the CU CS dept. is <u>not</u> affiliated with this tutoring service.

Today

Graphs:

No lecture Friday

- Traversal
  - ↳ Graph as tree
  - ↳ Breadth First
- Breadth First Search
  - ↳ shortest distance

We showed that you can traverse a graph in the print() method.

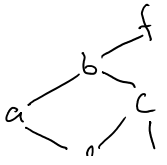
This method only prints the vertices based on what order the vector was populated.

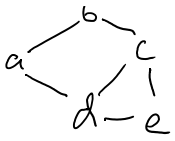
What if we want to know some more info about a particular vertex.

How the vertex relates to the rest of the graph? How many degrees of separation b/w 2 vertices? What is the shortest path b/w two vertices?

E.G. google maps, facebook, flight routing

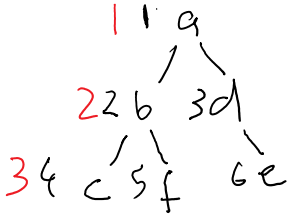
Say, given this graph:





How does "a" relate to all other vertices separation-wise?

Redraw the graph as a tree with "a" as root.



find(c) w.r. a

following breadth first,

a → b

a → c 1 edge away

not following breadth first

a → b → c 2 edges away

not shortest path

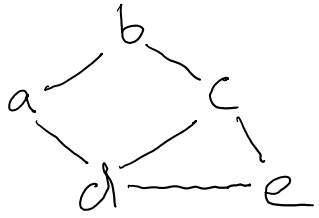
Lets update our vertex struct definition keep track of visited nodes.

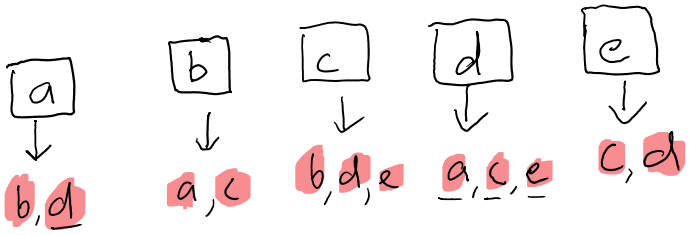
```
struct vertex
{
    string key;
    vector<adjVertex> adj;
    bool visited;
}
```

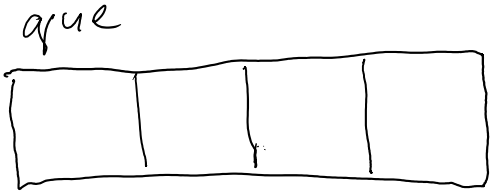
e.g. bFT(a)

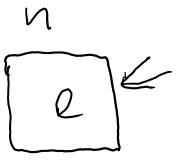
```
}

breadthFirstTraverse ( value )
    vertex = search (value);
    print (vertex.key) ;
    vertex.visited = true;
    que.enque (vertex)
    while (! que.isEmpty ())
    {   n = que. Deque;
        for(x=0 to n.adj.end)
        if(!n.adj[x].visited)
        {   n.adj[x].v.visited = T;
            print(n.adj[x].v.key)
            q.enque(n.adj[x].v)
        }
}
```

e.g. bFT(a)









print

a, b, d, c, e

vertex breadthFirstSearch (startValue, searchValue)

a , c

```
struct vertex
{   string key;
    vector<adjVertex> adj;
    bool visited;
    int distance;
}
```