# **Objective:** To gain experience implementing linked data structures by implementing a cursor-based list using doubly-linked nodes.

**To start the homework:** Download and extract the file hw3.zip from eLearning at Course Content | Unit 1 | Homework #3. The hw3.zip file contains:

1. the Node class (in the node.py module) and the Node2Way class (in the node2way.py module)
2. the skeleton CursorBasedList class (in the cursor\_based\_list.py module) which you will complete
3. the cursorBasedListTester.py file that you can use to interactively test your CursorBasedList class.

Recall that in a **cursor-based list** a *cursor* (indicating the *current item*) can be moved around the list with the cursor being used to identify the region of the list to be manipulated. We will insert and removing items relative to the current item. A *current item* must always be defined as long as the list is not empty.

| **Cursor-based operations** | **Description of operation** |
| --- | --- |
| L.getCurrent() | Precondition: the list is not empty. Returns the current item without removing it or changing the current position. |
| L.hasNext() | Precondition: the list is not empty. Returns True if the current item has a next item; otherwise return False. |
| L.next() | Precondition: hasNext returns True. Postcondition: The current item has moved right (i.e., toward the tail) one item |
| L.hasPrevious() | Precondition: the list is not empty. Returns True if the current item has a previous item; otherwise return False. |
| L.previous() | Precondition: hasPrevious returns True. Postcondition: The current item has moved left (i.e., toward the head) one item |
| L.first() | Precondition: the list is not empty. Makes the first item the current item. |
| L.last() | Precondition: the list is not empty. Makes the last item the current item. |
| L.insertAfter(item) | Inserts item after the current item, or as the only item if the list is empty. The new item is the current item. |
| L.insertBefore(item) | Inserts item before the current item, or as the only item if the list is empty. The new item is the current item. |
| L.replace(newValue) | Precondition: the list is not empty. Replaces the current item by the newValue. |
| L.remove() | Precondition: the list is not empty. Removes and returns the current item. Making the next item the current item if one exists; otherwise the tail item in the list is the current item unless the list in now empty. |
| len(L)  Calls \_\_len\_\_ method | Precondition: None. Returns the number of items in the list. |
| L.isEmpty() | Precondition: None. Returns True if the number of items in the list is 0; otherwise return False. |

The cursor\_based\_list.py file contains a skeleton CursorBasedList class. **You MUST uses a doubly-linked list implementation with a *header* node and a *trailer* node. All “real” list items will be inserted between the header and trailer nodes to reduce the number of “special cases”** (e.g., inserting first item in an empty list, deleting the last item from the list, etc.). An empty list looks like:

Empty CursorBasedList with header and trailer nodes.

The skeleton CursorBasedList class’s \_\_init\_\_ method already creates the above empty list.

Use the provided cursorBasedListTter.py program to test your list methods. NOTE: You don’t need to do all the list methods before testng. I suggest doing \_\_len\_\_, isEmpty, getCurrent, and insertAfter methods first.

Starting with a newly constructed empty CursorBasedList and doing an insertAfter of item ‘a’, then the list would look like:

CursorBasedList after insertAfter method called with the item 'a' .**On eLearning (Course Content | Unit #1 | Homework #3 subfolder), submit a single .zip file, hw3.zip containing the following:**

1. the Node class (in the node.py module) and the Node2Way class (in the node2way.py module)
2. the completed CursorBasedList class (in the cursor\_based\_list.py module)
3. the cursorBasedListTester.py file

Note: No design document needed for this homework.

(If you miss the deadline, you can still submit it without a late penalty. However, this assignment is good preparation for Test 1. Plus, there will be a homework 4, etc. and you don’t want to get too far behind!)