1) Dynamic Connectivity

union (a, b) → connect 2 objects

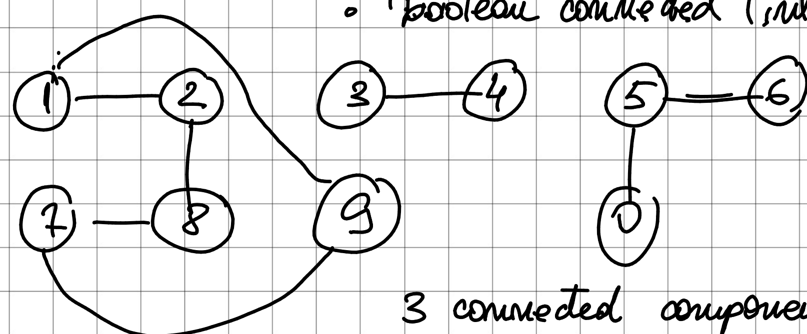
find (a, z) – is there a connection?

* connected components : max set of objects that are mutually connected

$\boxed{\text{Data Type}}$

class UF (int N)

- void union (int p, int q)
- boolean connected (int p, int q)



3 connected components

II) Quick Find

- data structure : array id[ ], size N
- interpretation : p and q are connected iff they have the same id

find: check if p and q have the same id

union: to merge components $\rightarrow$ change all entries where id equals id[p] to id[q]

* Cost Model

	init	union	find
quick-find :	$O(N)$	$O(N)$	$O(1)$

$\Rightarrow$ n union commands on n objects

$\Rightarrow \boxed{O(n^2)}$ quadratic time

III) Quick-Union

- data structure : id[] of int, size N
- interpretation : id[i] = parent of i

* set of trees, a forest

find : check if p and q have the same root

union : set id of p's root to the id of q's root class : QuickUnion

public boolean connected (int p, int q)

public void union (int p, int q)

private int root (int i)

$\boxed{\text{cost}}$	init	union	find
quick union	N	N^{+} cost of finding roots	N

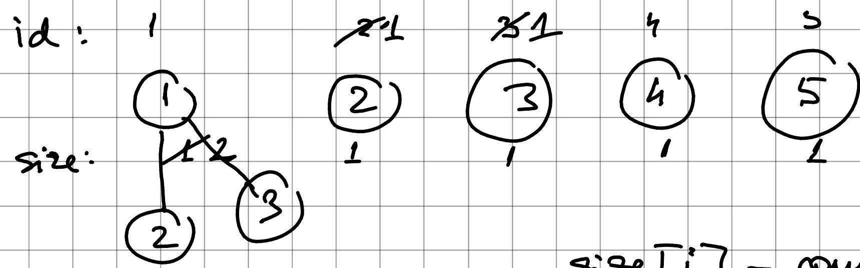
IV) Quick Union Improvements

1. weighting

- avoid tall trees
- smaller tree goes below larger tree, no matter of the order of the union arguments

- data structure : int[] id
  int[] size =>

size[i] = no. of objects rooted in the tree at i



size[i] = counts the no. of objects in the tree, not the levels

union (1, 2)

union (3, 2)

Running time

- find : time is proportional to depth of p and q
- union: constant time given roots

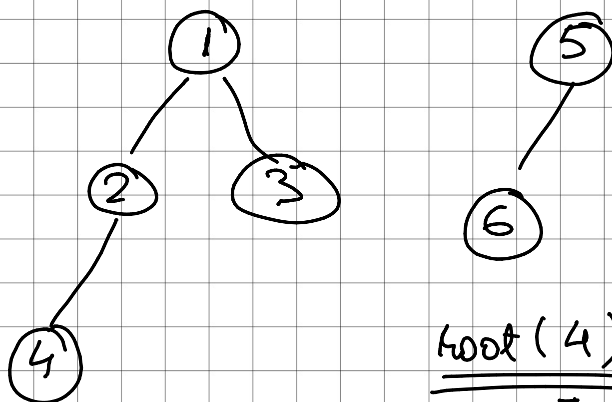
Proposition: depth of any node x is at most $\boxed{\log_2 N}$

cost	init	union	connected
weighted QU	N	lg N	lg N

② path compression

$id[i] = id[id[i]]$

make every other node in path point to its grandparent (halving path length)



root(4)

$id[4] = 2$

$id[4] = id[id[4]]$

$id[2] = 1$

$id[4] = 1$

$i = id[4] = 1$

* we could add another loop in the root function to set the id of each examinated node to the root

$$\boxed{\lg^* N < 5}$$

Applications

Percolation

p = open sites

$1-p$ = blocked sites

$p > p^*$: almost certain it percolates

$p < p^*$: almost certain it doesn't percolate

$$\boxed{p^* \approx 0.593}$$