# Week B Notes

## Data Structures

* What is a data structure
  + A data organization management and storage format that enables efficient access and modification
* Why use them
  + Organization of data
  + Efficient modification
  + Convenient operation
  + Some something access
* Linear vs Hierarchical
  + Linear structures (arrays, linked lists, stacks, queues)
    - Elements are arranged in a sequence and have some sort of ordering
    - Elements within the data structure are not necessarily related to each other
    - Simple structures makes them easier to implement
  + Hierarchical structures (trees, graphs, heaps)
    - Elements are not organized in a sequential manner
    - Elements are sorted within the structure based upon their relation to each other
    - More complex structure makes them harder to implement
* Basic Data Structure Operations
  + Inserting
  + Searching
  + Sorting
  + Deleting
  + Traversing
  + Merging

## Linear Structures

* Arrays
  + Most basic data structure, often used as a part of more complex data structure
  + Holds a fixed number of elements, which can be accessed by their position in the array, or index
  + Elements are homogenous (of the same type)
* Linked List
  + Technically, a sequence of data structures called nodes
  + Nodes contain the data being stored, as well as a reference to the next node in the structure
    - These node references are the links in the “linked” list
  + The node at the beginning of the linked list is known as the head
    - Unsurprisingly, the node at the end is known as the tail
  + Types of linked lists
    - Simple linked list
    - Doubly linked list
    - Circular linked list
* Stacks
  + Very much like a real-world “stack”
  + Operations on the stack only apply to the element on the top of the stack; LIFO
  + Inserting an element onto the stack is known as pushing
    - Kdf
    - Fsa
* Queues
  + Again, very similar to their real-world counterpar
    - Think of waiting queue at the bank or grocery store
  + The structure is open at both ends, new elements are added to the end and removals take place at the front FIFO

## Hierarchical Structures

* Trees
  + Very common structure
    - OS File system structure
    - In-memory representation of an HTML document
  + Tree components/concepts
    - Nodes
    - Edges
    - Parent
    - Leaf
    - Parent
    - Child
    - Sibling
    - Path/Branch
  + Trees do not allow siblings to fuck
* Graphs
  + A more general, less-restricted tree
    - If anything, trees are specific type of graph
  + Can be used to represent complex relationships between entities
  + Graph components/concepts
    - Vertex
    - Edge
    - Weight
    - Paths
    - Cycles
* Heaps
  + A specialized tree structure, which is a binary tree
    - Binary tree – is a tree in which each node has at most 2 child nodes
  + Two primary types of heaps
    - Max-heap – the root node’s value is the max value in the structure
    - Min-heap – the root node’s value is the min value in the structure
  + Used with a sorting algorithm known as heapsort
* Side Note
  + Data structures are not independent and separate from one another
    - Many data structures are actually comprised of other data structures
  + Think about what data structure will best fit you needs when working with data within your applications

## Big O Notation

* Big – O
  + Worst-case time/space complexity
* Big Θ (Theta)
  + Avg-Case time/space complexity
* Big Ω (Omega)
  + Best-case time/space complexity
* Big O complexity
  + O(1) - Constant
  + O(log(n)) - Logarithmic
  + O(n) - Linear
  + O(n\*log(n)) – Log-linear
  + O(n^2) - Exponential
  + O(2^n) - Polynomial
  + O(n!) - Factorial