Linked Lists

CIT 243

Before We Get Too Far...

Value vs. Reference

- Value - data being stored
- Reference - a “link” to a data storage location

Java is a Pass by Value language:

- When you pass a variable into a method, you pass the value held in that variable, not the variable itself

Let's Make That Slightly More Confusing

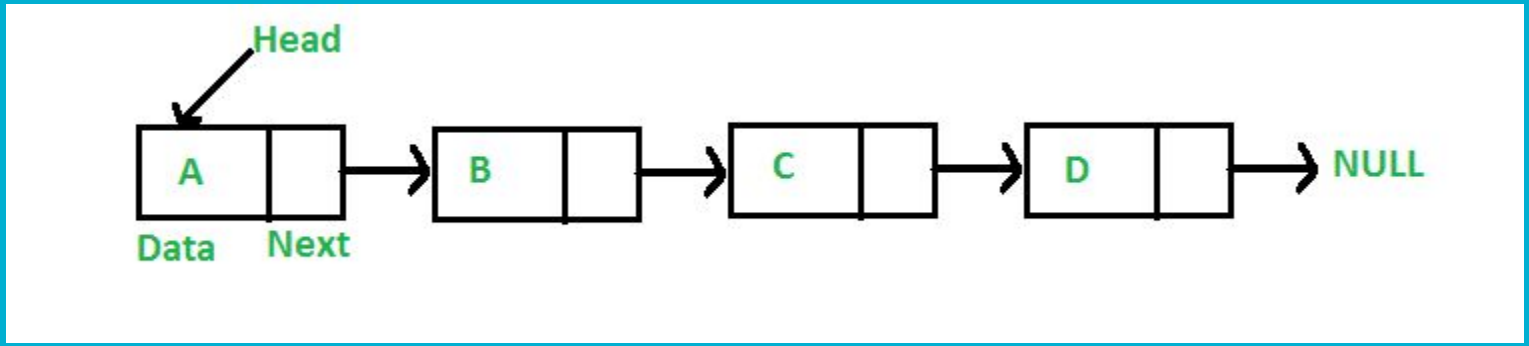
- Dog myDog = new Dog();
  - myDog doesn't contain a Dog. Instead, it contains a "reference" to a Dog
- dog2 = myDog;
  - Both dog2 and myDog point to the same dog
- Java is Pass by Value, but the Value in this variable is a reference

The shorthand is: If you pass an object into a method, you can mutate the object inside of that method

Linked Lists

Linked List

- Made up of traversable “Nodes”
- Each Node contains some data and a link to the next element
- Has a “Head” to represent the start of the list



Advantages of Linked Lists

- Easy to Modify
  - Insertion and Deletion are easy
- Only takes up as much space as it needs

Disadvantages of Linked Lists

- No “random” access
- Reverse traversing is difficult

Java's Built-In Linked List

- LinkedList<Something> myList = new LinkedList<>();
- Probably has everything you need

https://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html

Let’s Build Our Own Anyways

- We need a “Node”
- We need a reference to the Head of our list
- We should be able to:
  - Add stuff to the list
  - Remove stuff from the list
  - Get a value from the list
  - Get the size of the list

Let’s Improve It

- How could we make it easier to add stuff to the end of the list?
- Let’s add a prepend method to add stuff to the beginning of the list
- Let’s create an add method that adds data to a specified index

Let’s Make it Doubly Linked

- Each Node has a reference to both the next and previous element