Abstract Data Types and Data Structures

Used in Algorithm Design

ADT	Operation	Op. type	Best Data Structure	Big O	Alt data structure	Big O
Stack (LIFO Queue)	Top	R	Dynamic Array	O(1)	Linked List	O(1)
	Push	W	Dynamic Array	O(1)	Linked List	O(1)
	Pop	W	Dynamic Array	O(1)	Linked List	O(1)
Queue (FIFO Queue)	Top	R	Linked List with tail	O(1)	Circular Array with 2 pointers	O(1)
	Enqueue	W	Linked List with tail	O(1)	Circular Array with 2 pointers	O(1)
	Dequeue	W	Linked List with tail	O(1)	Circular Array with 2 pointers	O(1)
Priority Queue	Top	R	Binary Heap	O(1)	Balanced Binary Search Tree	O(log n)
	Enqueue	W	Binary Heap	O(log n)	Balanced Binary Search Tree	O(log n)
	Dequeue	W	Binary Heap	O(log n)	Balanced Binary Search Tree	O(log n)
Set	HasKey	R	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)
	Insert	W	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)
	Delete	W	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)

Map	Get	R	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)
	Set	W	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)
	Delete	W	Hash Table	O(n), Expected O(1)	Balanced Binary Search Tree	Guaranteed O(log n)
Local Range	Range	R	Balanced Binary Search Tree (for example, B+ tree)	O(log n) + output size		
	Nearest Neighbors	R	Balanced Binary Search Tree (for example, B+ tree)	O(log n) + k, where k is the number of neighbors		
	Predecess or	R	Balanced Binary Search Tree	O(log n)		
	Successor	R	Balanced Binary Search Tree	O(log n)		
	Insert	W	Balanced Binary Search Tree	O(log n)		
	Delete	W	Balanced Binary Search Tree	O(log n)		