ese 101 6-4-20

- last lecture today!
- Tomorrow (friday) is last office hour
- last quiz (quiz #5) is next Tuesday.
  1 hour window is
  12:00 pm Tuesday June 9
  - 6:am am Wednesday June 10
- absolutely last ext. of Pa7, 1 more day only.
- Complete SETs to get Extra credit.

Disjoint Set ADT

State:

Pairwise disjoint

$$\mathcal{S} = \{ \{ \ldots \}, \{ \ldots \}, \ldots, \{ \ldots \} \}$$

operations:

- MakeSet(x):
  Create new set $\{x\}$ in $\mathcal{S}$. x may not belong to any existing member of $\mathcal{S}$.

$$\mathcal{S} = \mathcal{S} \cup \{\{x\}\}$$

- Union(x, y) pre: $S_x \neq S_y$
  unites S_x and S_y in $\mathcal{S}$,
  and reduces # of sets in $\mathcal{S}$
  by 1

$$\mathcal{S} = \mathcal{S} - \{S_x\} - \{S_y\} \cup \{S_x \cup S_y\}$$

- FindSet(x)
  returns (a pointer to) the representative
  of the set S_x. Note: $S_x == S_y$
  iff FindSet(x) == FindSet(y)

we will analyze runtimes using parameters

$$n = \#\text{MakeSet ops.}$$

and

$$m = (\#\text{MakeSet}) + (\#\text{Union}) + (\#\text{FindSet})$$

note: since Union reduces $|\mathcal{S}|$ by 1, must have

$$(\#\text{union}) \leq n-1$$

will analyze sequences of operations. runtimes will depend on particular history of the underlying data structure.

Two Questions:

(1) why?

(2) How?

Ex. (answer to why)

Problem: determine # of conn. components in a Graph $G = (E, V)$, and for any two $x, y \in V$ answer the question: is x reachable from y.

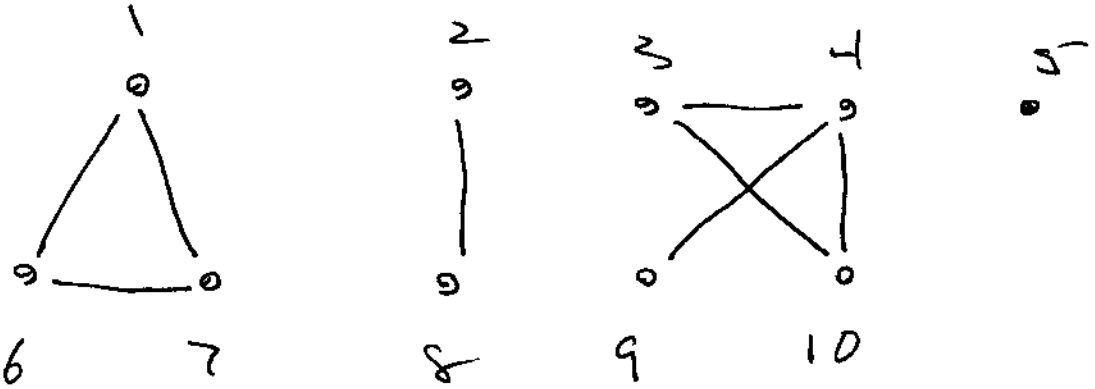
Components(G)

```
1. for each x in V
2.     MakeSet(x)
3. for each (x, y) ∈ E
4.     if FindSet(x) ≠ FindSet(y)
5.         Union(x, y)
```

SameComponent(x, y)

```
1. if FindSet(x) == FindSet(y)
2.     return true
3. return false
```

Ex.



E	$\mathcal{S}$
loop 1-2	$\{1\}\{2\}\{3\}\{4\}\{5\}\{6\}\{7\}\{8\}\{9\}\{10\}$
(1,7)	$\{1,7\}\{2\}\{3\}$ ... $\{10\}$
(2,8)	$\{1,7\}\{2,8\}\{3\}\{4\}$ ..... $\{10\}$
(3,4)	$\{1,7\}\{2,8\}\{3,4\}\{5\}$ ... $\{10\}$
(6,7)	$\{1,6,7\}\{2,8\}\{3,4\}\{5\}$ .... $\{10\}$
(9,4)	$\{1,6,7\}\{2,8\}\{3,4,9\}\{5\}$ ... $\{10\}$
(4,10)	$\{1,6,7\}\{2,8\}\{3,4,9,10\}\{5\}$
(1,6)	" " " "
(3,10)	" " " "

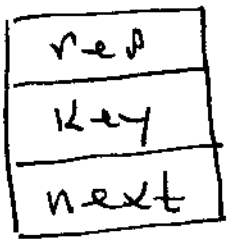
Two answers to how:

(21.2) Linked List Representation

(21.3) Disjoint Set Forest Rep.

(21.2) Linked List

Node

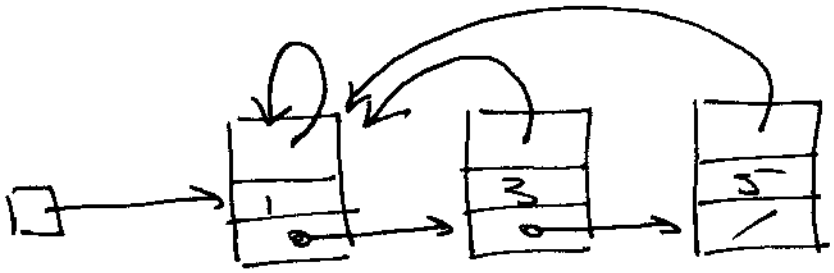


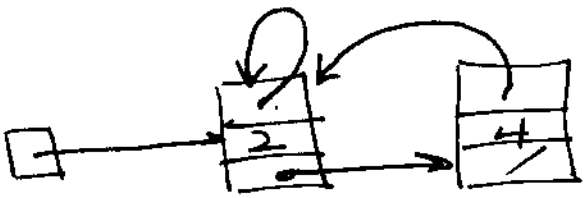
each set in collection is a linked list.

rep Node is head of list.

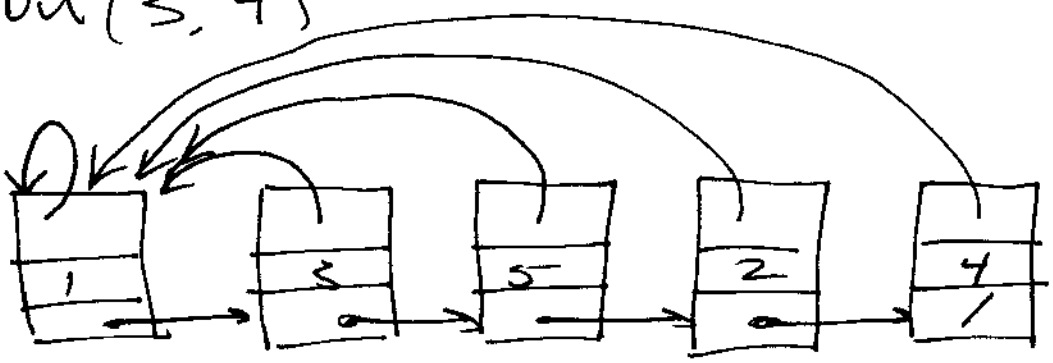
Ex.

$S = \{\{1, 3, 5\}, \{2, 4\}\}$





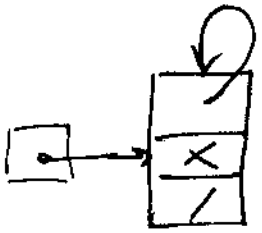
union(3, 4)



note :- cost in # of rep. updates which is $|S_y|$

MakeSet(x)

1. create new list



cost $\Theta(1)$

FindSet(x)

1. return Rep[x] Cost = $\Theta(1)$

Union(x, y)

1. Concatenate lists S_x & S_y
2. Update Rep. Pointers in S_y to point to Rep[x]. (Cost = $|S_y|$).

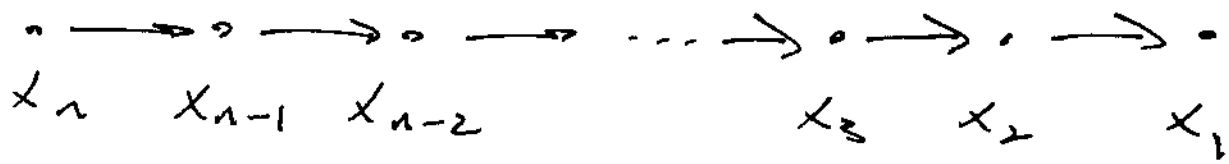
Note: certain seq. of operations can have poor ~~total~~ runtime.

Ex.

MakeSet(x_1)
MakeSet(x_2)
⋮
MakeSet(x_n)
} n

Union(x_2, x_1)
Union(x_3, x_2)
⋮
Union(x_n, x_{n-1})
} $n-1$

create linked list:



updates $= 1 + 2 + 3 + \cdots + (n-1)$

$$= \frac{n(n-1)}{2} = \Theta(n^2)$$

weighted union Heuristic

store the length of each list.
modify Union(x,y) so it always concatenates shorter of S_x & S_y to back of longer list.

Prev. example now costs $\Theta(n)$

$$\#\text{ updates} = \underbrace{1+1+1+\cdots+1}_{n-1}$$

$$= n-1 = \Theta(n).$$

Thm worst can be is $O(m+n\log n)$.