Singly-Linked List

→ collection of nodes that form a linear sequence
→ each node stores a reference to an obj. that is part of the sequence, as well as a reference to next node of list

→ list maintains member named 'head' at front of list
→ some lists also maintain a 'tail' for last member
→ a list is out of elements when the pointer to the next element returns None/null value

Inserting an Element at Head

→ linked lists have no fixed size; uses space proportionally to curr. # of elements
→ to insert @ head:
- create a new node
- set element to the new element
- set 'next' link to the previous head
- then set the lists 'head' to be our new node

PSUEDO CODE

```
add_first(L,e):
  newest = Node(e)          // create new node w/ value
  newest.next = L.head      // set new node reference to old head
  L.head = newest           // set var head to ref. new node
  L.size += 1               // increment node ct
```

Insert Element at Tail

- create new node
- assign 'next' pointer to be None
- set 'next' ref of old tail to new node
- update tail ref to new node

```
add_last(L,e):
  newest = Node(e)
  newest.next = None
  L.tail.next = newest
  L.tail = newest
  L.size += 1
```