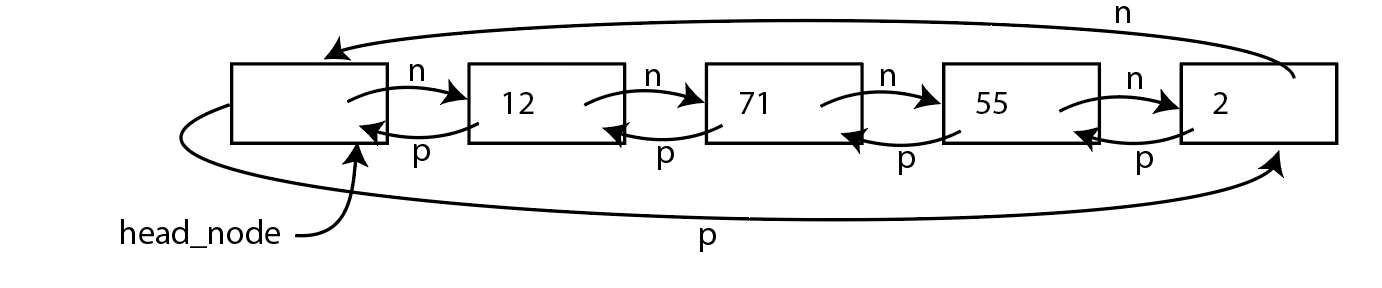
Lab 7a – Circular Linked List

In this exercise you will create a circular linked list with pointers in both directions and a “fake node” at the head of the list.

The image below shows a circular linked list with 4 values... 12, 71, 55, and 2.



The empty list should still contain the “head\_node” where the next and previous pointers point to the head\_node.

The ADT will be called CircularLinkedList.

The following functions are required for your class:

* insert\_front(x): This will put a new item into the circular linked list at the front. In the example shown the new value would appear between the head\_node and the 12.
* insert\_back(x): This will put a new item into the list at the back. In the example show, the new value would appear between the head\_node and the 2.
* debug\_forward(): Print all the values in the list in forward order.
* debug\_reverse(): Print all the values in the list in reverse order.
* list\_forward(n): Generate a Python list with n values from the circular list in forward order. If n is larger than the size of the list, we will see values repeating. In the example above if we call list\_forward(6) we should get back a list with [12, 71, 55, 2, 12, 71].
* delete\_front(): Remove the first item in the list
* delete\_back(): Remove the last item in the list.

This time the implementation is important as it must be a linked set of nodes.