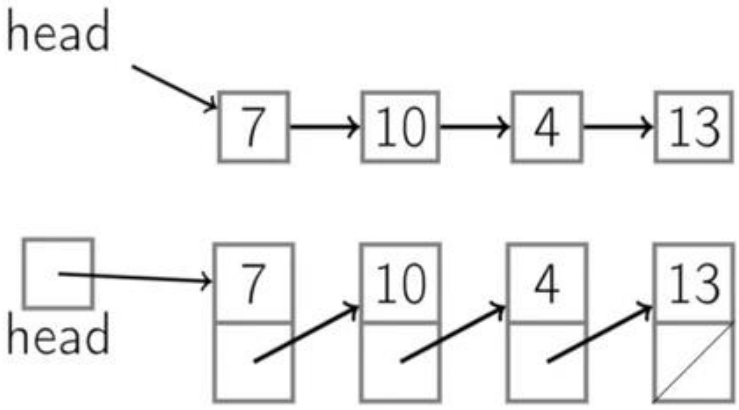
Singly-Linked Lists

Singly-Linked List



Node contains:

- key
- next pointer

List API

`PushFront(Key)`	add to front
`Key TopFront()`	return front item
`PopFront()`	remove front item
`PushBack(Key)`	add to back
`Key TopBack()`	return back item
`PopBack()`	remove back item
`Boolean Find(Key)`	is key in list?
`Erase(Key)`	remove key from list
`Boolean Empty()`	empty list?
`AddBefore(Node, Key)`	adds key before noc

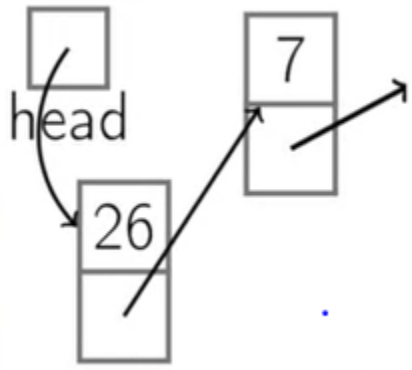
Time Complexity Analysis of Singly Linked List

- PushFront O(1)
- PopFront O(1)
- PushBack (no tail) O(n)
- PopBack (no tail) O(n)
- PushBack (with tail) O(1)
- PopBack (with tail) O(n)

Singly-linked List

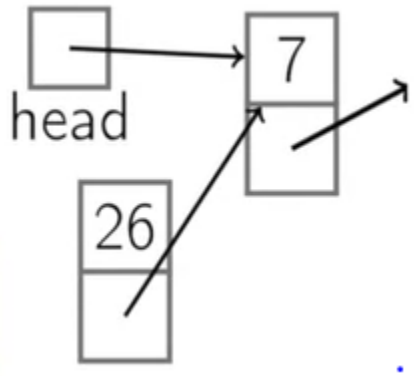
```
PushFront(key)

node ←new node
node.key ← key
node.next ← head
head ← node
if tail = nil:
    tail ← head
```



Singly-linked List

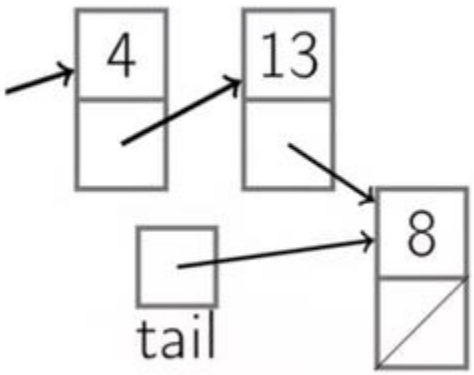
```
PopFront()
if head = nil:
  ERROR: empty list
head ← head.next
if head = nil:
  tail ← nil
```



Singly-linked List

PushBack(*key*)

```
node ←new node
node.key ← key
node.next =nil
if tail = nil:
    head ← tail ← node
else:
    tail.next ← node
    tail ← node
```



Singly-linked List

PopBack()

```
if head = nil:  ERROR: empty list
if head = tail:
   head ← tail ←nil
else:
   p ← head
   while p.next.next ≠ nil:
      p ← p.next
   p.next ← nil; tail ← p
```