

YOU KNOW THIS METAL  
RECTANGLE FULL OF  
LITTLE LIGHTS?

YEAH.



I SPEND MOST OF MY LIFE  
PRESSING BUTTONS TO MAKE  
THE PATTERN OF LIGHTS  
CHANGE HOWEVER I WANT.

SOUNDS  
GOOD.



BUT TODAY, THE PATTERN  
OF LIGHTS IS *ALL WRONG!*

OH GOD! TRY  
PRESSING MORE  
BUTTONS!  
*IT'S NOT  
HELPING!*



# Section 6:

## HW6

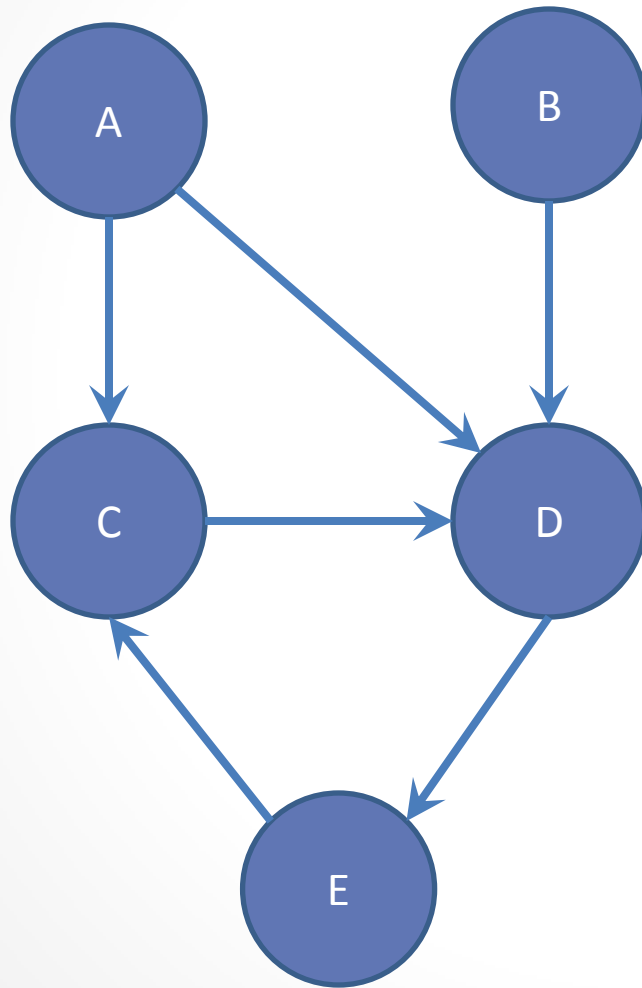
Slides by Alex Mariakakis

with material from Krysta Yousoufian,  
Mike Ernst, Kellen Donohue

# Handling Expensive RIs

- **Problem:** a thorough `checkRep()` may take a while to execute; if it is called every time the graph is modified, your code may fail the 30 second timeout per test
- **Simple solution:** use a “debug flag” boolean to turn `checkRep()` on or off
- **Fancy solution:** make multiple `checkRep()` methods of different complexity and switch between them using an enum

# Graphs



**Can I reach B  
from A?**

# Breadth-First Search (BFS)

- Often used for discovering connectivity
- Calculates the shortest path if and only if all edges have same positive or no weight
- Depth-first search (DFS) is commonly mentioned with BFS
  - BFS looks “wide”, DFS looks “deep”
  - Can also be used for discovery, but not the shortest path

# BFS Pseudocode

```
public boolean find(Node start, Node end) {  
    put start node in a queue  
    while (queue is not empty) {  
        pop node N off queue  
        if (N is goal)  
            return true;  
        else {  
            for each node O that is child of N  
                push O onto queue  
        }  
    }  
    return false;  
}
```

# Breadth-First Search

Q: <>

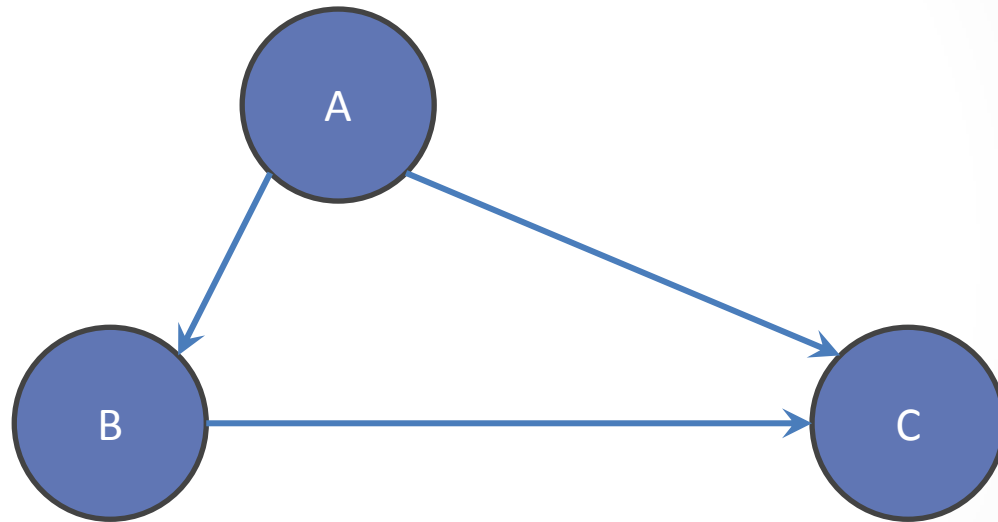
Q: <A>

Q: <>

Q: <B>

Q: <B, C>

DONE



# Breadth-First Search with Cycle

Q: <>

Q: <A>

Q: <>

Q: <B>

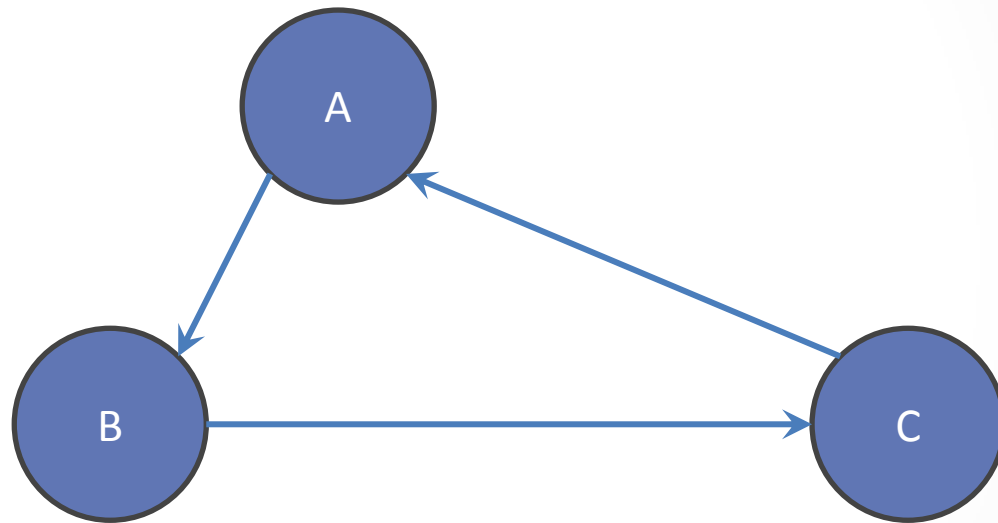
Q: <>

Q: <C>

Q: <>

Q: <A>

NEVER DONE





# BFS Pseudocode

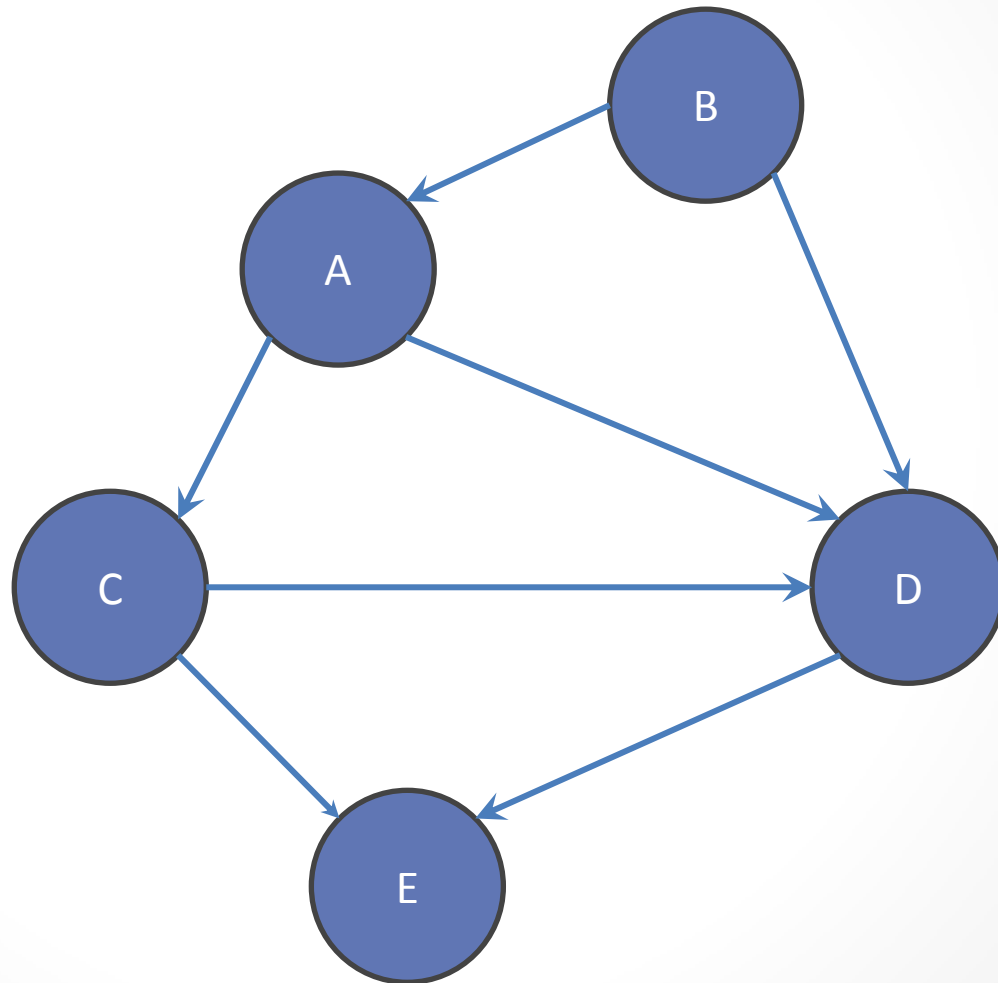
```
public boolean find(Node start, Node end) {  
    put start node in a queue  
    while (queue is not empty) {  
        pop node N off queue  
        if (N is goal)  
            return true;  
        else {  
            for each node O that is child of N  
                push O onto queue  
        }  
    }  
    return false;  
}
```

**Mark the node as visited!**



# Breadth-First Search

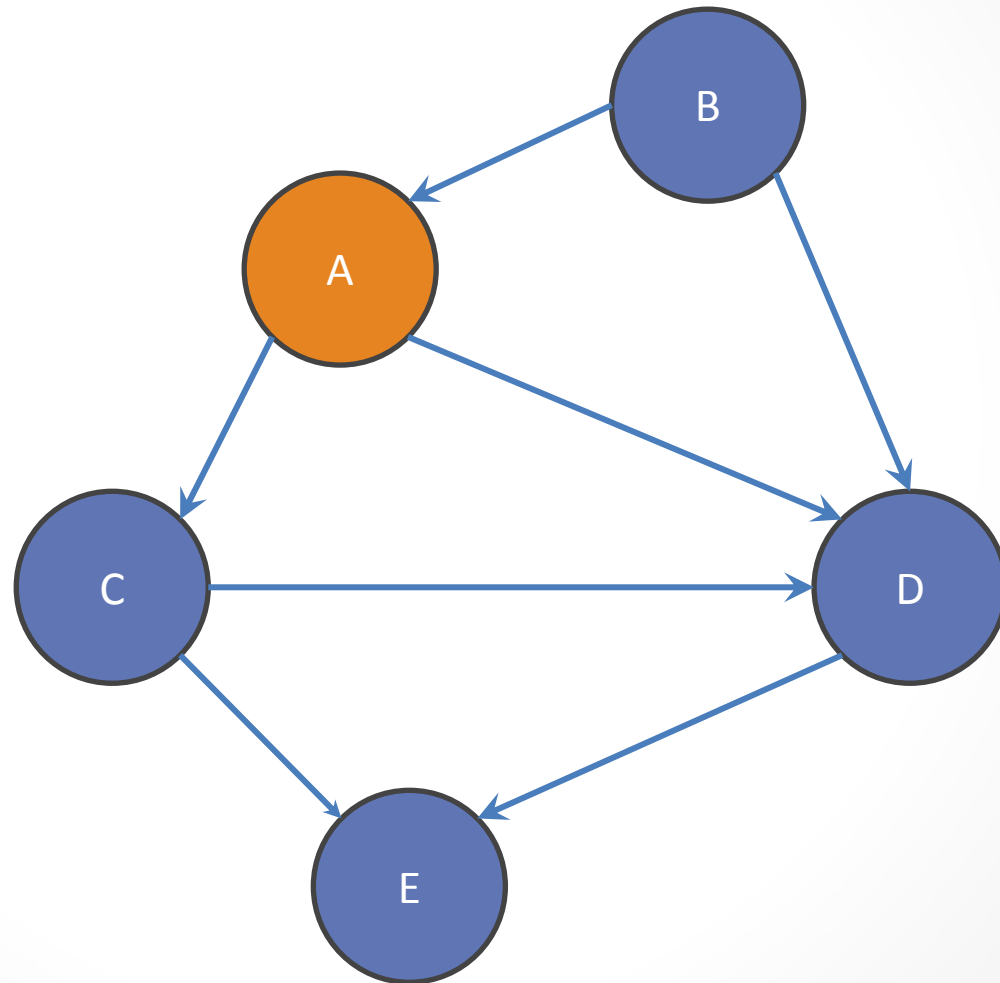
Q: <>



# Breadth-First Search

Q: <>

Q: <A>

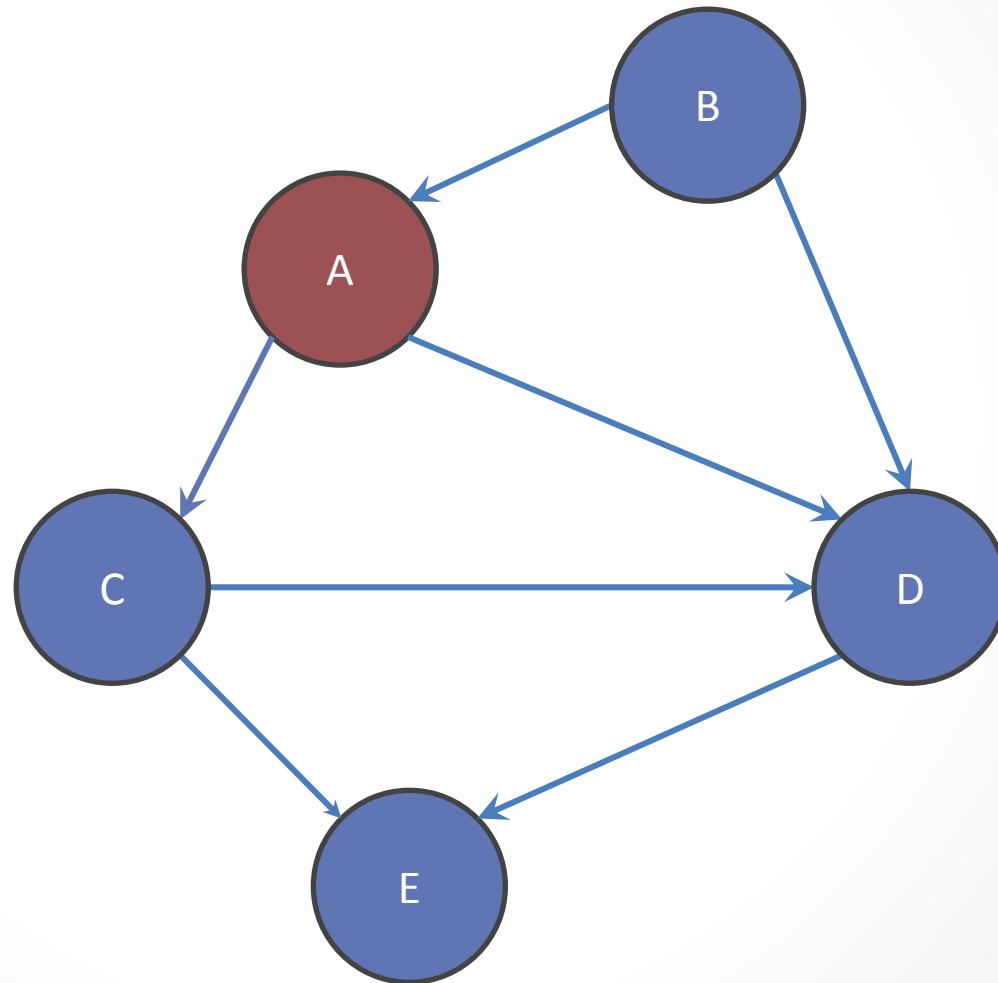


# Breadth-First Search

Q: <>

Q: <A>

Q: <>



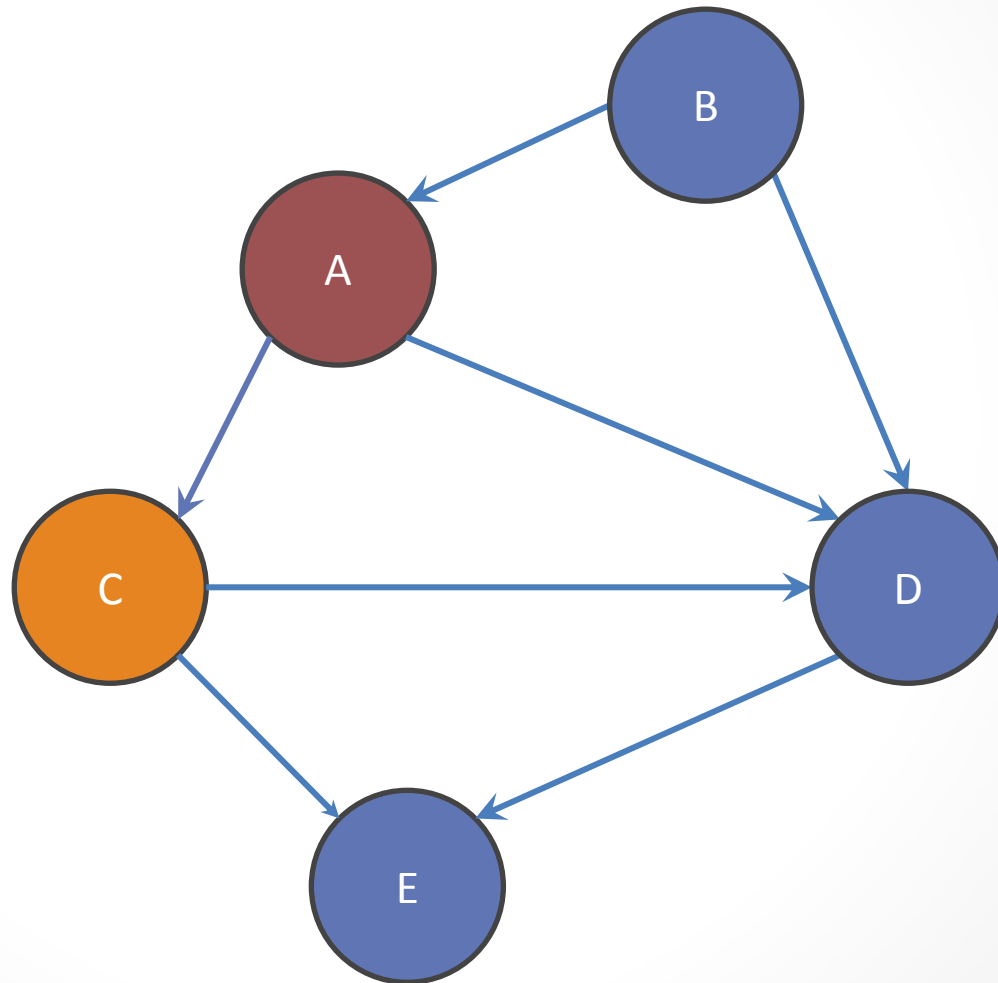
# Breadth-First Search

Q: <>

Q: <A>

Q: <>

Q: <C>



# Breadth-First Search

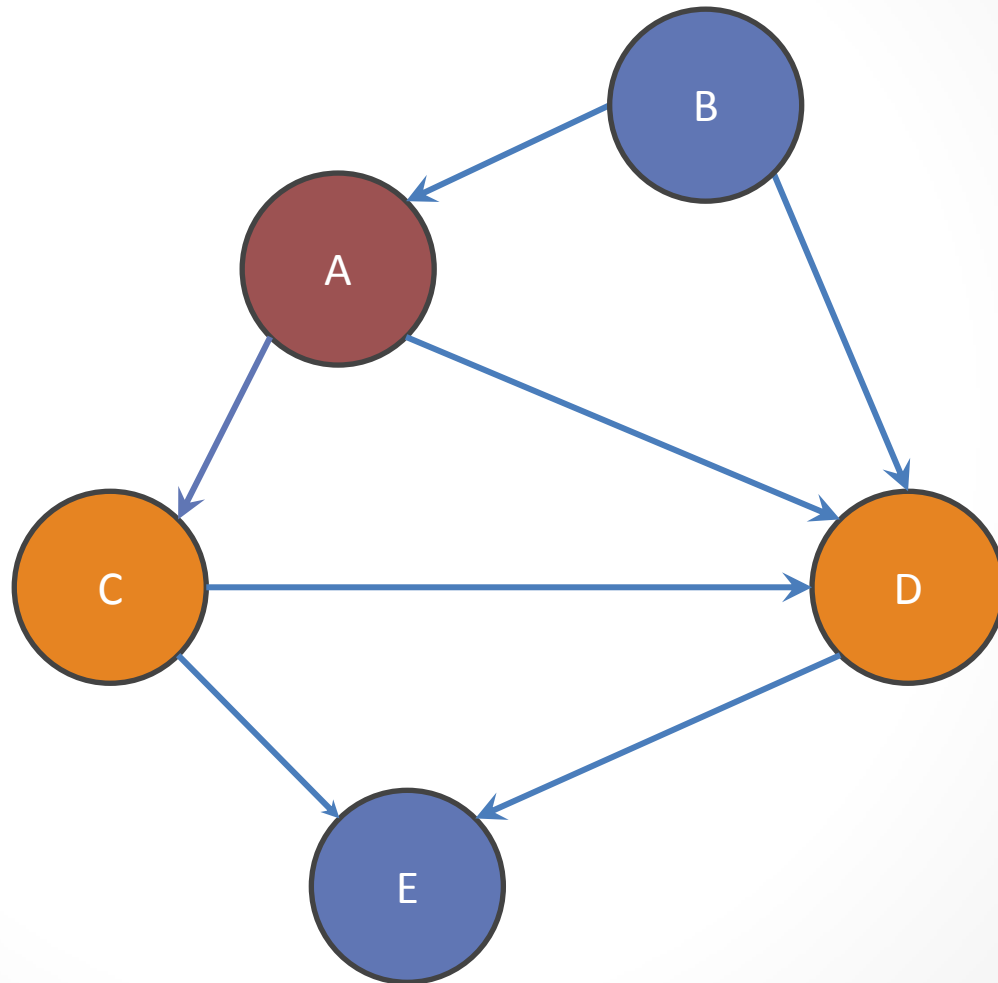
Q: <>

Q: <A>

Q: <>

Q: <C>

Q: <C ,D>



# Breadth-First Search

Q: <>

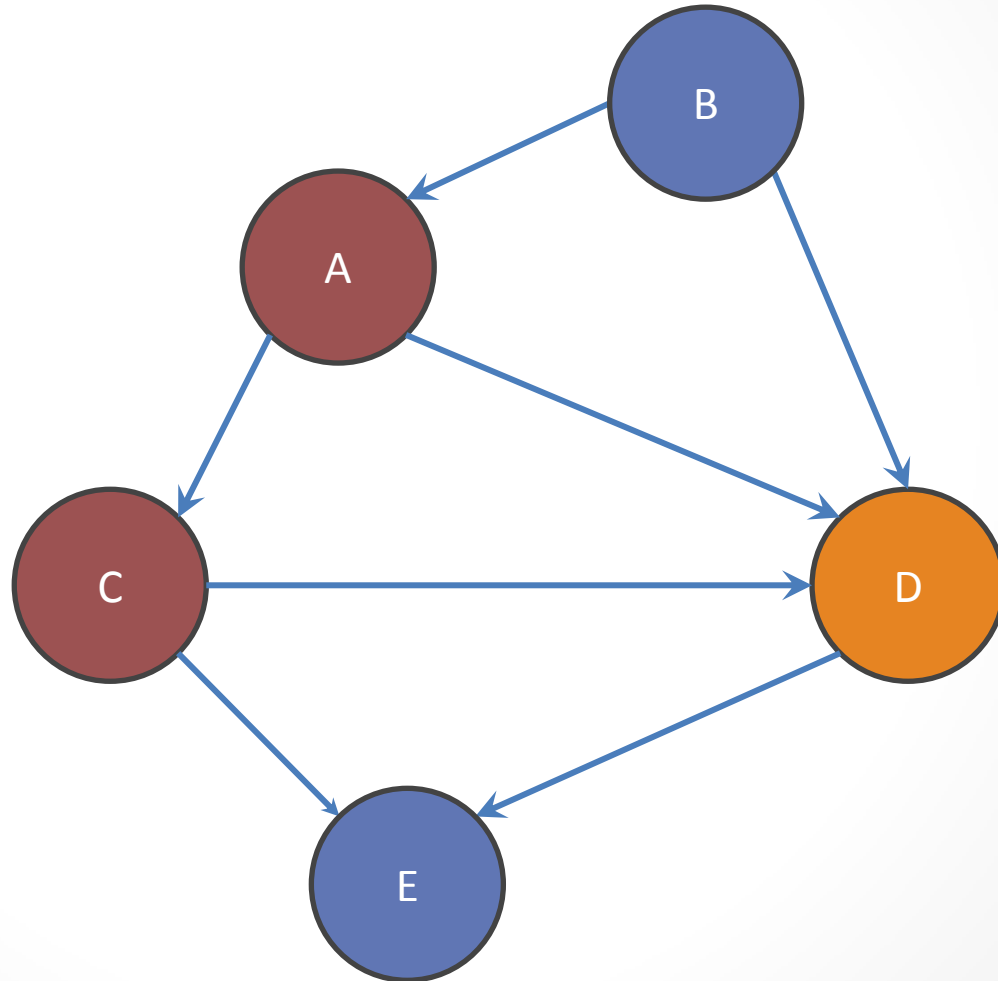
Q: <A>

Q: <>

Q: <C>

Q: <C ,D>

Q: <D>



# Breadth-First Search

Q: <>

Q: <A>

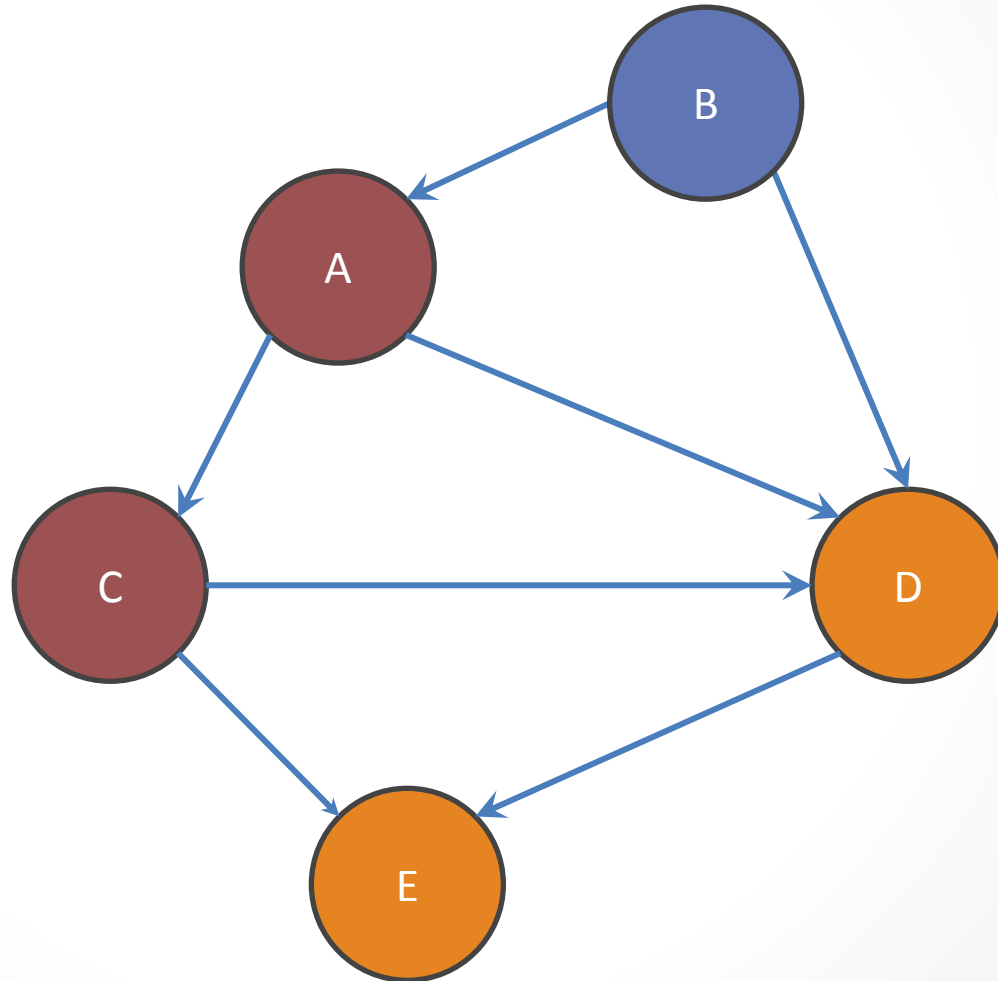
Q: <>

Q: <C>

Q: <C, D>

Q: <D>

Q: <D, E>





# Breadth-First Search

Q: <>

Q: <A>

Q: <>

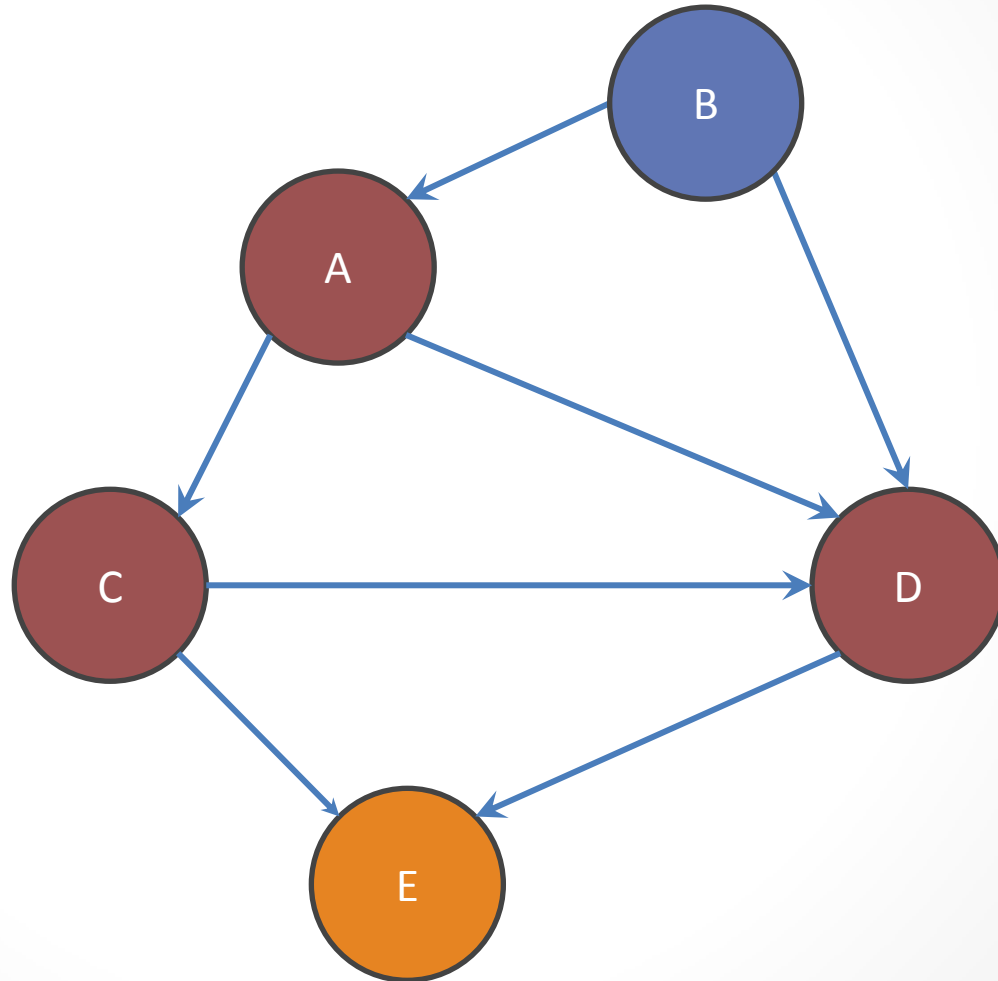
Q: <C>

Q: <C, D>

Q: <D>

Q: <D, E>

Q: <E>



# Breadth-First Search

Q: <>

Q: <A>

Q: <>

Q: <C>

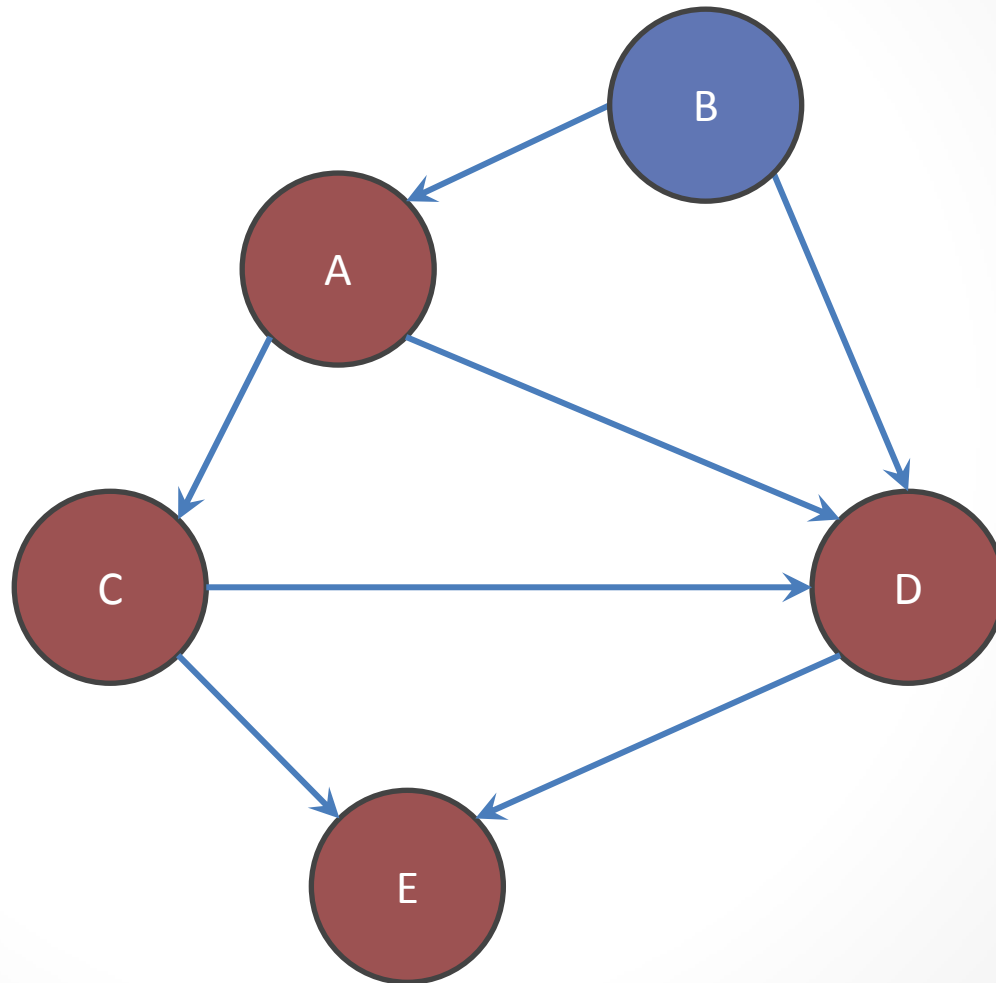
Q: <C, D>

Q: <D>

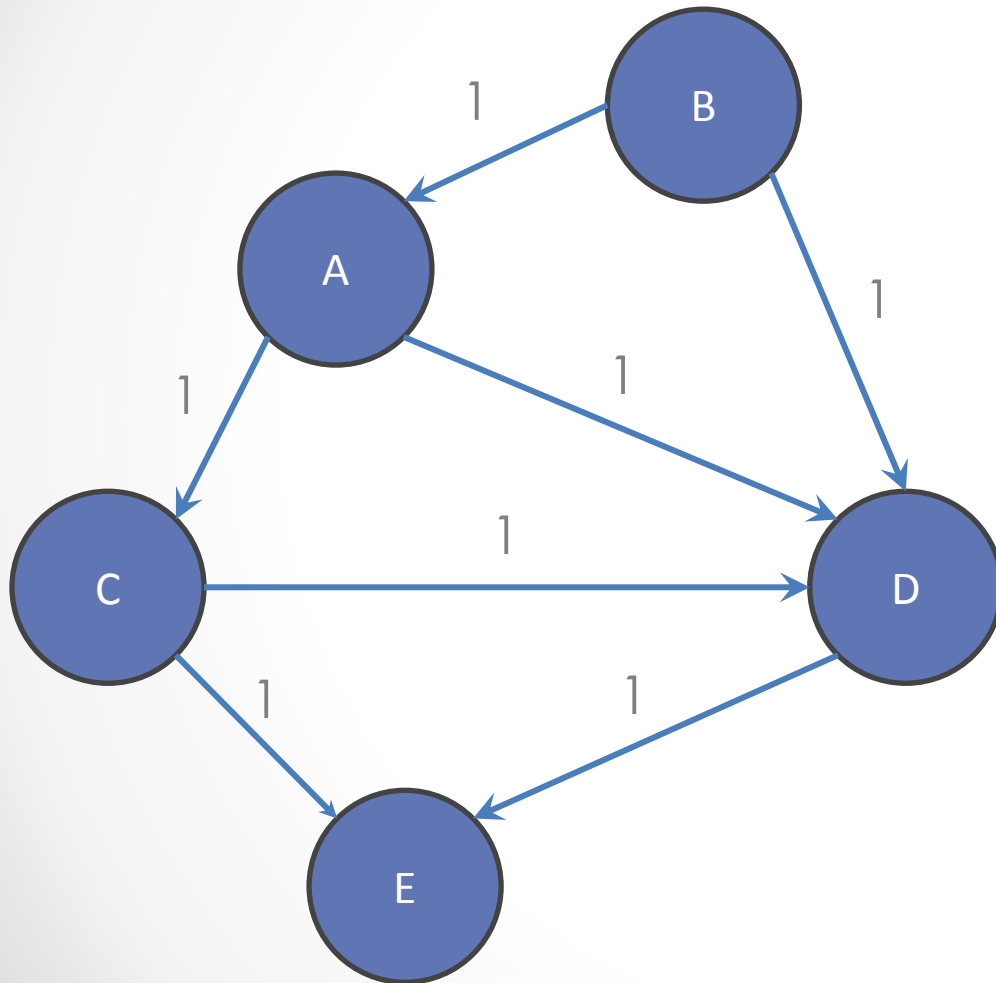
Q: <D, E>

Q: <E>

DONE



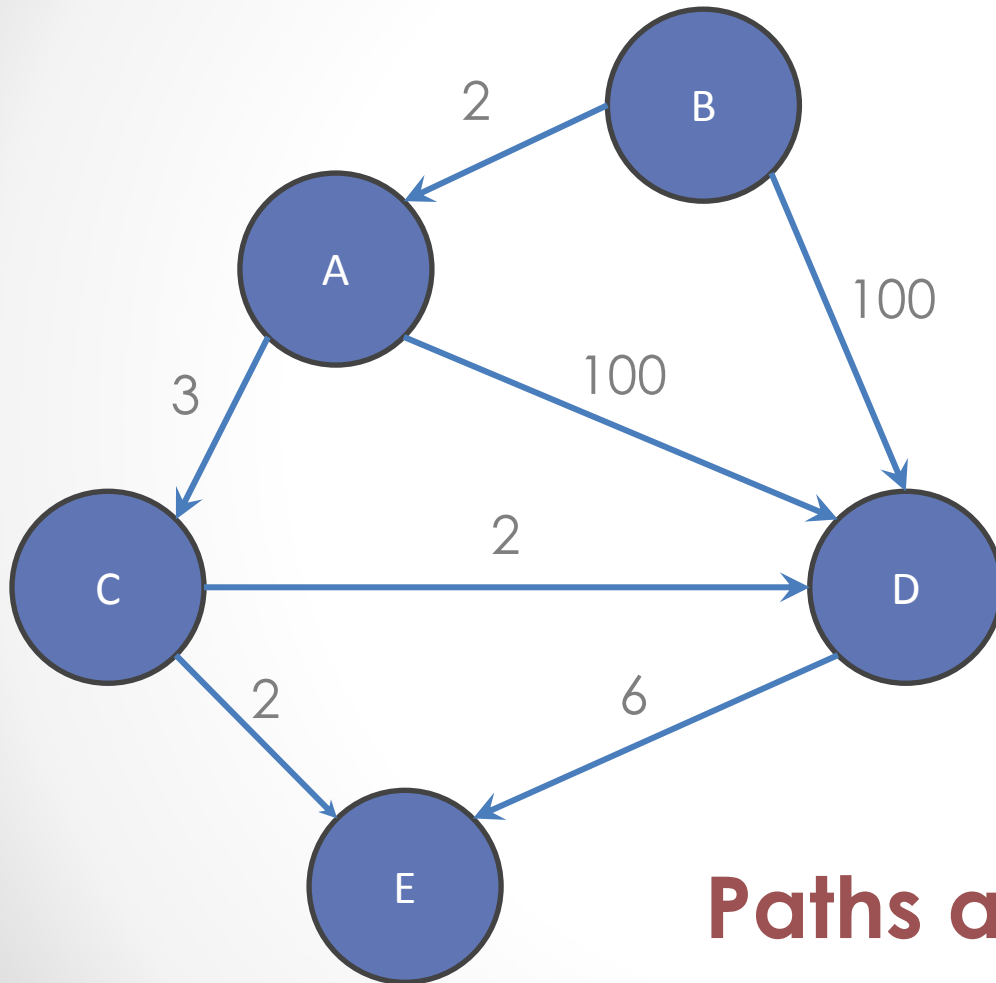
# Shortest Paths with BFS



From Node B

Destination	Path	Cost
A	<B,A>	1
B	<B>	0
C	<B,A,C>	2
D	<B,D>	1
E	<B,D,E>	2

# Shortest Paths with Weights



From Node B

Destination	Path	Cost
A	<B,A>	2
B	<B>	0
C	<B,A,C>	5
D	<B,A,C,D>	7
E	<B,A,C,E>	7

**Paths are not the same!**

# Demo

Parsing the Marvel data