CSE 250 Recitation

September 16~17: PA1, Lists, Arrays, and Code Analysis

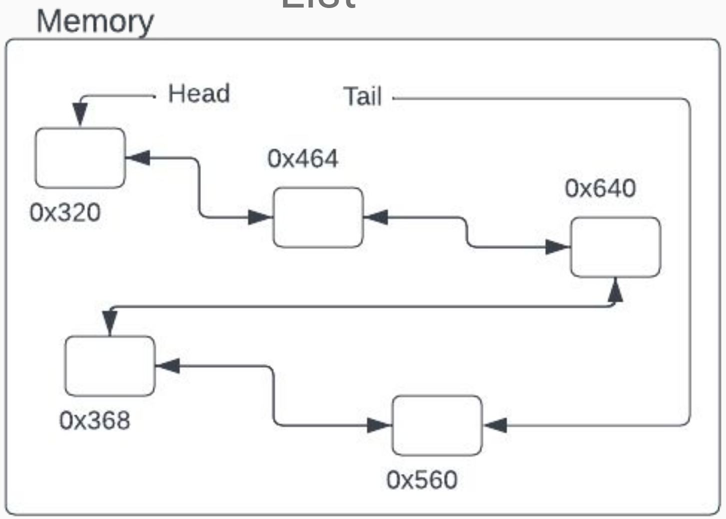
PA1: Implementation

- **PA1** has us implementing a sorted doubly linked list
- Draw out some examples of linked lists that meet the specifications of **PA1**
- Now that you have some example lists, draw out what happens when we:
    - Insert a value into the linked list
    - Remove a value from the linked list
    - Find a value in the linked list
    - Find an element at a specific position

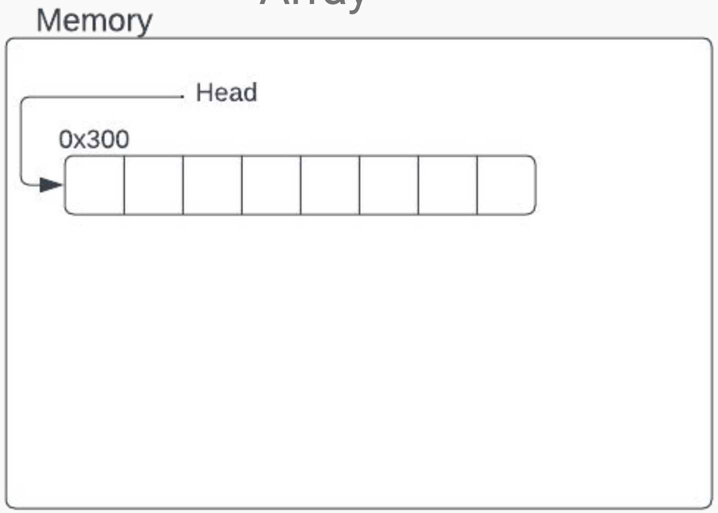
Focus on understanding the process, drawing it out visually.
Don't worry about code. See page 8 of the handout for verification.

Linked Lists vs. Arrays

List



Array



Linked Lists vs. Arrays

Key features of Linked Lists:

- The list is made up of nodes scattered throughout memory
- In a singly linked list a node will only carry a reference to the next node
- In doubly linked list a node will hold a reference to the next and previous nodes in the list
- The only way to find a node in the list is to traverse each element (unless you already have a reference to that node)
- Linked Lists will also hold a reference to the head and (usually) the tail

Linked Lists vs. Arrays

Key features of Arrays:

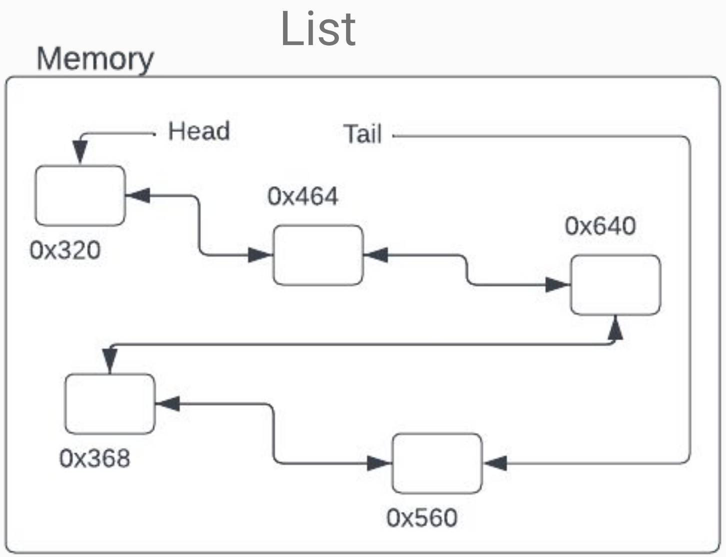
- Arrays are made of one continuous chunk of memory
- Can find an index by doing addition on the array's starting address
- Indices only need to hold the value (no need to carry references to other nodes)

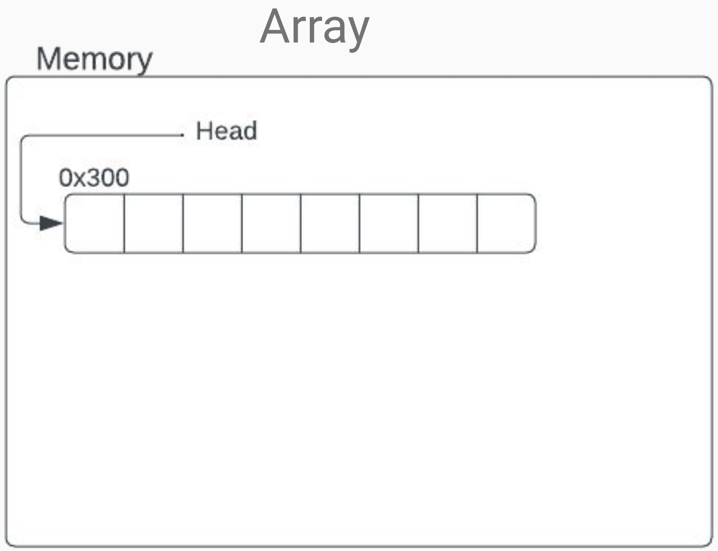
Linked Lists vs. Arrays

Describe an algorithm for each of the following, and determine the complexity:

- Finding an element at a particular index for Arrays and Linked Lists
- Printing out each element of an Array and Linked List
- Changing the value at a particular index for Arrays and Linked Lists
- Changing the value at a particular index in a Linked List if you already have a reference to the node

Linked Lists vs. Arrays





Code Analysis

```
int sumList(List<Integer> list){
  int rslt = 0;
  for(int i = 0; i < list.size(); i++){
    int temp = list.get(i);
    rslt += temp;
  }
  return rslt;
}
```

Code Analysis

```java
int sumLinkedList(SortedList<Integer> list){
  int rslt = 0;
  Optional<LinkedListNode> n = list.headNode;
  while (n.isPresent()){
    int temp = n.get().value;
    rslt += temp;
    n = n.get().next;
  }
  return rslt;
}
```