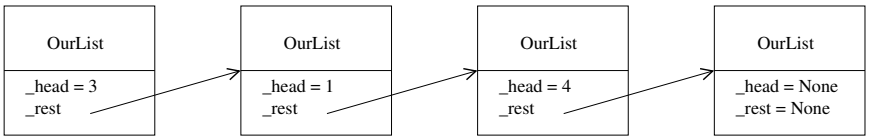
Working with the OurList class

As our next example, we develop our own list implementation from scratch. Obviously, Python already includes the built-in list class, so there is no need for us to do this. At the same time, it is empowering to realize that we can develop such an interesting class ourselves, had it not already been provided. The idea is to view a list as a recursive structure similar to our Bullseye class. Each list will be represented with two attributes: _head, which represents the first element of the list, and _rest, which is itself a list of all remaining items. So a list of three elements is represented as a first element and a remaining list of two items. That list of two items is represented as a single element followed by a list with one item. Even the list of length one is viewed as a single element followed by a list of zero remaining elements. Below is the diagram representing the list [3, 1, 4]:



Our goal is to understand how several methods of the OurList class should work. You will be playing the role of an OurList instance. Your job is to perform the role of a member function. To do this, run the simulator program. You will be asked what method you want to experiment with. For each experiment, you will be given a member function that you need to simulate, including any parameters passed to the function. Inside the simulator you will be able to call member functions for _rest, set the value of member variables, return from the function or raise an exception.

Just like we saw in the Bullseye class, each method will have to deal with multiple cases. In particular, there might be cases when the OurList instance is empty and when it is not. For each of the methods we are examining, you will need to determine what cases need to be dealt with and what to do in each of the cases. You will be explaining what the cases are and what to do in each of them.

For each function, run experiments with the simulator. It will tell you if you are giving the correct results. Continue running experiments with each function until you get 5 to 10 correct answers. The experiments will run through all of the possible cases in a random order.

For each method answer the following questions:

- What cases does the method need to deal with?
- For each case...
  - What should the method do? (For example, does it call a member function of _rest, does it check the value of _head, ...)
  - What, if anything, is returned by the method.

1. Consider how __len__() should work. This is the method that is called when you want to to know the length of a list. It returns the length of the current OurList instance.
2. The __contains__ method is called whenever you want to see if a particular item is in the list. (Note that the Python syntax item in numbers is equivalent to the call numbers.__contains__(item).) The item you are searching for is passed as a parameter. The method returns True if the item is in the list and False otherwise.
3. The __getitem__ method returns the item stored at a particular index of the list and the parameter is this index. It should either return the value at the location of the list or throw an exception.
4. Finally, we need to consider how append works. It is given a item as a parameter and that item should be appended to the end of the list. The function returns nothing, but does mutate the list.