1) Build a singly linked list that supports the following operations:

```
prepend(value)          -> Add a node in the beginning
append(value)           -> Add a node in the end
pop()                   -> Remove a node from the end
popFirst()              -> Remove a node from the beginning
head()                  -> Return the first node
tail()                  -> Return the last node
remove(Node)            -> Remove Node from the list
```

2) Reverse a singly linked list in place
3) Rotate a singly linked list by k places

```
Input: linked list = [1,2,3,4,5], k = 2
Output: [4,5,1,2,3]
```

4) Split a linked list in ***k*** consecutive linked list parts. The length of each part should be equal if possible. If not, they shouldn't differ more than one. Some parts may be empty.

```
Input: linked list = [1,2,3], k = 5

Output: [[1],[2],[3],[],[]]

Input: linked list = [1,2,3,4,5,6,7], k = 4

Output: [[1, 2],[3, 4],[5, 6],[7]]
```

5) Round Robin
   L is a linked list
   1. process p = L.popFirst( )
   2. Give a time slice to process p
   3. L.addLast(p)
   Are we really going to remove a node from the beginning and put it in the end 🙂?

   Create a circularly linked list of ***n*** processes. Simulate giving time to the CPU for each process. Do some work randomly. Update the progress bar.

6) Doubly Linked List