Homework #2

P1: What will the computer prints if the following program is executed correctly?

```
l = new Stack();
l.push("3")
l.push("5")
l.push("10")
l.pop()
print(l.pop())
l.push("1")
l.push("hello ")
l.push("-1")
while (not l.isEmpty()) {
    print(l.pop())
}
```

P2: Implement a stack using a singly linked list L. The operations, `STACK-EMPTY`, `PUSH` and `POP` shall takes $O(1)$ time. Pseudocode is enough for this problem. Assume the L and nodes have the interface we have in the class.

P3: Consider a new abstract data type, deque. A deque (pronounced "deck") is an abstract data type that generalizes a queue, for which elements can be added to or removed from either the front (head) or the back (tail). This data structure allows for more versatile operations than standard queues or stacks.

A deque usually needs to support the following:

1. pushFront(data): Adds an element to the front of the deque. Input: data (the value to be added). Operation: Inserts the new element at the beginning of the deque.
2. pushBack(data): Adds an element to the back of the deque. Input: data (the value to be added). Operation: Inserts the new element at the end of the deque.
3. popFront(): Removes an element from the front of the deque. Output: The value of the removed element. Operation: Deletes the first element of the deque.

4. popBack(): Removes an element from the back of the deque. Output: The value of the removed element. Operation: Deletes the last element of the deque.
5. peekFront(): Returns the value of the front element without removing it. Output: The value of the front element. Operation: Retrieves the value of the first element

Now try to implement a deque with a doubly linked list.

P4: Give a $\Theta(n)$-time nonrecursive procedure that reverses a singly lined list of n elements. The procedure should use no more than constant storage beyond needed for the list itself.