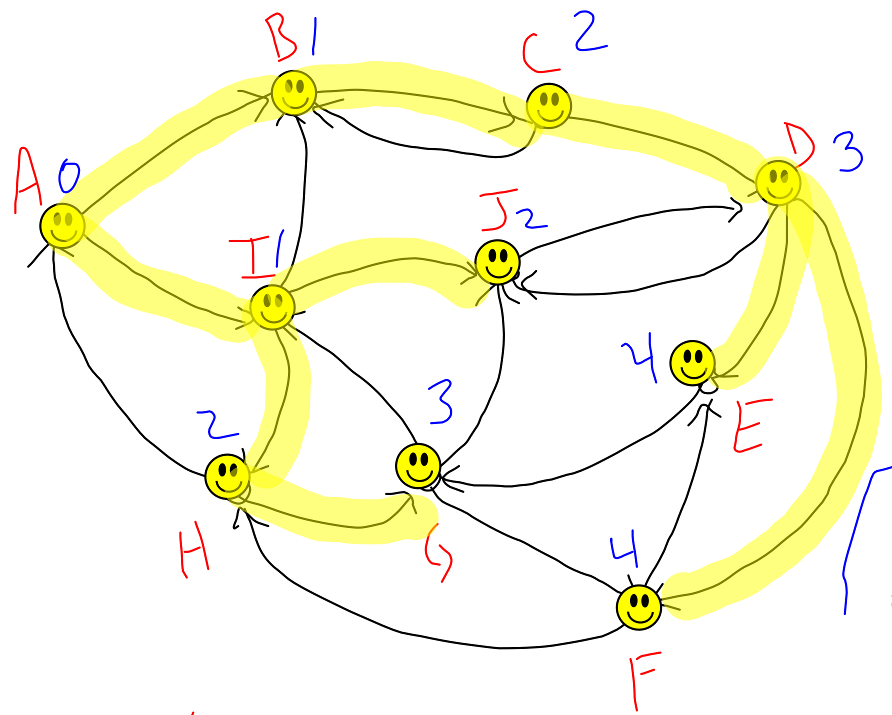
Searches in Graphs!

Consider a directed graph D



Q: ~~A B I C H J D G E F~~

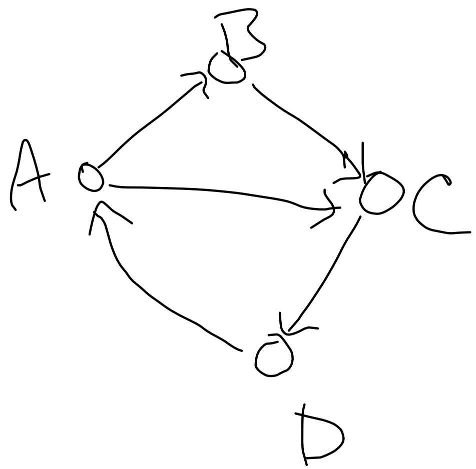
Done.

Let's do a Breadth-First Search of D, starting at A.

- Let Q be an empty queue (First-in-first-out list)
- Put A on Q.
- At each step,
  - pop an element u from Q.
  - Add u's unqueued neighbors to Q.
- Stop when Q is empty.

Def: The *distance* between two vertices in a directed graph is the ~~te~~ # of arcs (edges) on the shortest path between those two vertices.

Eg:



$d(A, C) = 1$

$d(A, A) = 0$

$d(D, C) = 2$

$d(B, A) = 3$

In BFS, we want to record $d(\text{root}, \text{other vertices})$, and since the root is fixed, we'll use shorthand $d[A]$ to mean $d(\text{root}, A)$. Also $\pi[A]$ = parent of A in BFS-tree.

BFS (D, A)

D = digraph, A = root vertex

```
#define INFINITY 1000000
```

~~#define INFT~~

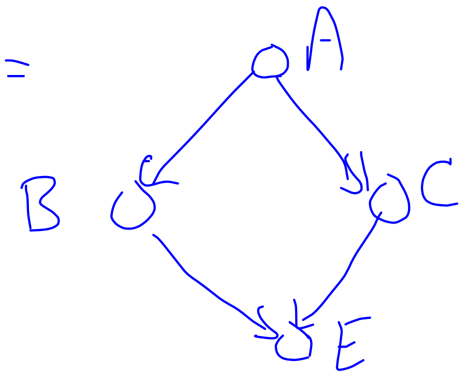
Some # that is guaranteed to be > any actual dist.

```
Let Q be a queue,
Q ← ∅
Add A to Q.
d[A] = 0
π[A] = Nil
while (Q ≠ ∅)
    do pop u from Q.
    for each out-neighbor v of u
        do if d[v] = ∞
            then d[v] = d[u] + 1
                 π[v] = u
                 push v onto Q.
```

for each vertex $v \in D$, do $d[v] = \infty$

Note: BFS builds a sub spanning tree of D, ~~sometimes~~

D =



BFS(D,B) gives B→E

BFS(D,A) gives the whole graph.

BFS(D,E) gives only E.

BFS(D,A) Builds a spanning subtree on ~~B~~ those vertices of D which are reachable from A.

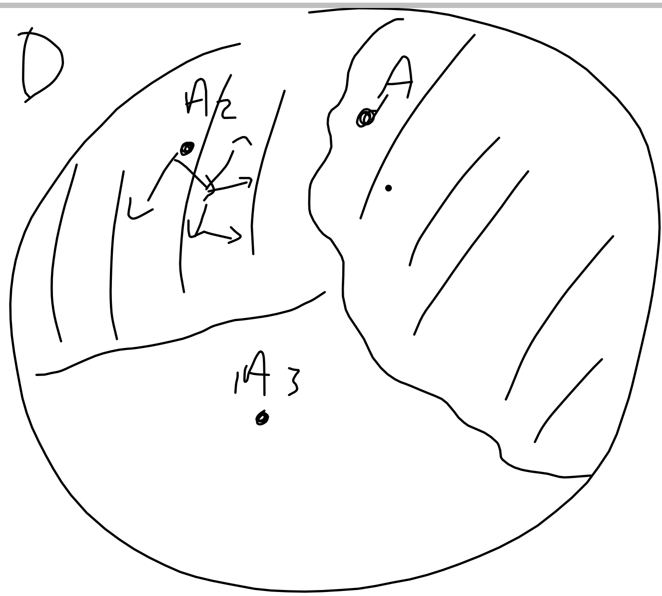
Question: What is the maximal strongly connected component of D containing A.

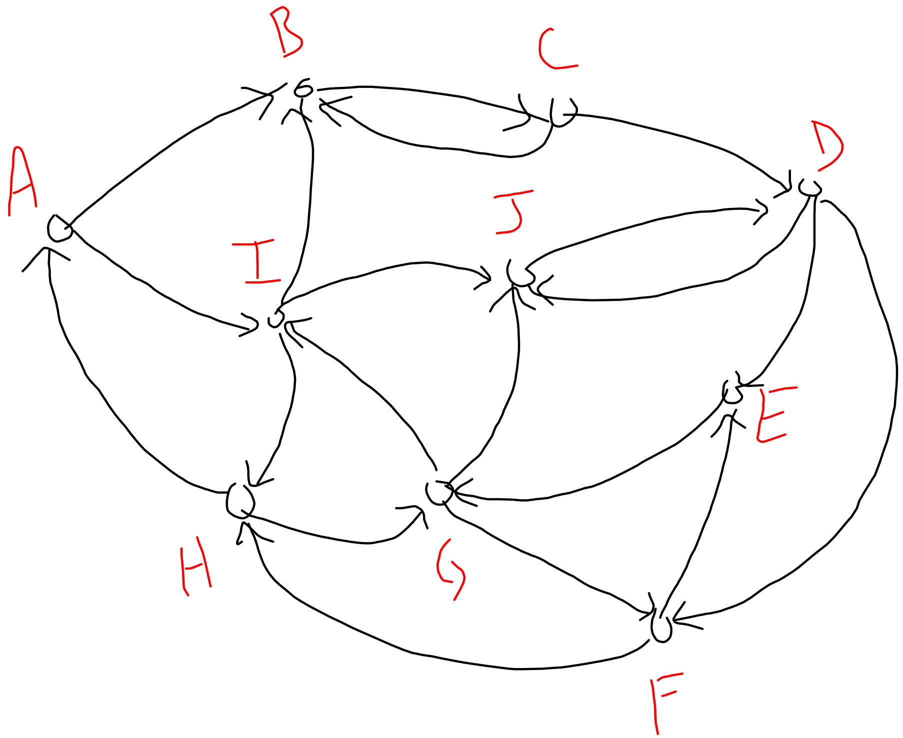
BFS (D)

While there exists a vertex v of D with $d[v] = \infty$

do BFS(D, v)

Nevermind





Title: Mar 27 - 10:04 AM (6 of 6)