**FCC Intro to Data Structures and Algorithms**

**The Linked List:**

* Sequential access linear data structure in which every element is a separate object called a “Node”, which has 2 parts:
  + The data
  + The reference (or pointer) which points to the next Node in the List
* Sequential access is similar to the Stack or Queue…can only access them in a particular way…not random access like a list
* Linear means they are linked, one right after the next
* Node holds two pieces of information:

|  |  |
| --- | --- |
| * Data | * Reference/pointer to next * Node |

* Data is where the strings, Booleans, integers, etc. are stored
* Reference/pointer is a reference to the next Node in the Linked List

**Setting Up A Linked List:**

* Starts with a Head Node: **[Data:1 |Reference: null]** when no other Nodes have been created. Add a Node:
* **HN = [1 | ] [2 | ] [3 | null]** …..last Node is Tail Node
* So Tail Node always has a null value; tells computer that we are at the end of the Linked List

**Adding and Removing Elements From a Linked List:**

* Data can flow in and out of any point of a Linked List
* **Three Ways to Add/Remove from a Linked List:**
  + Add/Remove from Head
  + Add/Remove from Middle
  + Add/Remove from Tail
* Whenever we change a Node in a Linked List, we also have to change its pointers, which can get complicated
* **Basic Linked List:**
  + **HN = [1 | ] [2 | ] [3 | null]**
* **Adding to the Head of a Linked List:**
  + All that needs to be done is to set the new Node’s pointer to point to the former Head of the Linked List
* **Removing from the Head of a Linked List:**
  + Set the Head Node’s pointer to a null value, and it gets cut off
* **Adding a Node to the Middle of a Linked List:**
  + Make the pointer of the new Node point to the Node after the location we want to insert at
  + Set the Node before the location we want to insert at, to point to the new Node
* **Removing a Node from the Middle of a Linked List:**
  + Make the pointer that points to the soon to be removed Node, point to the Node after the one that is being removed
* **Adding to the Tail of a Linked List:**
  + Make the current tail point to the new Node you are adding
* **Removing from the Tail of a Linked List:**
  + Make the object before the tail point to a null value, signifying that it is the end