Aaron Jones
Lab 14 – APE Linked List
6/3/2014

Part 1

```
public class LinkedListA extends LinkedList
{
  // Complete the 'add by index' method below.
  // The inherited class 'LinkedList' uses a dummy head node.
  // Note if you use a 'find' method you must write it.
  // Be sure to test for an invalid index and throw an
  // IndexOutOfBoundsException if found...
  // Don't forget to update size!

  public void add(int index, Comparable dataToAdd)
  {
    if(index >= 0 && index <= size)
    {
      Node newNode = new Node(dataToAdd);
      if(index == 0)
      {
       newNode.next = head.next;
        head.next = newNode;

      }
      else
      {
        Node prev = find(index - 1);
        newNode.next = prev.next;
        prev.next = newNode;
      }
     size++;

    }
    else
    {
      throw new IndexOutOfBoundsException("Index out of Bounds" + index);

    }
  }// end add

  //Add find method
  private Node find(int index)
  {
    Node curr = head.next;
    for(int skip = 0; skip < index; skip++)
    {
      curr = curr.next;
```

```
        }
        return curr;
      }


   }// end class
```

Part 2:

```
public class LinkedListB extends LinkedList
  {
    //  Complete the 'removeAll' method below such that
    //  all occurances of a data value are removed from the list for
    //  each element of the passed-in array.
    //  For example, if the list contained values 3, 2, 2, 3, 4, 2, 5
    //  and the passed-in collection of items to remove is 2, 4 the
    //  resultant list would be reduced to 3, 3, 5.
    //  Return 'true' if the list was altered - return 'false' if the list
    //  is unchanged.
    //  If the array is null, throw a NullPointerException.
    //  Note the inherited class 'LinkedList' uses a dummy head node.
    public boolean removeAll(Comparable [] array)
    {
    //  Your code goes here...
       Comparable [] array2;
       array2 = array;
       if(array.length >= 0)
       {
         if(array.length < array2.length)
         {
           return true;
         }
         else
         {
           return false;
         }
       }
       else
       {
         throw new NullPointerException("List is empty");
       }
    }// end removeAll
```

Aaron Jones
Lab 14 – APE Linked List
6/3/2014

```
}// end class
```