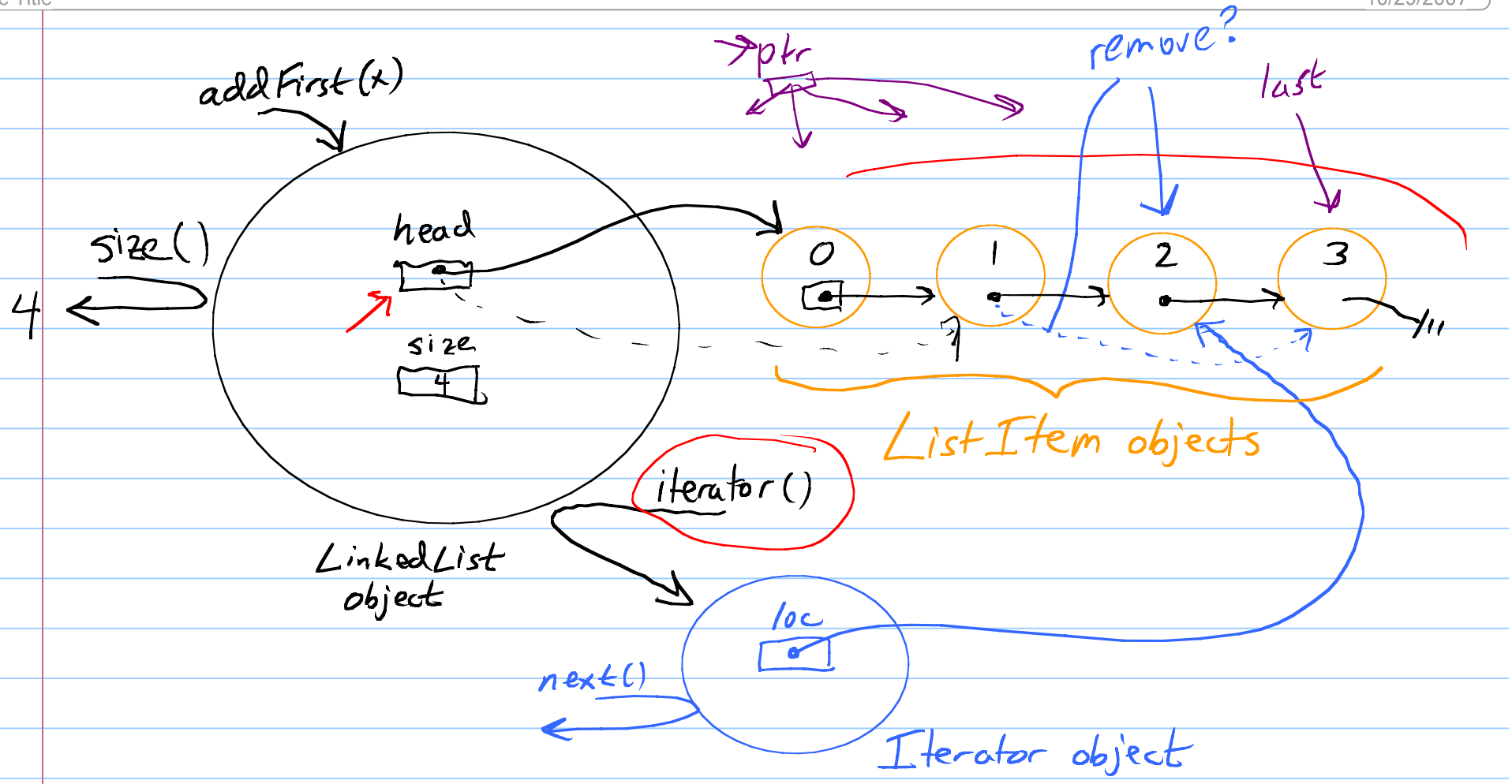
Implementing a List ADT using Encapsulated List Structure

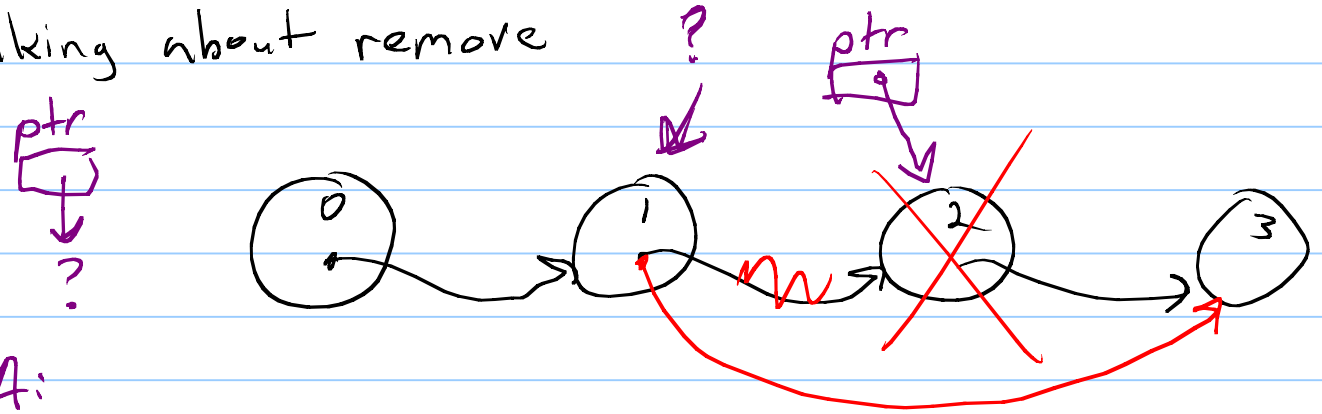
Note Title 10/29/2007



List Methods to be Supported

	Method Name	Parameters	Effects	Return
	constructor	—	init. an empty list	(the empty list)
instance variable	size	—	—	int – # of elements
head ref.	addFirst	x	put x at front of list	—
tail ref.	addLast	x	put x at back of list	—
search	indexOf	x	—	int – x's position (first occurrence)
search	remove	x	take x out of the list	boolean – true if successful
class	iterator	—	—	an Iterator positioned before the first item

Thinking about remove

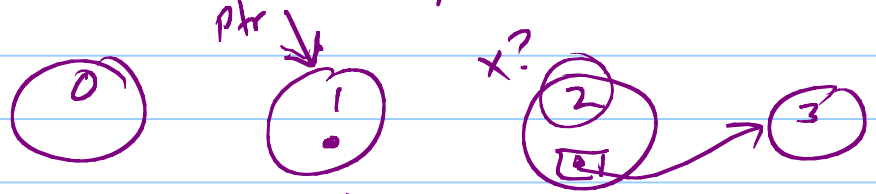


Plan A:

1. find x
2. search again to look for item before x

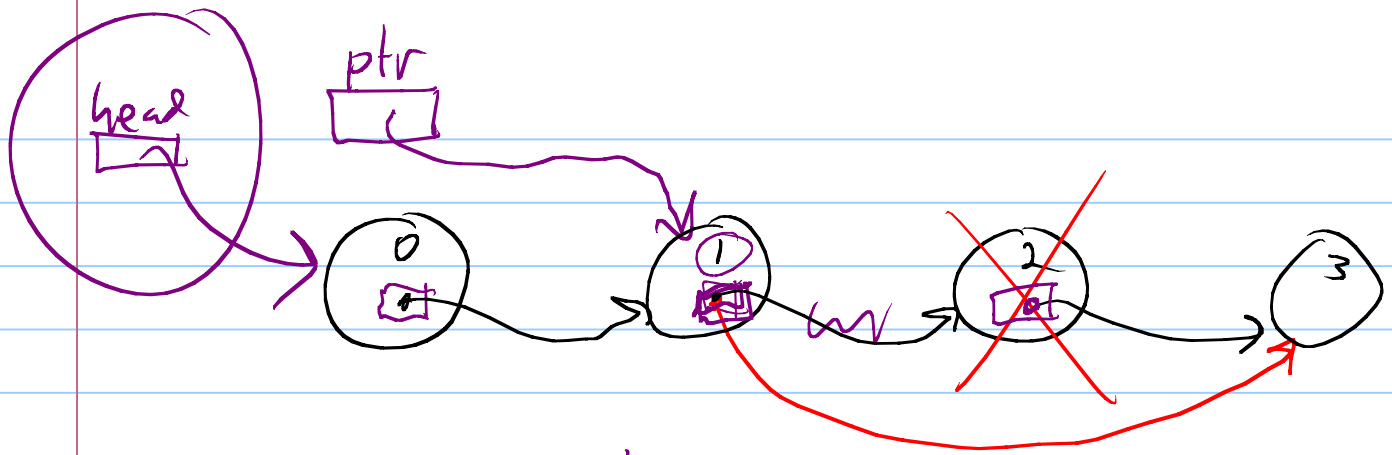
Plan B:

1. As we search, keep the pointer one "behind"

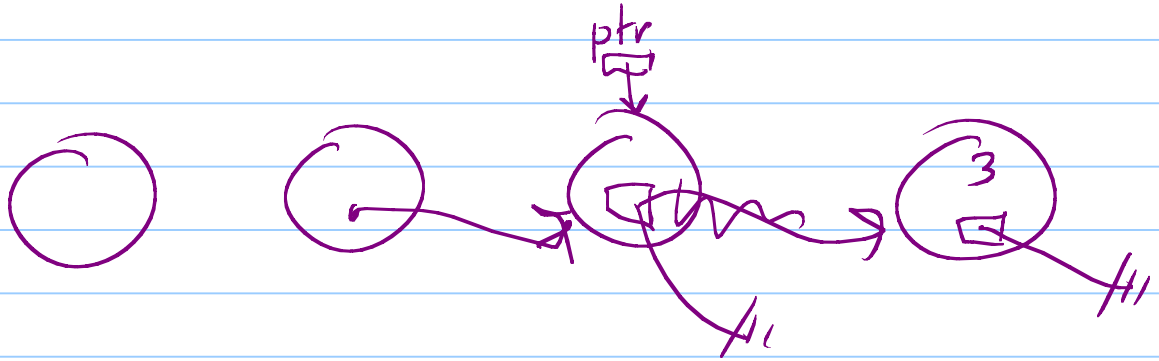


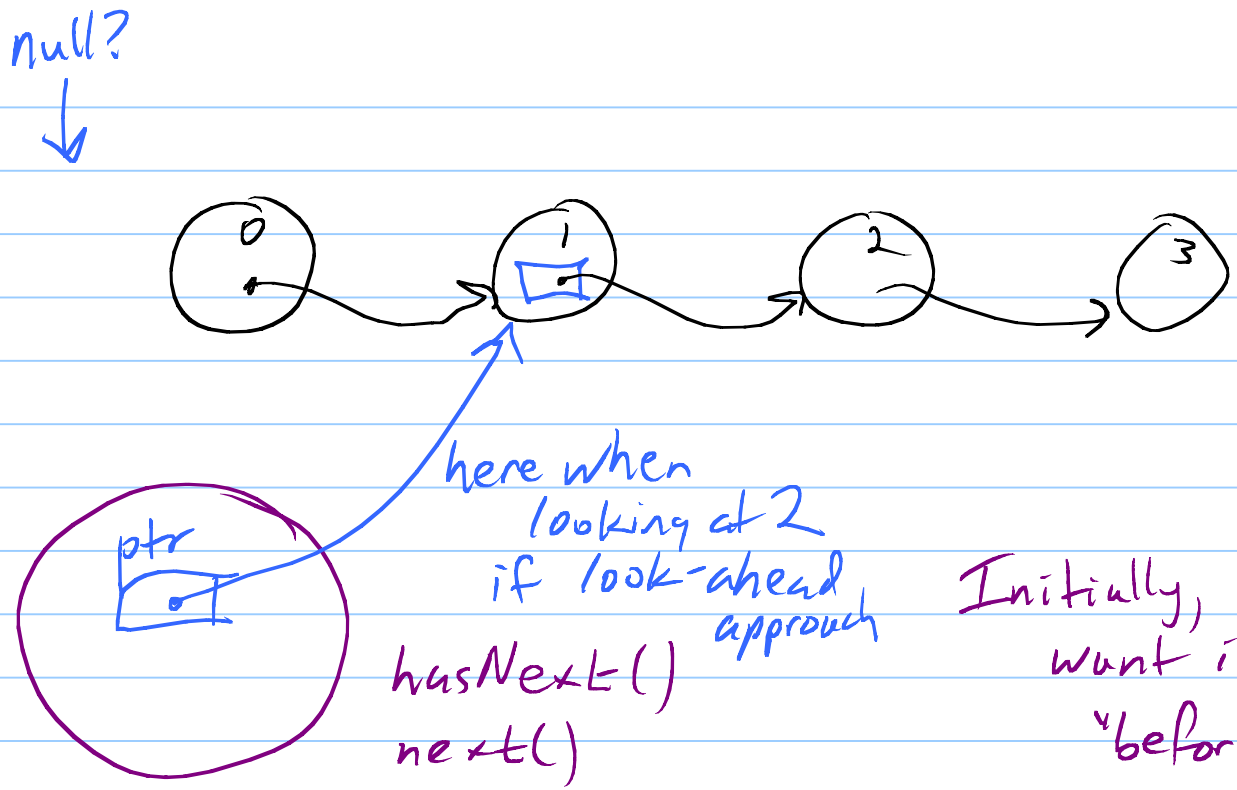
(ptr.next.number == x)

Problem:
What about first item in list?



ptr.next = ptr.next.next



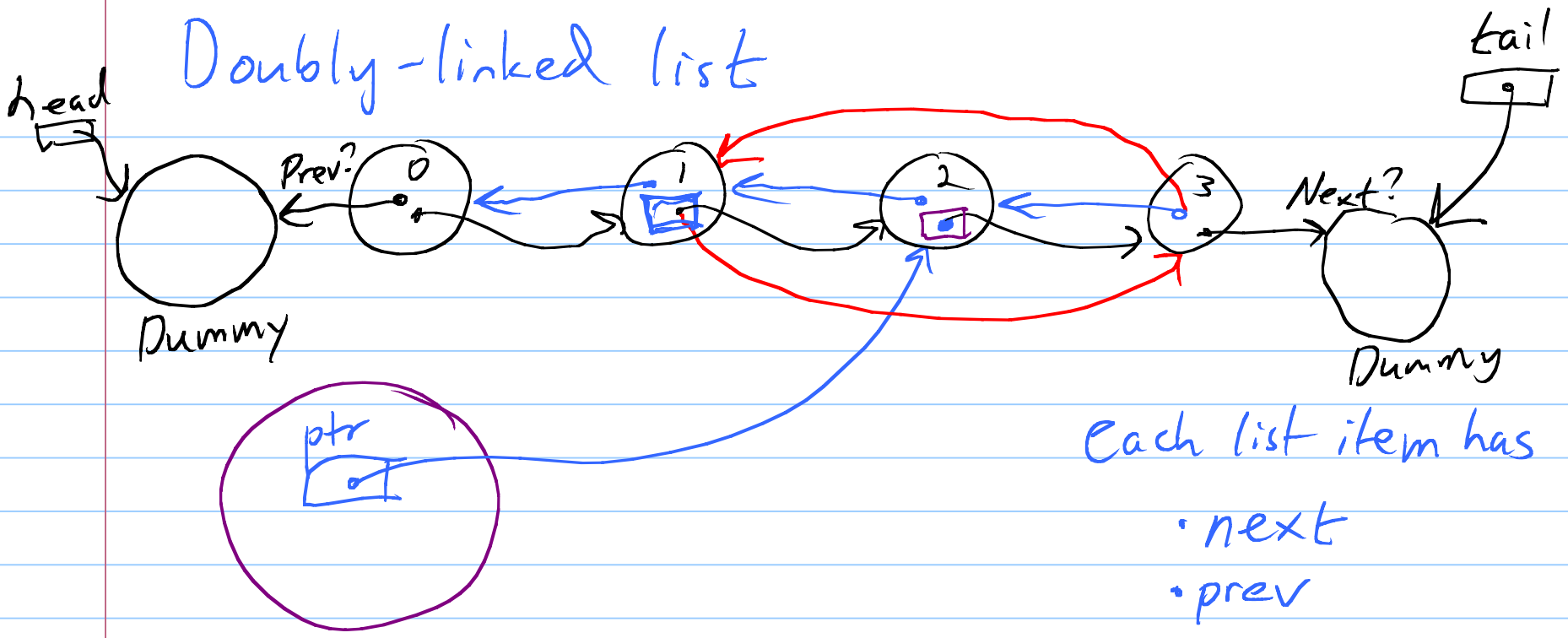


hasNext()
next()
remove()

Initially, want iterator just "before" the start of the list

Idea A: Start at null
advance by ptr = ptr.next (if ptr == null, let ptr = head)

Doubly-linked list



Each list item has

- next
- prev

To remove item at ptr:

```
ptr.prev.next = ptr.next;
ptr.next.prev = ptr.prev;
```

In general, invariant:

```
item.next.prev == item
item.prev.next == item
```