CSC148  Intro. to Computer Science

Lecture 5: Linked Lists


Office Hours: W 16:00–17:45 BA4222

ahchinaei@cs.toronto.edu
http://www.cs.toronto.edu/~ahchinaei/

Course webpage:
http://www.cdf.toronto.edu/~csc148h/winter
Review

❖ So far
  ▪ Class design and implementation
  ▪ Composition and inheritance
  ▪ Inheriting, extending, and overriding
  ▪ Specific examples:
    • Shape: square, right angled triangle
    • Container: stack, sack
  ▪ Intro to linked lists

❖ Today
  ▪ More on inked lists
  ▪ Wrappers and helpers
Regular lists vs. linked lists

- Regular Python lists:
  - pro(s): *it can efficiently be accessed*
  - con(s): they allocate large blocks of contiguous memory, which becomes increasingly difficult as memory is in use.

- Linked list nodes reserve just enough memory for the object value they want to refer to, a reference to it, and a reference to the next node in the list
  - Pro(s): *it can efficiently grow and shrink, as needed*
For now, we implement a linked list as objects (nodes) with a value and a reference to other similar objects.

12 → 99 → 37 → □
Helper: Node

Examples:

```
<table>
<thead>
<tr>
<th>value</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td></td>
</tr>
</tbody>
</table>
```

```
99 →
```
class LinkedListNode:
    
    Node to be used in linked list

    === Attributes ===
    @param LinkedListNode next_: successor to this LinkedListNode
    @param object value: data this LinkedListNode represents

    def __init__(self, value, next_=None):
        
        Create LinkedListNode self with data value and successor next_.

        @param LinkedListNode self: this LinkedListNode
        @param object value: data of this linked list node
        @param LinkedListNode|None next_: successor to this LinkedListNode.
        
        @rtype: None

        self.value = value
        self.next_ = next_
What other methods does class node, ie. LinkedListNode need?
Wrapper: LinkedList

LinkedListNode back
LinkedListNode front
int size

value | Next_ | ....... | value | Next_

Example:

| 12 | 99 | 37 |

Size: 3
Wrapper: LinkedList class

class LinkedList:
    """
    Collection of LinkedListNodes
    """

    === Attributes ==
    @param: LinkedListNode front: first node of this LinkedList
    @param LinkedListNode back: last node of this LinkedList
    @param int size: number of nodes in this LinkedList
        a non-negative integer
    """

def __init__(self):
    """
    Create an empty linked list.
    """

    @param LinkedList self: this LinkedList
    @rtype: None
    """

    self.front, self.back = None, None
    self.size = 0
Wrapper: LinkedList class

What other methods does class LinkedList need?