Lecture 14: Finish stacks, start queues

Admin

Skim Doubly Linked Lists, section 10.2

This week: Stacks and Queues (not priority queues, though.)

Reading for Friday/Monday lecture on binary tree nodes

B.5.2, Rooted trees

B.5.3, Binary trees

I have to move my Thursday office hours 2-3 to Friday 2-3 this week (and maybe after this week)

LAST grading slots are in process of posting; we'll alert you

This weekend: Homework 2, Big Integer with doubly linked list

How stacks?

Unsorted dynamic array

push: add

pop: remove last element

top: return last element

empty: count

Notice how we reuse pre-existing code here…

How stacks?

Unsorted dynamic array

push: add

pop: remove last element

top: return last element

empty: count

Linked list:

push: add to head

pop: remove from head

top: return head element

empty: head_ptr

Queues

Opposite rule from stacks. First come, first served.

Linked list:

push: add to tail

pop: remove from head

top: return head element

empty: head_ptr

Queues

Opposite rule from stacks. First come, first served.

Unsorted dynamic array

- push: add to end
- pop: remove ‘first’ element
- front: return ‘first’ element
- empty: count

Keep track of first and last elements

Queues

Unsorted dynamic array

push: add to end

pop: remove ‘first’ element

front: return ‘first’ element

empty: count

push(1)	1
push(2)	1 2
pop()	_ 2
push(3)	_ 2 3
push(4)	_ 2 3 4
pop()	_ _ 3 4

Queues

Circularize the array. Suppose its size is 4.

push(1)	1	
push(2)	1 2	
pop()	_ 2	
push(3)	_ 2 3	
push(4)	_ 2 3 4	
pop()	_ _ 3 4	
push(5)	5 _ 3 4	wrap around…
push(6)	5 6 3 4	
	B F	

What condition could we use to test for this?