Linked Lists

1. Linked Lists can grow and shrink automatically.
2. Each item is a node which stores.
   1. The value
   2. The address of the next node
   3. In other words, each node points to or references the next node.
3. First node = Head
4. Last node = Tail
5. Time Complexity
   1. Lookup = O(n)
   2. By Index = O(n)
   3. Insertion = Depends where we want to insert
      1. At the end
         1. We simply need to create a new node and have the Tail point to it.
         2. O(1)
      2. At the beginning
         1. We simply need to create a new node that points to the Head.
         2. Then make the new node the Head.
         3. O(1)
      3. In the middle
         1. We’d have to traverse the list so O(n)
         2. Then we’d have to update the links which is an O(1) operation.
         3. O(n)
   4. Delete
      1. At the beginning
         1. Move the head to the next node and remove the link from the previous head so it doesn’t reference the second node anymore.
         2. O(1)
      2. At the end
         1. We can’t go backwards in a Linked List so we have to traverse from the start.
         2. Stop at the node previous to the last node, and delete its reference to it.
         3. Have the tail point to the previous node.
         4. O(n)
      3. In the middle
         1. Traverse the list to find the node to delete.
         2. Set the pointer of the previous node to the next node.
         3. O(n)