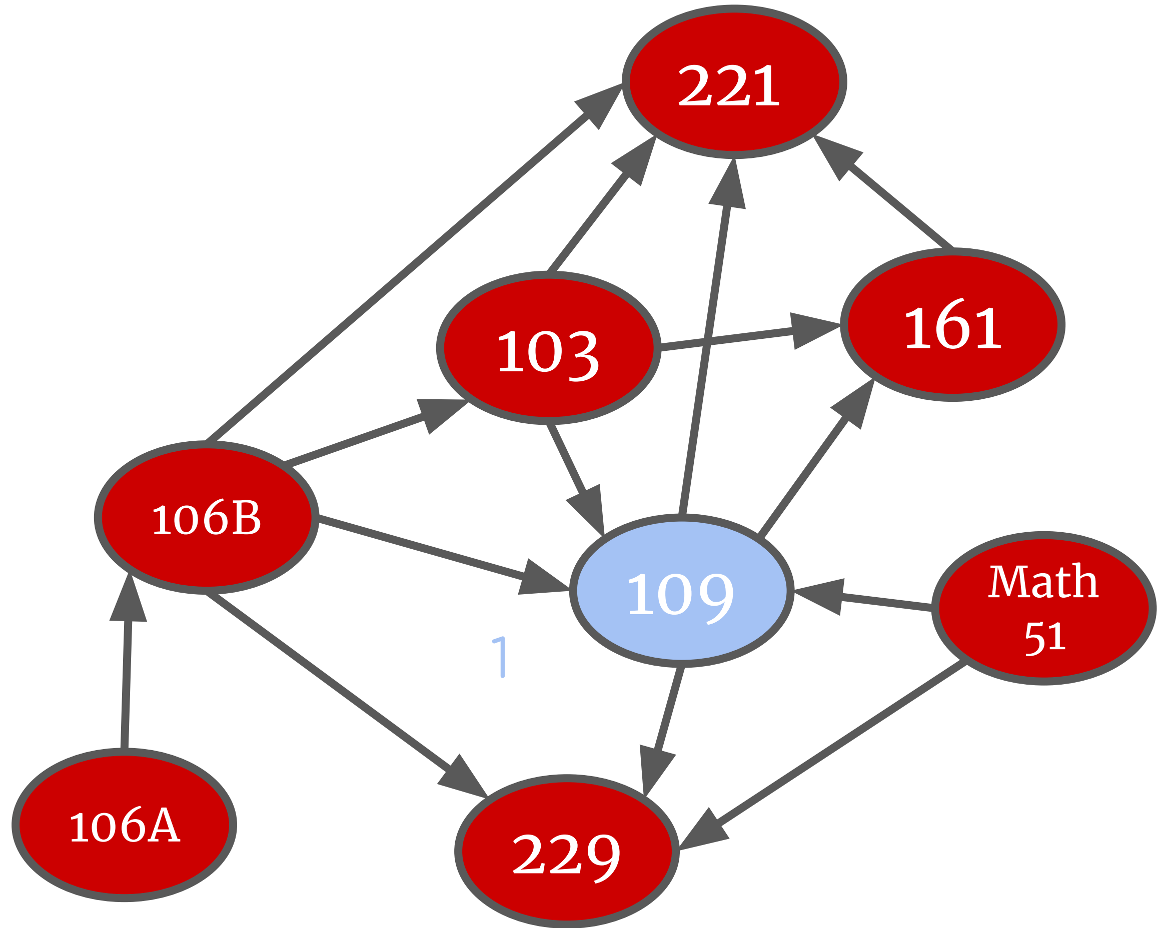
But then 109 finished before 221...

But 109 had to fully explore 221 first!

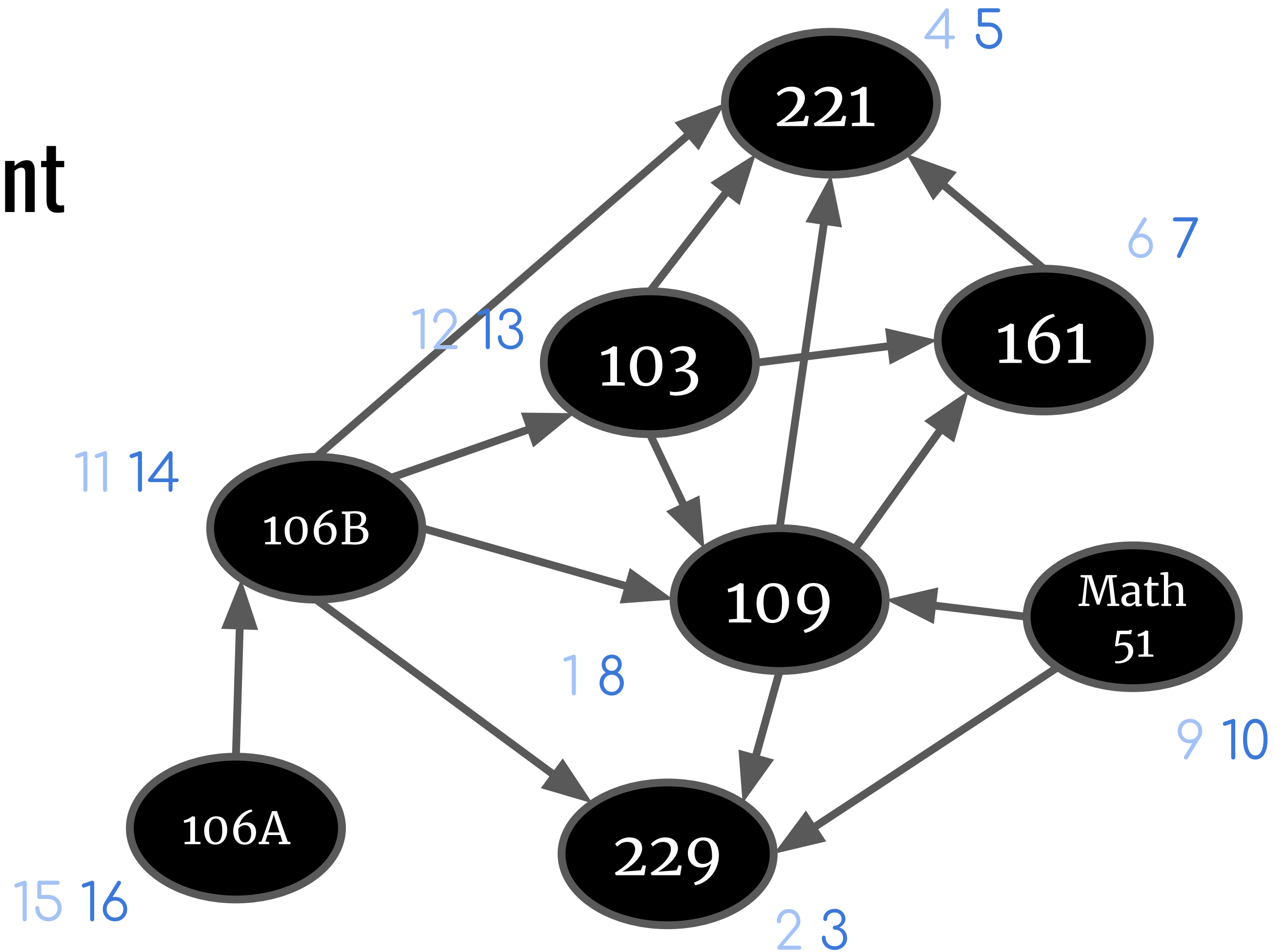


What if we'd made different choices?

Here I'll do different random starts, and break ties in reverse order

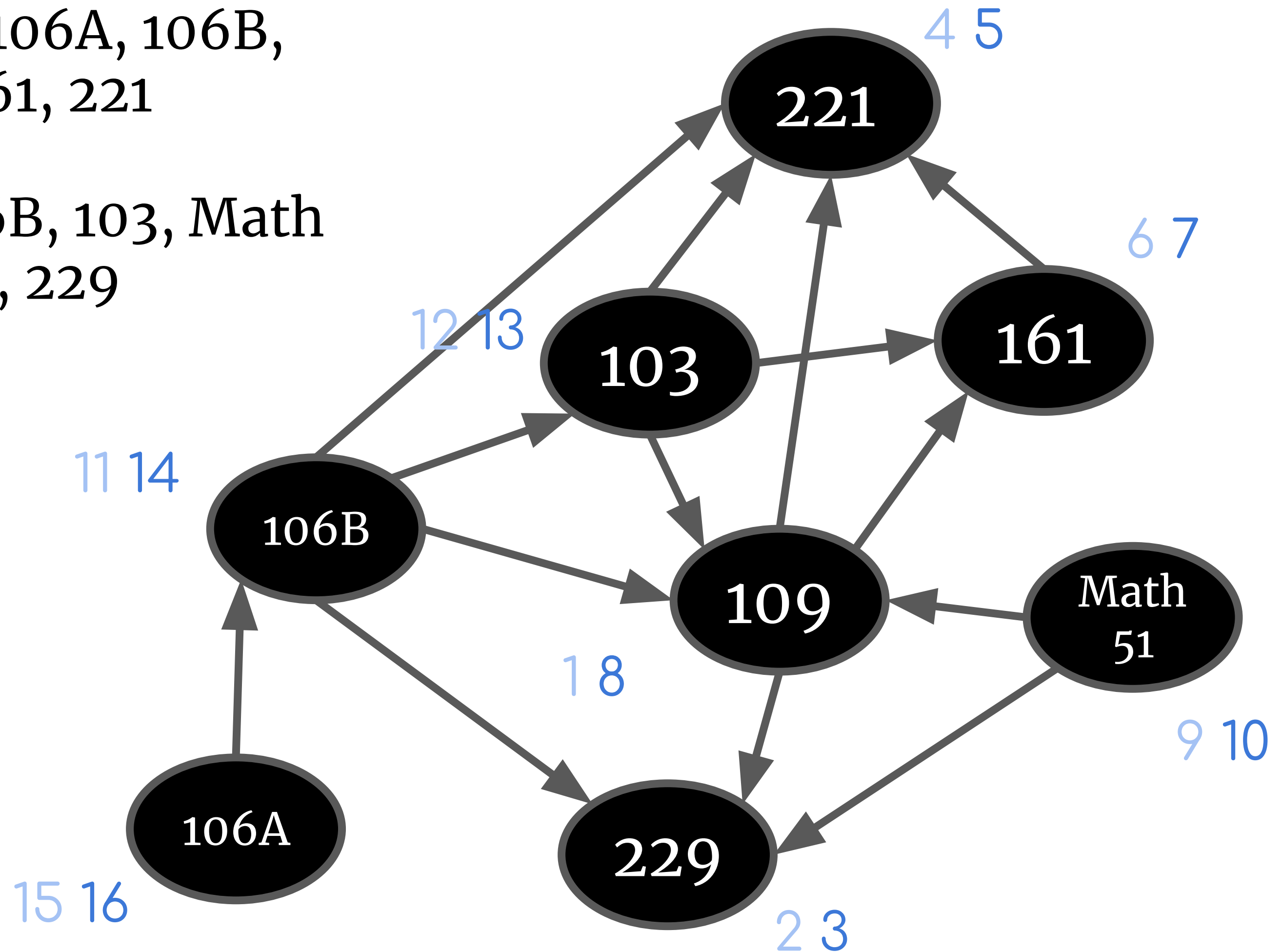


What if we'd made different choices?



Try 1: Math 51, 106A, 106B, 103, 109, 229, 161, 221

Try 2: 106A, 106B, 103, Math 51, 109, 161, 221, 229



Topological sort debrief

- Only makes sense for directed acyclic graphs (DAGs)
- Same running time as DFS: O(n + m)
  - We don't need to sort the list of finishing times at the end. We can just record nodes as they finish.
- Sensitive to start location and tiebreak rules, but can never give a *wrong* answer (e.g. where you take a class before its prereq)

7/15 Lecture Agenda

- Announcements
- Part 4-3: DFS and Topological Sort
- 10 minute break!
- Part 4-4: Kosaraju's Algorithm

7/15 Lecture Agenda

- Announcements
- Part 4-3: DFS and Topological Sort
- 10 minute break!
- Part 4-4: Kosaraju's Algorithm

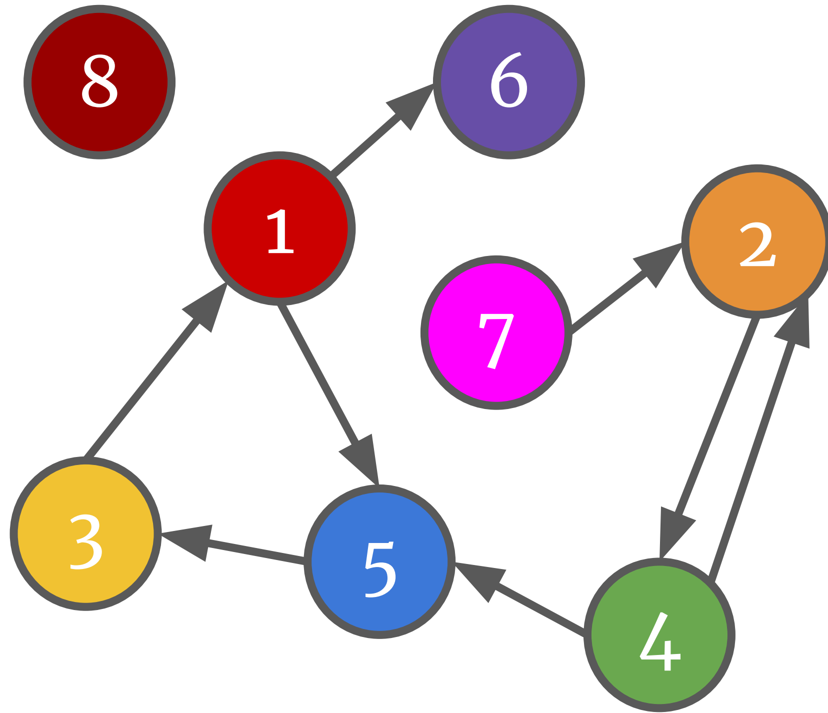
WORLD 4-4

Kosaraju's Marvelous SCC-finder

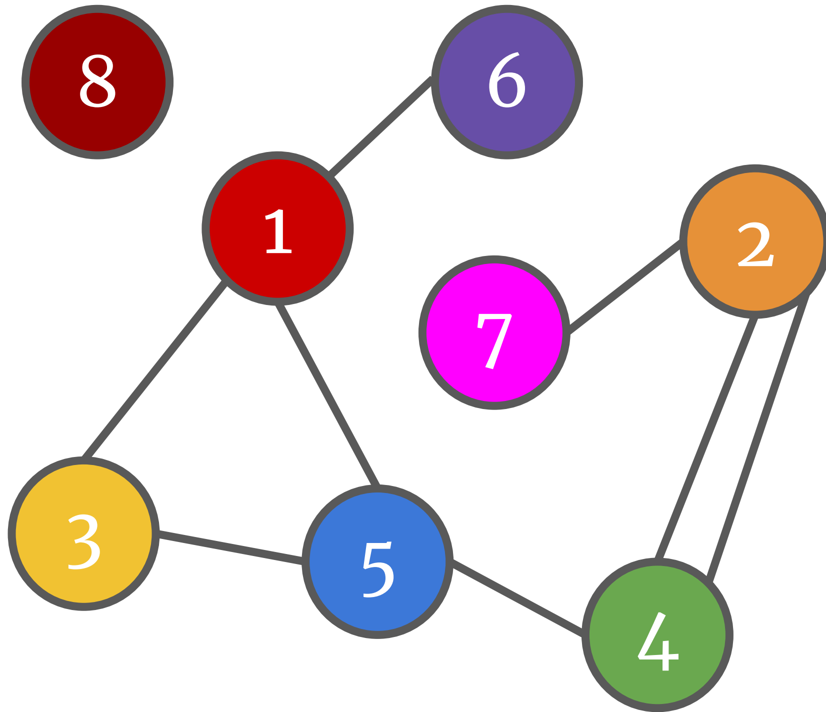
Divide and Conquer
Sorting & Randomization
Data Structures
Graph Search
Dynamic Programming
Greed & Flow

Special Topics

What do connected components even mean now?



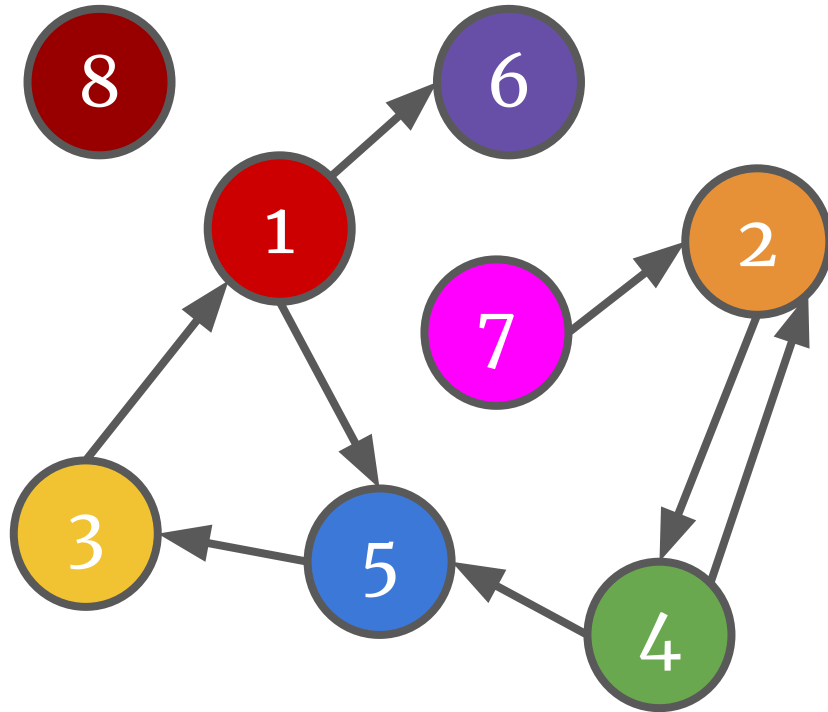
Weakly connected components



Pretend the edges are undirected, then use our previous definition of "connected component".

This isn't really important for us – it's just so the next name makes sense...

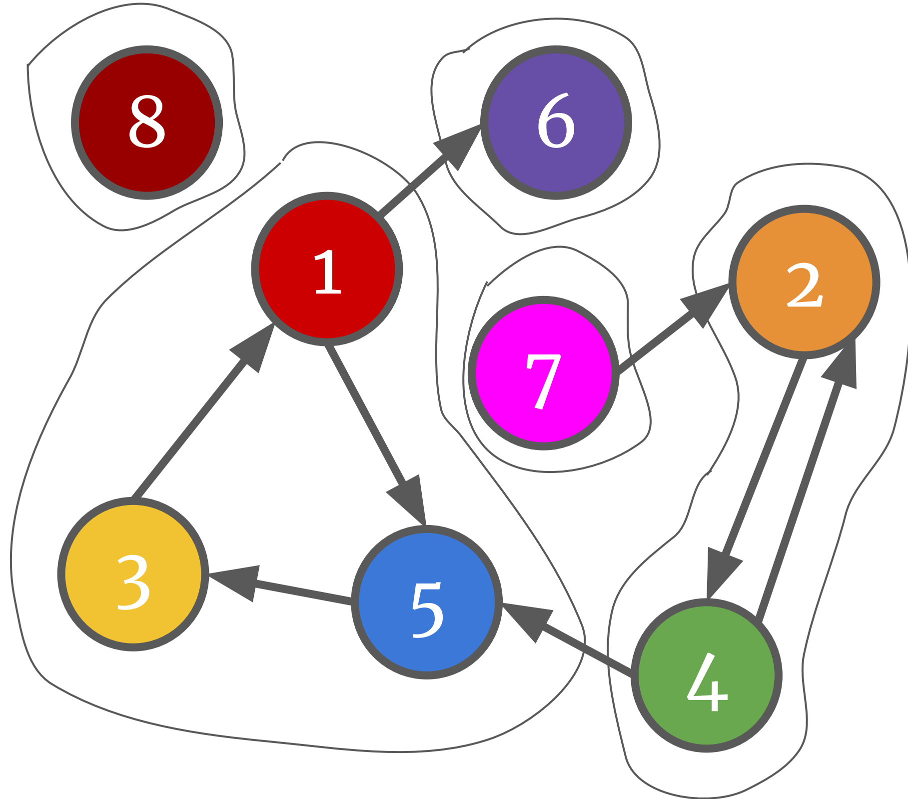
Strongly connected components (SCC)



An SCC is a set of nodes in a directed graph, each of which is reachable from all of the others in the SCC.

How many distinct SCCs are there here?

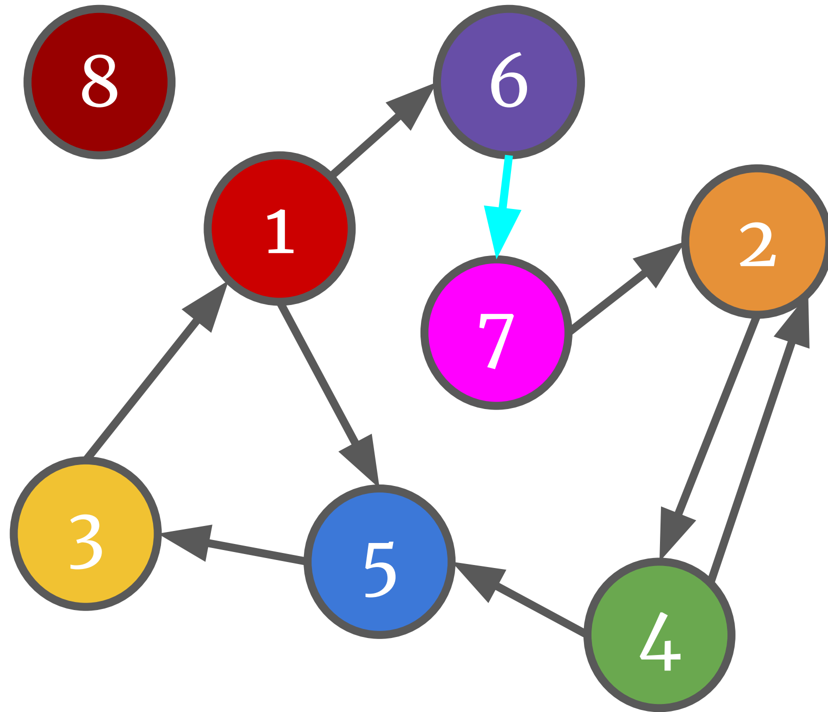
Strongly connected components (SCC)



An SCC is a maximal set of nodes in a directed graph, each of which is reachable from all of the others.

How many distinct SCCs are there here?
Five!

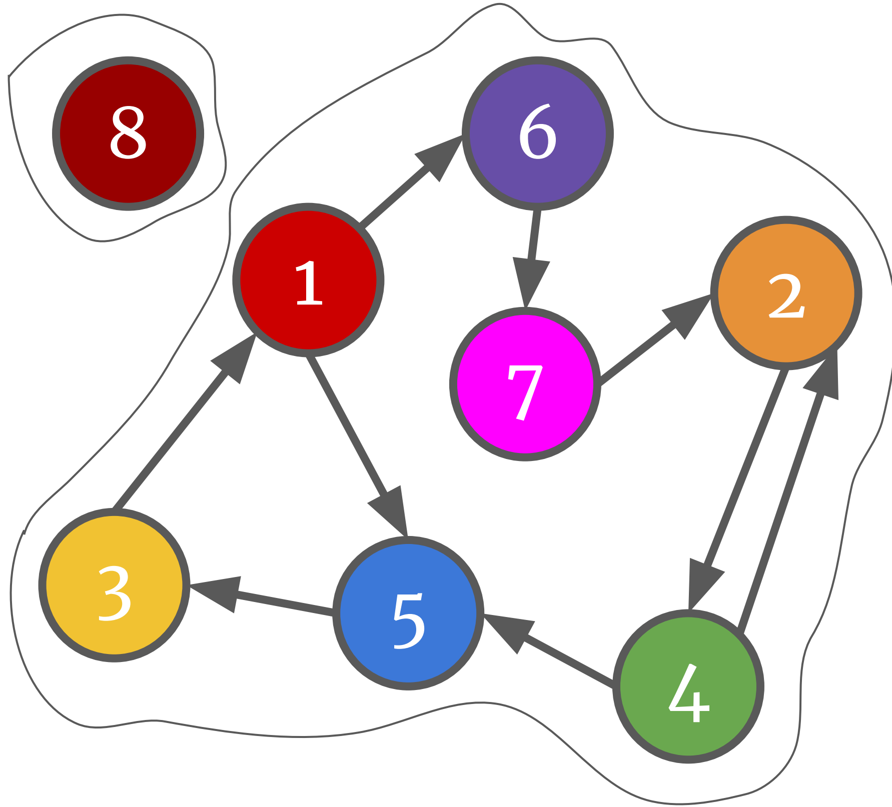
Strongly connected components (SCC)



An SCC is a set of nodes in a directed graph, each of which is reachable from all of the others.

How about now?

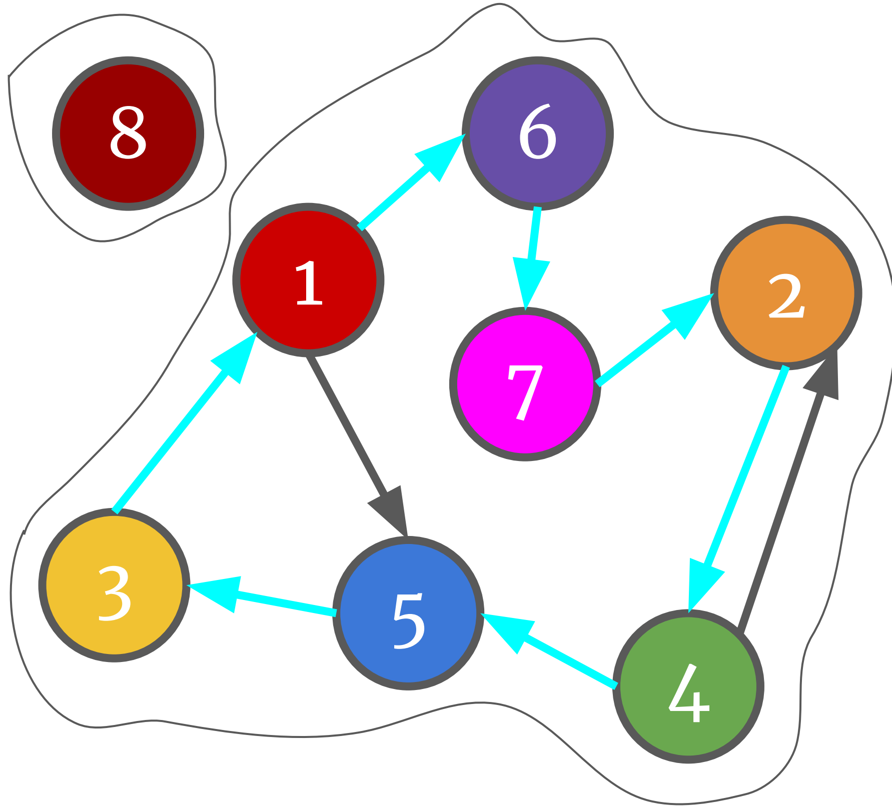
Strongly connected components (SCC)



An SCC is a set of nodes in a directed graph, each of which is reachable from all of the others.

How about now?
Two!

Strongly connected components (SCC)



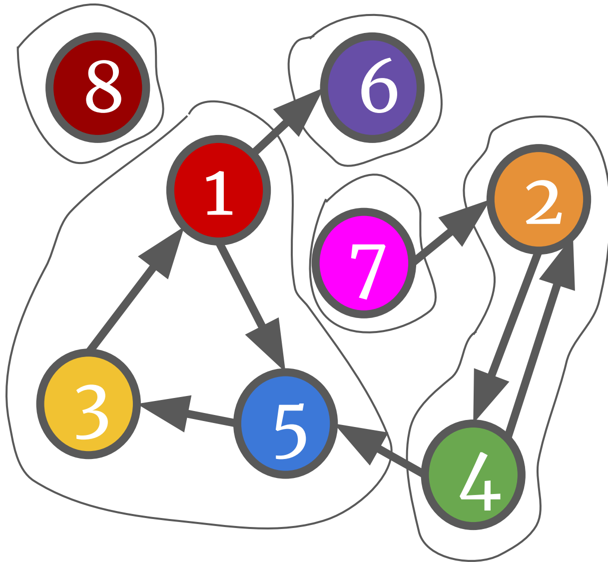
An SCC is a set of nodes in a directed graph, each of which is reachable from all of the others.

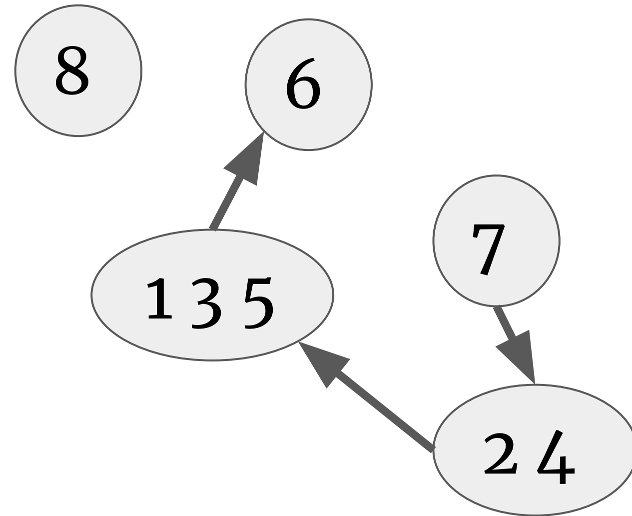
How about now?
Two!

To ponder: is a cycle required to demonstrate this?

Why care about SCCs?

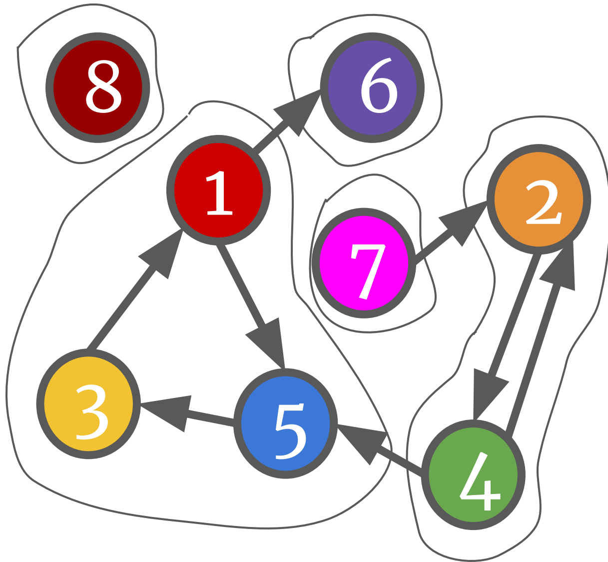
- They break a graph down into little self-contained universes. This matters a lot in, e.g., social networks.

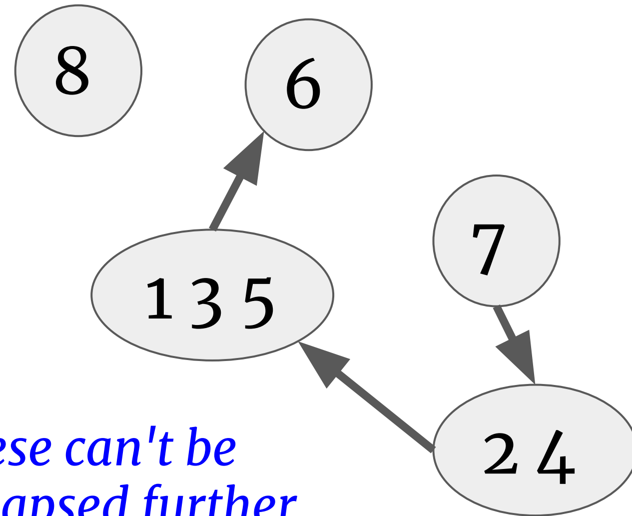




Why care about SCCs?

- They break a graph down into little self-contained universes. This matters a lot in, e.g., social networks.



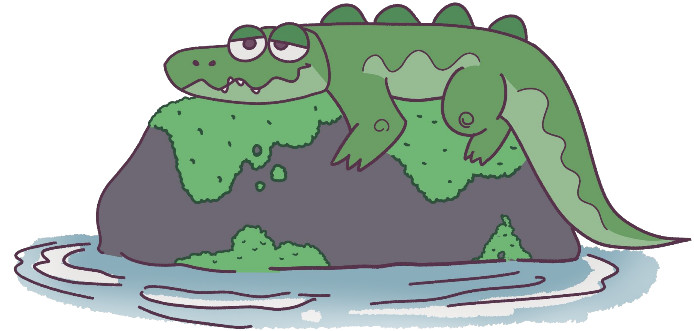


These can't be collapsed further into SCCs.

How do we find SCCs?

- One idea is to do one DFS starting from every node, then do something with those results…

Too much work! There has to be a better way…



Kosaraju's Algorithm

- Named for S. Rao Kosaraju, a prof at Johns Hopkins
- Finds SCCs in a directed graph $O(n+m)$ time
- That's not a typo!