

# E-LeetCode: Graphs Session

UW  
DATA SCIENCE  
CLUB.



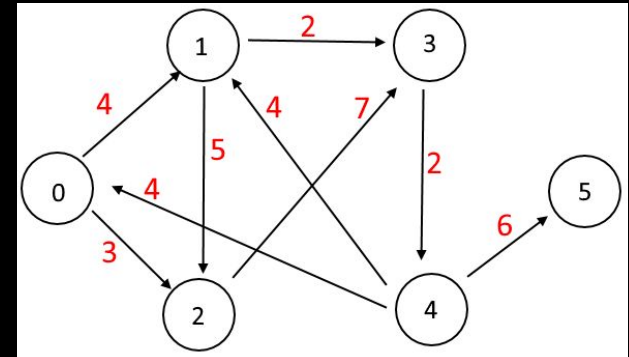
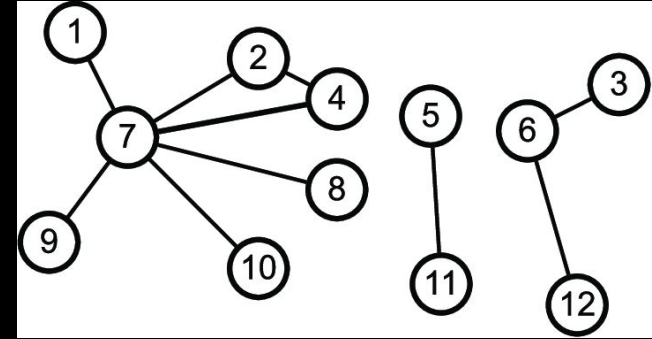
Presented by Jack Douglas

# Session Outline

1. Graph Basics
  - a. What is a graph?
  - b. How are they stored?
  - c. Space and Time Complexity Comparison
2. Types of Graph Questions
3. Depth First Search (DFS)
  - a. What is DFS?
  - b. Pseudocode
4. Number of Islands LeetCode Problem
  - a. Discuss our approach
  - b. Implement!

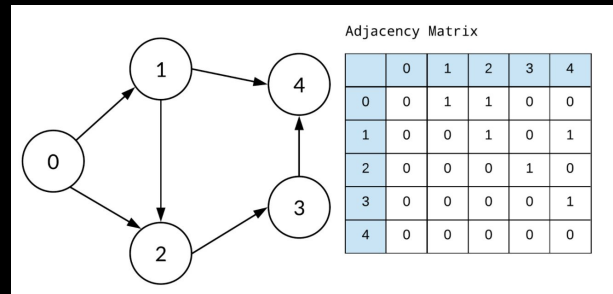
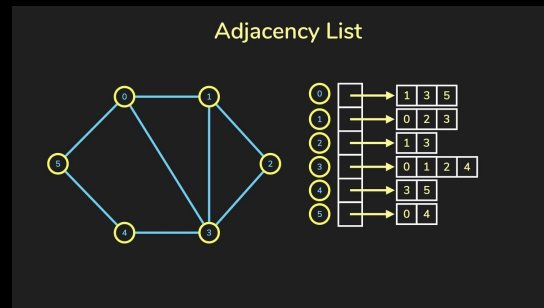
# What is a graph?

- Vertices/Nodes (usually denoted  $V$ )
  - Collection of numbers, letters, coordinates, or any other object
- Edges (usually denoted  $E$ )
  - Connect two vertices
  - Can be weighted or unweighted
  - Can be directed and undirected
- Graph vs. Tree
  - Graph is non-linear
  - Graph has no root



# How are graphs stored?

- Vertices are stored in set/list
- Edges usually stored in 1 of 2 ways
  1. Adjacency List
    - Implemented as dictionary with the key being a vertex and the value being a list/set of the vertex's neighbours
  2. Adjacency Matrix
    - Implemented as a  $|V| \times |V|$  list of lists



# Space and Time Complexity Comparison

- **Adjacency List**
  - Implemented as dictionary with the key being a vertex and the value being a list/set of the vertex's neighbours
- **Adjacency Matrix**
  - Implemented as a  $|V| \times |V|$  list of lists

$|V|$  - Number of Vertices &&  $|E|$  - Number of Edges

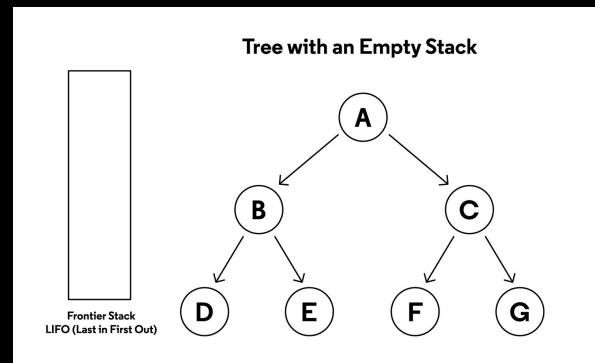
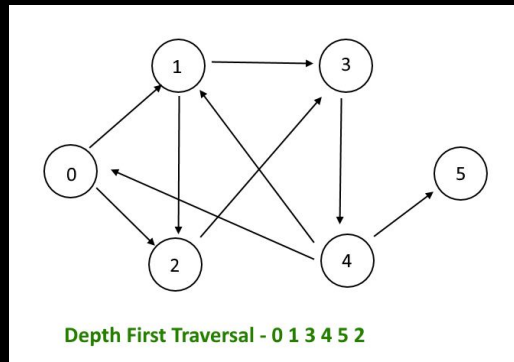
Operation	Adjacency List	Adjacency Matrix
Add Vertex	$O(1)$	$O( V^2 )$
Add Edge	$O(1)$	$O(1)$
Remove Vertex	$O( V  +  E )$	$O( V^2 )$
Remove Edge	$O( E )$	$O(1)$
Query	$O( V  +  E )$	$O(1)$
Storage	$O( V  +  E )$	$O( V^2 )$

# Types of Graph Questions

1. Depth First Search (DFS)
2. Breadth First Search (BFS)
3. Topological Sorting
  - a. <https://www.youtube.com/watch?v=eL-KzMXSXXI>
  - b. <https://www.interviewcake.com/concept/java/topological-sort>
4. Union-Find
  - a. <https://www.youtube.com/watch?v=ayW5B2W9hfo>
  - b. <https://algorithms.tutorialhorizon.com/disjoint-set-union-find-algorithm-union-by-rank-and-path-compression/>

# What is Depth First Search (DFS)

- Start at a vertex and travel as far down a path as you can, then backtrack until you find an unexplored path
- It can be implemented iteratively with a stack or recursively
- Applications
  - Detecting cycles in directed
  - Topological sorting of directed acyclic graphs (DAG's)



# DFS Recursion Pseudocode

Typical function signature looks like this:

dfs(current vertex, adjacency list, visited vertices set)

Pseudocode:

1. Add current vertex to visited vertices set
2. Iterate through neighbours of current vertex using adjacency list
3. If a neighbour has not been visited yet, recursively call dfs with the neighbour:

ie. dfs(neighbour, adjacency list, visited vertices set)



# Problem Statement

Number of Islands:

<https://leetcode.com/problems/number-of-islands/description/>

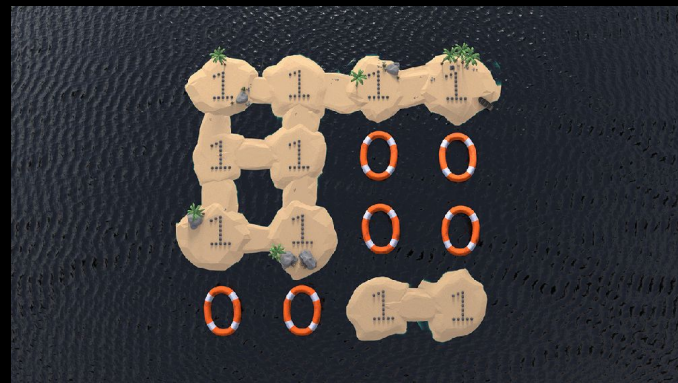
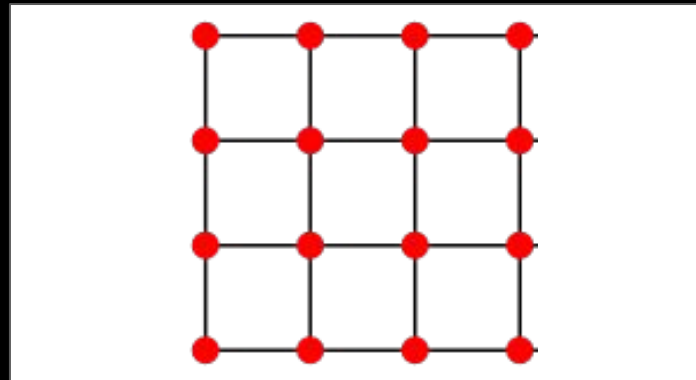
- Given an  $m \times n$  grid which contains 0's and 1's
  - 0 is water, 1 is land
- Return the # of islands
- How do we make this a graph question?

1	1	0	1	1
1	1	0	0	0
0	1	0	0	0
0	1	0	0	1
1	1	0	1	1

No of Islands: 3

# Identifying the Vertices and Edges

- Each land cell in the grid is vertex!
- Adjacent land cells represent edges!
- How will we use DFS?  
When will we call DFS?



# Key Observations

We're going to iterate through every cell in the grid

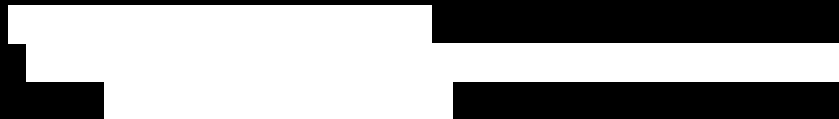
1. What do we do if the current cell is water?



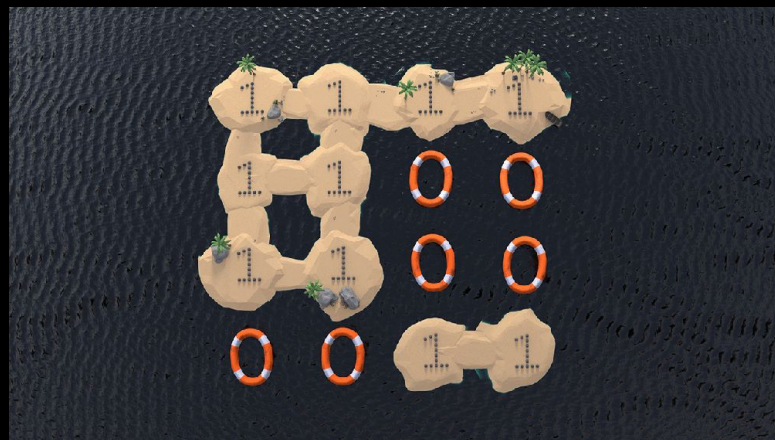
2.



3.



4.



# Key Observations

We're going to iterate through every cell in the grid

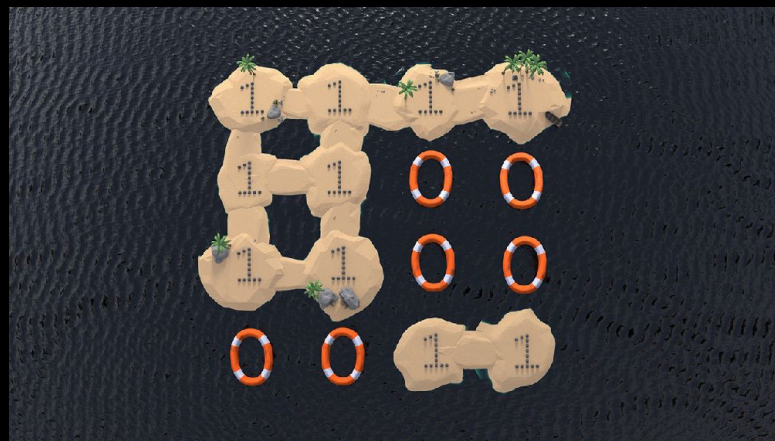
1. What do we do if the current cell is water?
  - a. **A: Nothing!**

2. What happens when we call DFS on a land cell?

[Redacted]

3. [Redacted]

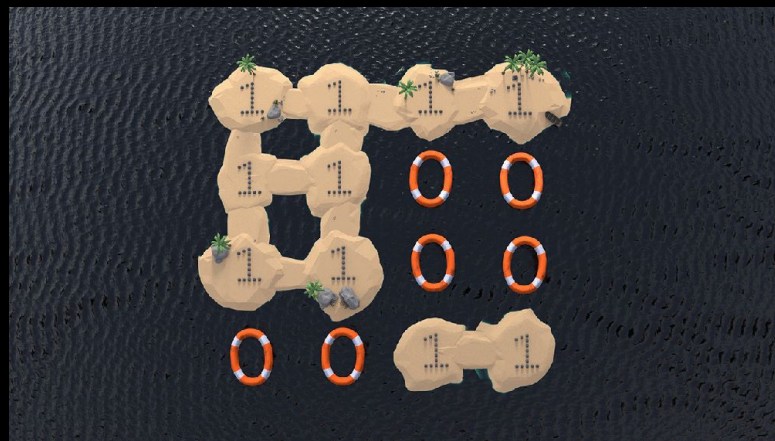
4. [Redacted]



# Key Observations

We're going to iterate through every cell in the grid

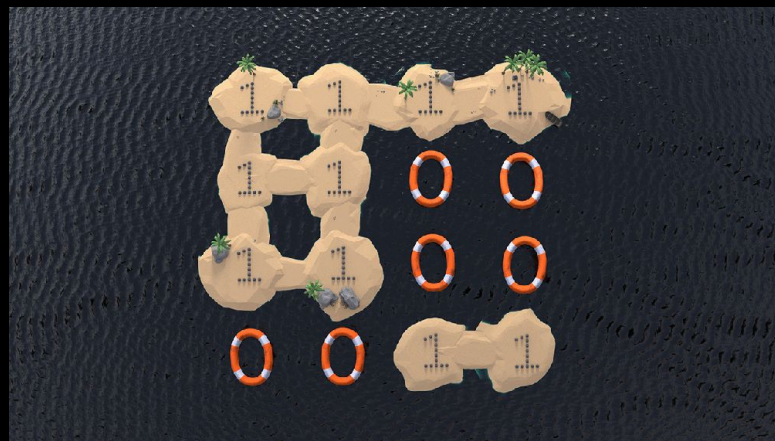
1. What do we do if the current cell is water?
  - a. A: Nothing!
2. What happens when we call DFS on a land cell?
  - a. A: We visit every land cell in the island!
3. When do we call DFS?
4.



# Key Observations

We're going to iterate through every cell in the grid

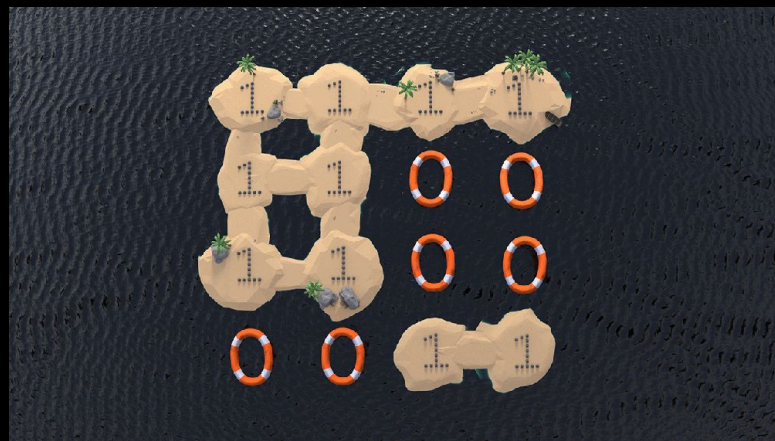
1. What do we do if the current cell is water?
  - a. A: Nothing!
2. What happens when we call DFS on a land cell?
  - a. A: We visit every land cell in the island!
3. When do we call DFS?
  - a. A: We call DFS when it's a land cell that we haven't visited!
4. What is the significance of this?



# Key Observations

We're going to iterate through every cell in the grid

1. What do we do if the current cell is water?
  - a. A: Nothing!
2. What happens when we call DFS on a land cell?
  - a. A: We visit every land cell in the island!
3. When do we call DFS?
  - a. A: We call DFS when it's a land cell that we haven't visited!
4. What is the significance of this?
  - a. A: We will be calling DFS at every new island!



# Let's Implement!

Number of Islands:

<https://leetcode.com/problems/number-of-islands/description/>

Solution:

<http://bit.ly/3UFCOav>



# Final Advice & Resources

- DFS and BFS can solve many problems and many of the same problems
  - For practice, implement the recursive and iterative for both
- More Graph Questions + Other LeetCode Questions
  1. Blind 75: <https://leetcode.com/discuss/general-discussion/460599/blind-75-leetcode-questions>
  2. Grind 75: <https://www.techinterviewhandbook.org/grind75>
  3. Neetcode: <https://neetcode.io/practice>