CSE 30331/34331: Data Structures

(Almost) Everything You Need to Know for the Final

Abstract data types

	insert	delete
stack	back	back
queue	back	front
deque	front/back	front/back
priority queue	anywhere	min
set	anywhere	by value

Sequence data structures

	find by position	find min/max or by value	insert at position	delete at position	delete min/max or by value	concatenate
array	O(1)	O(n)	back: O(1) else: O(n)	back: O(1) else: O(n)	O(n)	O(n)
circular array	O(1)	O(n)	front/back: O(1) else: O(n)	front/back: O(1) else: O(n)	O(n)	O(n)
linked list	front: O(1) else: O(n)	O(n)	front: O(1) else: O(n)	front: O(1) else: O(n)	O(n)	O(n)
linked list with tail pointer	front/back: O(1) else: O(n)	O(n)	front/back: O(1) else: O(n)	front: O(1) else: O(n)	O(n)	O(1)
doubly linked list	front/back: O(1) else: O(n)	O(n)	front/back: O(1) else: O(n)	front/back: O(1) else: O(n)	O(n)	O(1)

Set data structures

	find min/max	find by value	insert	delete min/max	delete by value	create from list	union
sorted array	O(1)	O(log n)	O(n)	max: O(1) else: O(n)	O(n)	O(n log n)	O(n)
sorted linked list	min: O(1) else: O(n)	O(n)	O(n)	min: O(1) else: O(n)	O(n)	O(n log n)	O(n)
binary heap	min: O(1) else: O(n)	O(n)	O(log n)	min: O(log n)	O(n)	O(n)	O(n)
B-tree **red-black tree**	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(n log n)	O(n)
hash table	O(n)	O(1)	O(1)	O(n)	O(1)	O(n)	O(n)

Sorting algorithms

intermediate data structure	one and rest	half and half
array	selection sort: O(n^2)	quicksort: O(n log n)
sorted array	insertion sort: O(n^2)	merge sort: O(n log n)
binary heap	heap sort: O(n log n)	–
order-preserving hash table	bucket sort	–

Graph search algorithms

agenda	algorithm
stack	depth-first search
queue	breath-first search
priority queue	Dijkstra's algorithm