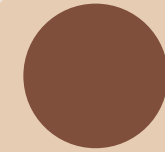
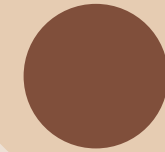


Topic 9.1: Linked Lists

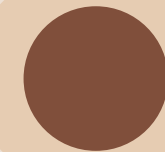
Learning Goals:



Create and manipulate LinkedLists



Compare usage scenarios for Lists & Arrays



Recursively traverse LinkedLists



Draw Memory/Reference Diagrams for LinkedLists



Explain the difference in running times and storage of lists & arrays

OrderedInsert

- Just like the add method from the previous notes but caring about order
- No Array shifting required; just move those references around
- We are **not** keeping them in order for a **binary search**
 - Binary Search is **IMPOSSIBLE** in LinkedLists
 - We do not have random access, we cannot binary search
- There are other data structures that allow VERY quick searching (wait for COMP 2140)

OrderedInsert

- **orderedInsert(Object o) // add o in ascending order**
 - Don't forget to check if:
 - the insert should happen at the front of the list (the smallest value in the list)
 - the insert should happen at the back of the list (the biggest value in the list)
 - the insert should happen in the middle of the list (smaller/bigger than some subset of items in the list)
 - the item is the same value as something else in the list (what should we do?)
- Lots of considerations, let's code this up
 - See OrderedInsertLLExample

Recursion X LinkedList

- How do we recursive print a linked list in order from top to end?
 - Main calls
 - `public void printInOrder()`
 - make a private helper
 - `private void printInOrder(Node curr)`
- How do we recursive print a linked list in **reverse** from end to top?
 - Main calls
 - `public void printReverse()`
 - make a private helper
 - `private void printReverse(Node curr)`

Recursion X LinkedList

- How do we recursive print a linked list in order from top to end?
 - Main calls
 - `public void printInOrder()`
 - make a private helper
 - `private void printInOrder(Node curr)`
- How do we recursive print a linked list in **reverse** from end to top?
 - Main calls
 - `public void printReverse()`
 - make a private helper
 - `private void printReverse(Node curr)`
- **ORDER MATTERS SOMETIMES WE REALLY DO NEED TO DRAW IT OUT!**

Take Home Message

- We have learned how to build our very own data structure: the linked list!
- it requires only 2 simple classes: LinkedList and Node
- The LinkedList class is where we put the methods that can access or modify the LinkedList (add, remove, toString, etc.)
- Whenever you implement a method:
 - Be careful of not breaking the list! (e.g. losing links)
 - Think about the special cases! Your method must handle all possible cases!
- DRAW IT OUT!