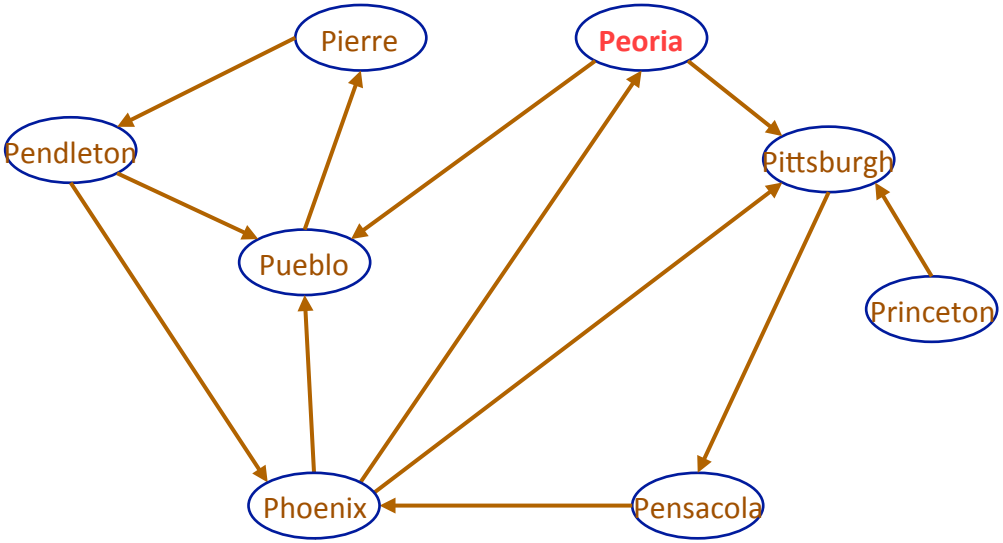
Single Source Reachability

Question

- What nodes are reachable from Peoria?



Single Source Reachability: Edge-List

```
findReachable (graph g, vertex start) {
  create a set of reachable vertices, initially empty. call this r.
  create a container for vertices known to be reachable. call this c
  add start vertex to container c
  while the container c is not empty {
    remove first entry from the container c, assign to v
    if v is not already in the set of reachable vertices r {
      add v to the reachable set r
      add the neighbors of v, not already reachable, to the
          container c
return r
```

Single Source Reachability: Stack

- Let's use a Stack as our container
- Basic algorithm:

  Initialize set of *reachable* vertices and add v_i to a stack

  While stack is not empty

    Get and remove (pop) last vertex *v* from stack

  if vertex v is not in reachable,

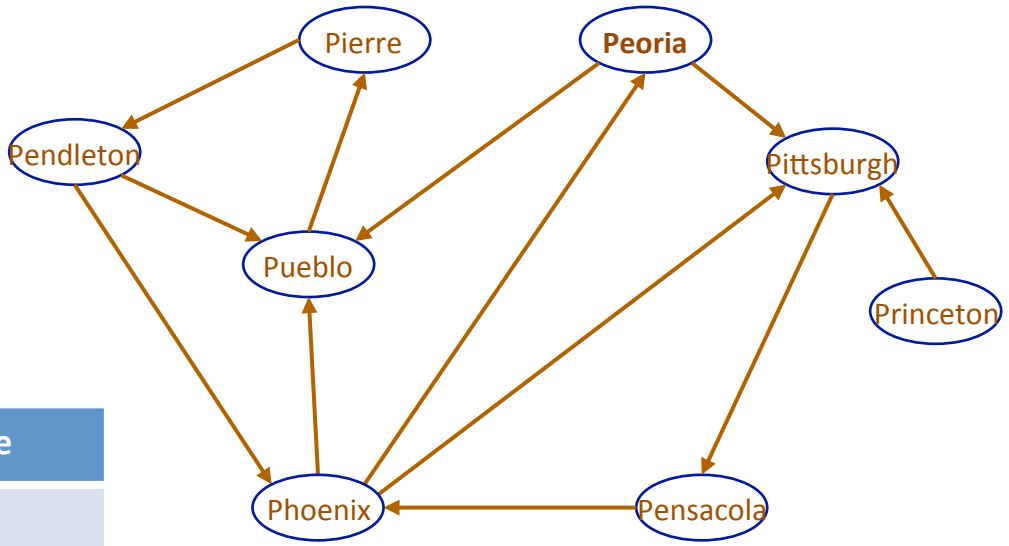
    add it to reachable

    For all neighbors, v_j, of *v, if* v_j *is NOT in reachable*

      add to stack

Single-Source Reachability: Stack

What cities are reachable from peoria? [Just for repeatability, when I push neighbors on the stack, I do so in alphabetical order)

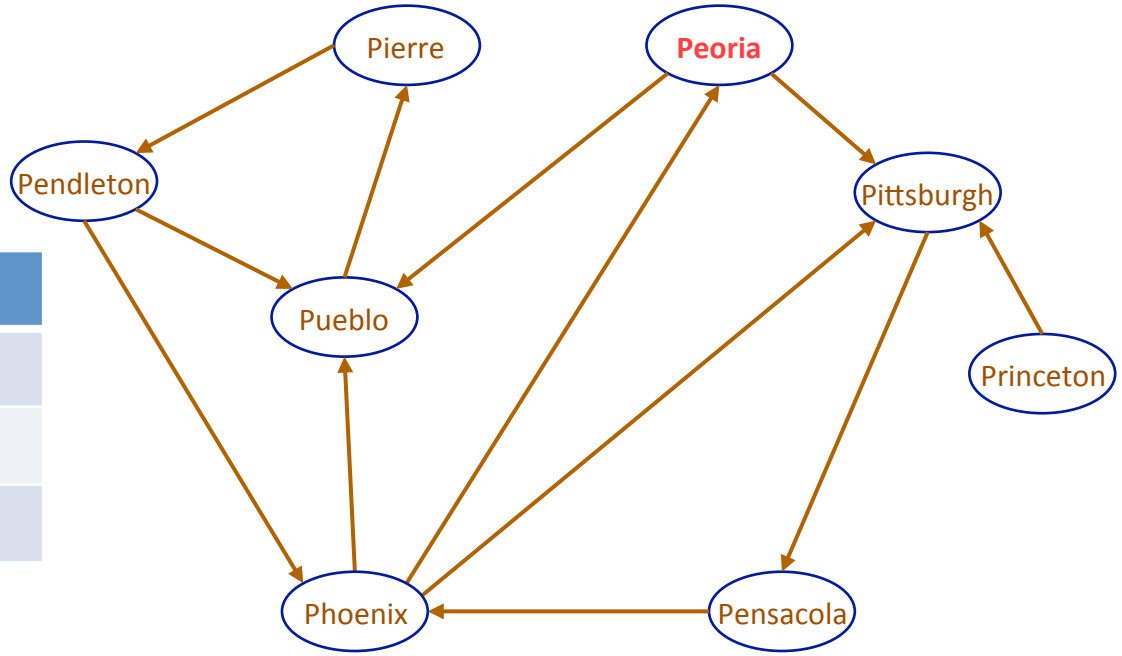


	Stack(top->bot)	Reachable
0	Peoria	{}

Single-Source Reachability: Stack

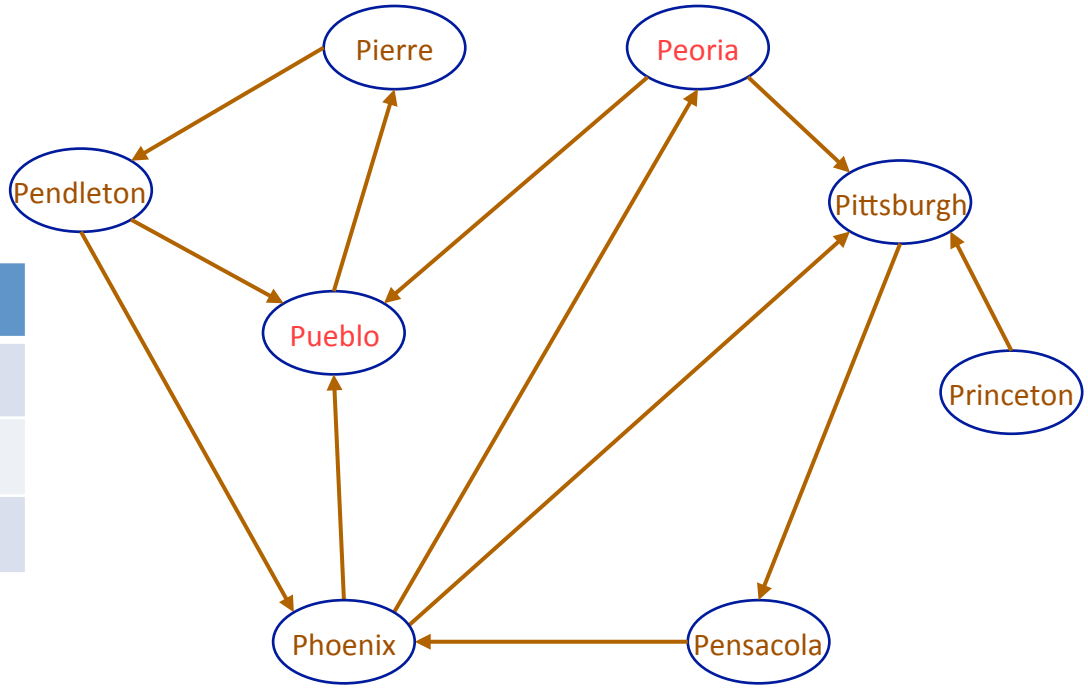
What cities are reachable from peoria?

	Stack(top->bot)	Reachable
0	peoria	{}
1	publeo, pittsburgh	peoria



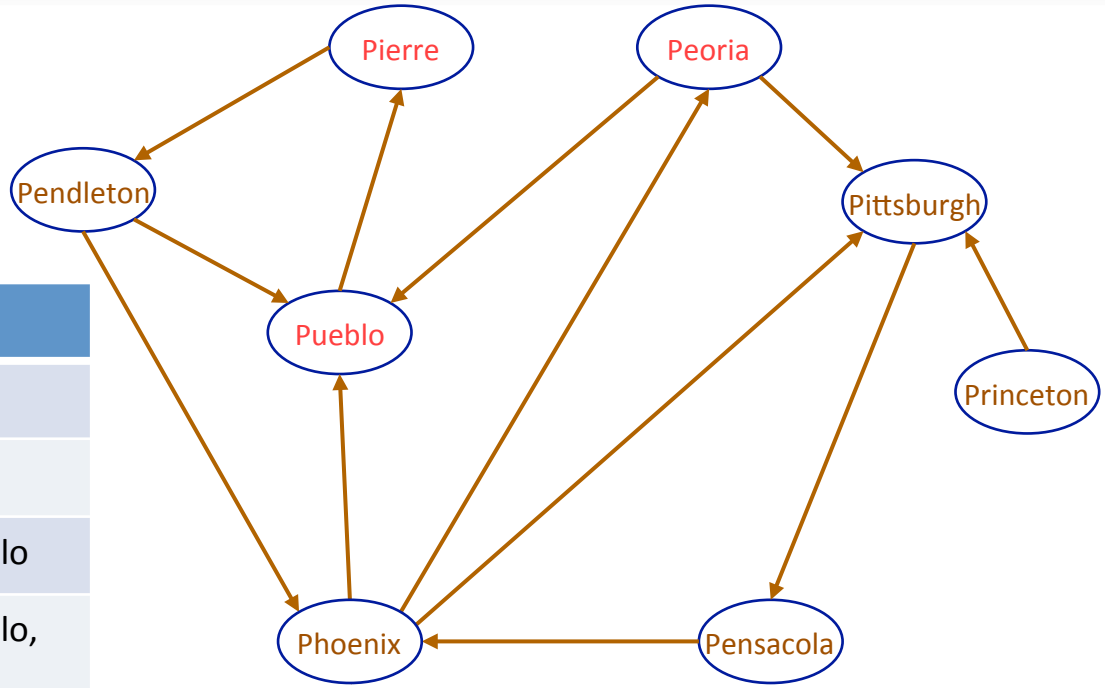
Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo

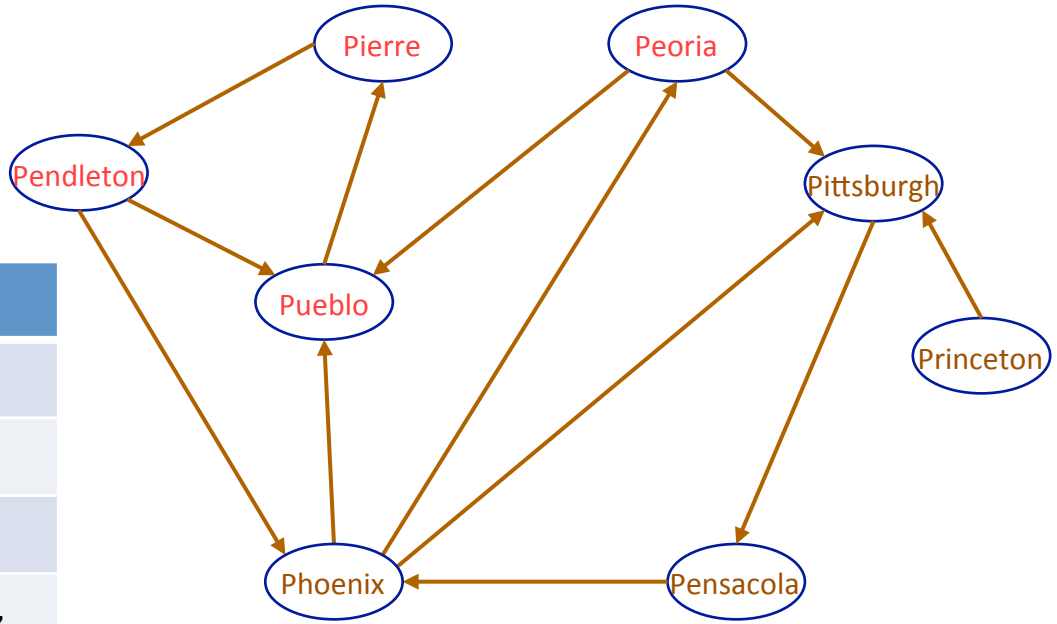


Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre



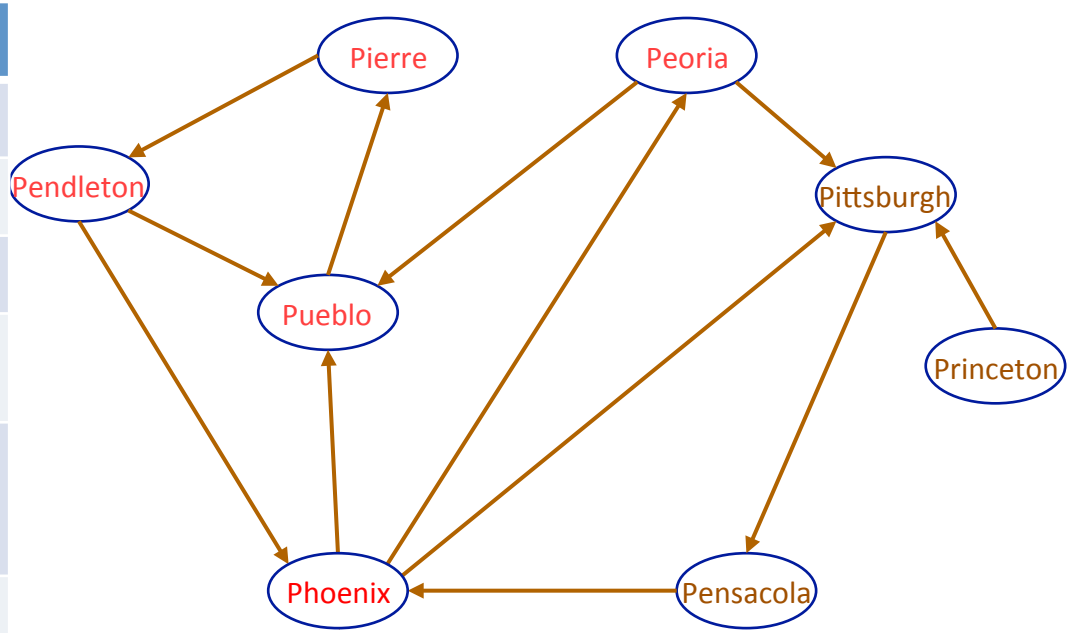
Single-Source Reachability: Stack



	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre
4	phoenix, pittsburgh	peoria, pueblo, pierre, pendleton

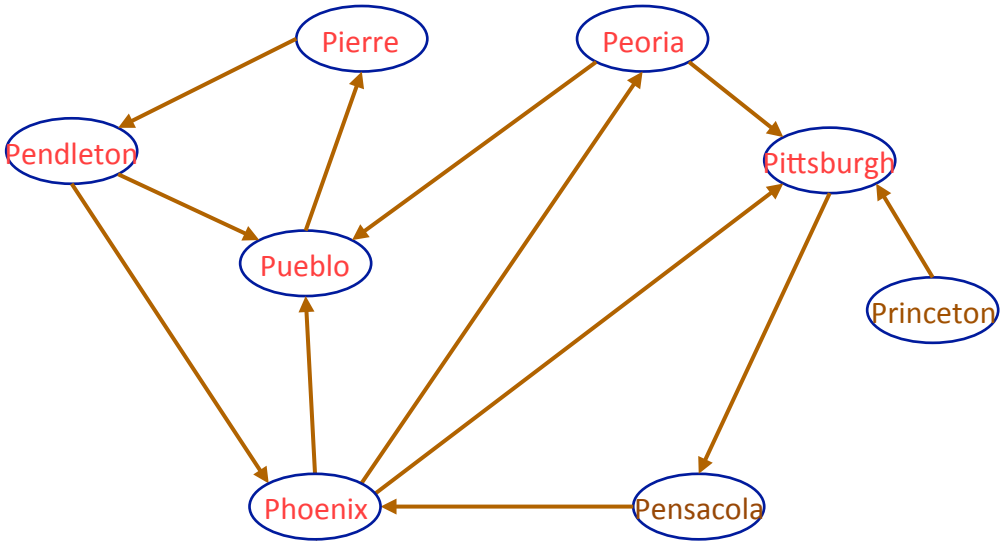
Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre
4	phoenix, pittsburgh	peoria, pueblo, pierre, pendleton
5	pittsburgh, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix



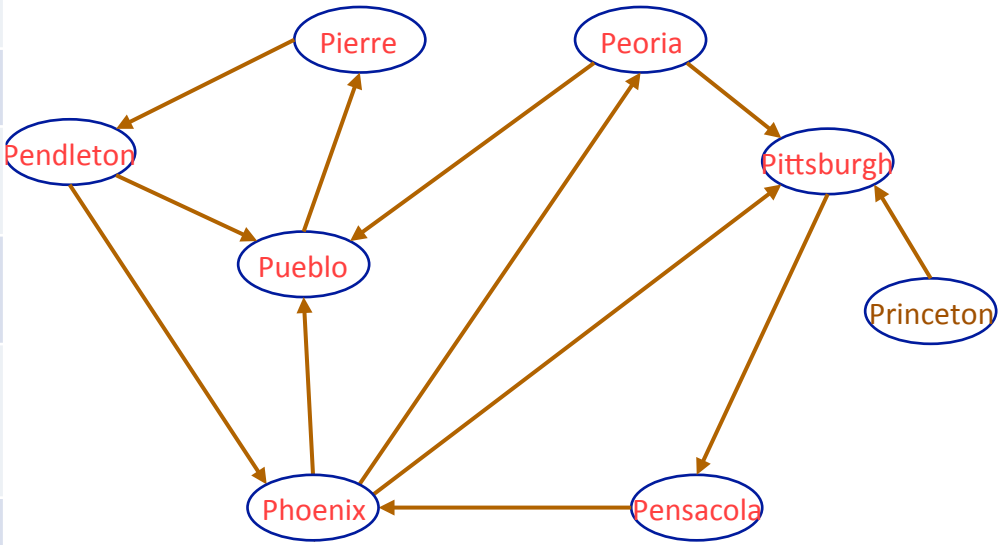
Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre
4	phoenix, pittsburgh	peoria, pueblo, pierre, pendleton
5	pittsburgh, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix
6	pensacola, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh



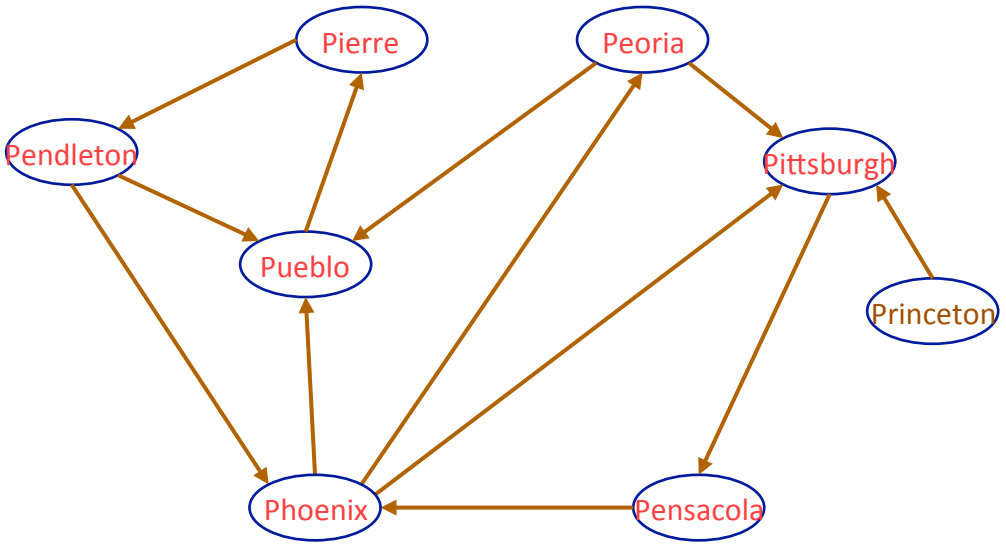
Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre
4	phoenix, pittsburgh	peoria, pueblo, pierre, pendleton
5	pittsburgh, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix
6	pensacola, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh
7	pittsburgh	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh,pensacola



Single-Source Reachability: Stack

	Stack(top->bot)	Reachable
0	peoria	{}
1	pueblo, pittsburgh	peoria
2	pierre, pittsburgh	peoria, pueblo
3	pendleton, pittsburgh	peoria, pueblo, pierre
4	phoenix, pittsburgh	peoria, pueblo, pierre, pendleton
5	pittsburgh, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix
6	pensacola, pittsburgh	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh
7	pittsburgh	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh,pensacola
8	{}	peoria, pueblo, pierre, pendleton, phoenix, pittsburgh,pensacola



Implementation

- Reachable: Any Bag Implementation
  - array, dynamic array, linked list, AVL tree (faster check for contains), HashTable
- Stack: dynamic array deque, LL Deque
- Graph Representation:
  - Dynamic array of LinkedLists
  - HashMap of LinkedLists
    - key = name of node
    - value = list of neighbors

Your Turn

- Worksheet 41
- Something to think about…
  - What happens if we use a Queue instead of a Stack to hold the unexplored neighbors?