



App Academy

Intro to Graphs



What exactly are graphs?

- We've actually been working with graphs since day 1!
 - linked lists
 - all types of trees
- A graph is simply a group of connected nodes
- Unlike linked lists and trees, the graphs we are talking about come in all sorts of different shapes and can be represented in a number of ways.
 - For example, cyclic graphs
- Almost no restrictions



Ways to represent Graphs and Terminology

- You may hear people referring to nodes and pointers as **vertices** and **edges** respectively instead. These are pretty much synonymous.
- Graphs can be **directed or undirected**, referring to whether edges point one way or both ways
- Edges and vertices have a relationship that helps us better understand their Big O complexities:
 - $E \leq V^2$, where E = edges and V = vertices
- Ways to represent graphs:
 - Matrix
 - Adjacency Matrix
 - Adjacency List



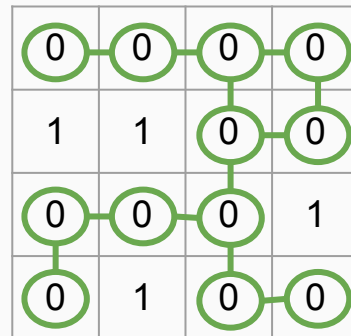
Matrices

- Graphs represented as a 2D array
- Typically an 4-directional undirected graph
- Coordinates represented with row and column indices
 - We can find values by using `grid[row][col]`
 - Some also use `x` and `y`, but I recommend against this because I've seen many people get confused during mock interviews.
- Commonly used for path representation

```
grid = [ [0, 0, 0, 0],  
         [1, 1, 0, 0],  
         [0, 0, 0, 1],  
         [0, 1, 0, 0] ]
```

0 = free

1 = blocked

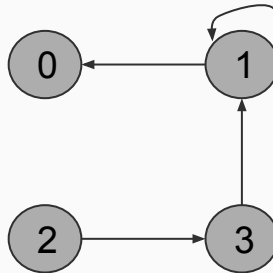


Adjacency Matrices

- The cells of a adjacency matrix are NOT nodes
- Instead, the indices represent vertices, and the values in the cells represent edges between vertices
- Always a square since both sides represent vertices

- Examples

- $\text{adjMatrix}[1][2] === 1$
 - An edge exists from vertex 1 to vertex 2
 - $\text{adjMatrix}[2][1] === 1$
 - An edge exists from vertex 2 to vertex 1
 - Order matters!!
 - $\text{adjMatrix}[0][1] === 0$
 - No edge exists from vertex 0 to vertex 1
 - $\text{adjMatrix}[1][1] === 1$
 - There exists a self looping edge at vertex 1
- Rare because it is space inefficient. Complexity is $O(V^2)$



$\text{adjMatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

	0	1	2	3
0	0	0	0	0
1	1	1	0	0
2	0	0	0	1
3	0	1	0	0



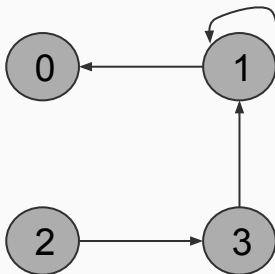
Adjacency Lists

- Very common way to represent a graph
- Uses nodes, similar to linked lists or trees
- Unlike linked lists or trees, there is no predefined number of connected neighbors
- We can represent neighbors in an array or set
- Much more space efficient than adjacency matrix since we only represent nodes that actually exist.
- Sometimes may have to build the adjacency list ourselves

```
class GraphNode {  
    constructor(val) {  
        this.val = val;  
        this.neighbors = [];  
    }  
}
```

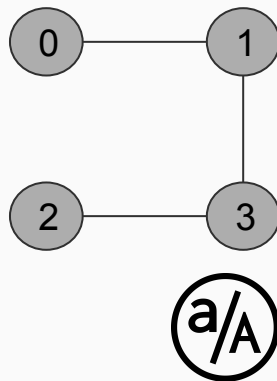
directed graph:

```
graph = {  
    0: [],  
    1: [0,1],  
    2: [3],  
    3: [1]  
}
```



undirected graph:

```
graph = {  
    0: [1],  
    1: [0,3],  
    2: [3],  
    3: [1,2]  
}
```



Questions?

