We search for values in various lists.

1. Project proposals! Comments back on Friday.

2. Questions?

3. Recall that we test to see if two values of type \texttt{s} are equal with \texttt{==}. Here’s the \texttt{==} method for our Ratio class:

   ```python
   def __eq__(self,other):
       return (self.num == other.num) and (self.den == other.den)
   ```

4. Similarly, we implement \texttt{a < b} with the \texttt{<} method. Here’s ratio’s \texttt{<} method:

   ```python
   def __lt__(self,other):
       return self.num*other.den < self.den*other.num
   ```

5. Reviewing: polished version of wrapper class \texttt{LinkedList}.
   
   (a) General inspection.
   
   (b) Building the iterator over linked lists (\texttt{iter}).
   
   (c) Unit testing.

6. Focus: Locating values in list classes.
   
   (a) We test for a value \texttt{v} in a list \texttt{l} with \texttt{v in l}. The \texttt{in} operator is a call to \texttt{\_\_contains\_}. How long does it take to perform this operation on a general list?
   
   (b) In our \texttt{ll} linked list class, which is a recursive structure, we can write a recursive \texttt{\_\_contains\_} method.
   
   (c) What is possible if we know that the list keeps its items in order with \texttt{\_\_lt\_}? For Python’s \texttt{list} class: binary search. For linked lists: harder.


8. Keeping lists in order: sorting.
   
   (a) A naive technique: bubble sort.
   
   (b) A technique for Halloween: Selection sort. Pull out the largest value.
   
   (c) A technique for card players: Insertion sort. Add values to a sorted set.
   
   (d) Quicksort.
   
   (e) Merge sort.