Arrays

(Space-Time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Accessing an element	O(1)	O(1)	O(1)
Updating an element	O(1)	O(1)	O(1)
Deleting an element	O(n)	O(n)	O(1)
Inserting an element	O(n)	O(n)	O(1)
Searching for an element	O(n)	O(n)	O(1)

Algorithm complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Quicksort	$O(n^2)$	O(n log(n))	O(n log(n))	O(log(n))
Mergesort	O(n log(n))	O(n log(n))	O(n log(n))	O(n)
Heapsort	O(n log(n))	O(n log(n))	O(n log(n))	O(1)
Bubble Sort	$O(n^2)$	$O(n^2)$	O(n)	O(1)
Insertion Sort	$O(n^2)$	$O(n^2)$	O(n)	O(1)
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	O(1)
Binary Search	O(log(n))	O(log(n))	O(1)	O(1)
Linear Search	O(n)	O(n)	O(1)	O(1)

Strings

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Accessing	O(1)	O(1)	O(1)
Deleting	O(n)	O(n)	O(1)
Inserting	O(n)	O(n)	O(1)
Searching (n = string length m = pattern length)	O(n * m)	O(n)	O(1)
Slicing (n = string length)	O(n)	O(n)	O(n)
Concatenating (n, m = string lengths)	O(n + m)	O(n + m)	O(n)
Comparison (n = shorter string length)	O(n)	O(n)	O(n)
Inserting (Trie) (m = key length)	O(m)	O(m)	O(1)
Searching (Trie) (m = key length)	O(m)	O(m)	O(1)

Algorithm complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Radix sort (m = longest string length)	O(n * m)	O(n * m)	O(n * m)	O(n + m)
Naive string search (m = size of pattern)	O(m*(n-m+1))	O(n * m)	O(n)	O(1)
Knuth-Morris-Pratt search	O(m + n)	O(n)	O(n)	O(m)
Boyer-Moore string search	O(n * m)	O(n)	O(n/m)	O(m)
Rubin-Karp Algorithm	O(m*(n-m+1))	O(n + m)	O(m)	O(m)

Stacks & Queues

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Delete (Stack)	O(1)	O(1)	O(1)
Insert (Stack)	O(1)	O(1)	O(1)
Search (Stack)	O(n)	O(n)	O(1)
Peek/Top (Stack)	O(1)	O(1)	O(1)
Delete (Queue)	O(1)	O(1)	O(1)
Insert (Queue)	O(1)	O(1)	O(1)
Search (Queue)	O(n)	O(n)	O(1)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Linear Search	O(n)	O(n)	O(1)	O(1)

Stack: **LIFO** (Last In First Out)

Queue: **FIFO** (First In First Out)

Linked Lists

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Accessing	O(n)	O(n)	O(1)
Deleting (after search)	O(1)	O(1)	O(1)
Inserting (after search)	O(1)	O(1)	O(1)
Searching	O(n)	O(n)	O(1)
Traversing	O(n)	O(n)	O(n)
Access (Skip List)	O(n)	O(log n)	O(log n)
Delete (Skip List)	O(n)	O(log n)	O(log n)
Insert (Skip List)	O(n)	O(log n)	O(log n)
Search (Skip List)	O(n)	O(log n)	O(log n)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Mergesort	O(n log n)	O(n log n)	O(n log n)	O(n)
Bubble Sort	$O(n^2)$	$O(n^2)$	O(n)	O(1)
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	O(1)
Insertion Sort	$O(n^2)$	$O(n^2)$	O(n)	O(1)
Linear Search	O(n)	O(n)	O(1)	O(1)

Trees

(Space-time complexity)

Time complexity:

		Worst Case Scenario	Average Case Scenario	Best Case Scenario
Binary Search Tree, Cartesian Tree, KD Tree	Delete	O(n)	O(log n)	O(log n)
	Insert	O(n)	O(log n)	O(log n)
	Search	O(n)	O(log n)	O(log n)
B-Tree, Red-Black Tree, Splay Tree, AVL Tree	Delete	O(log n)	O(log n)	O(log n)
	Insert	O(log n)	O(log n)	O(log n)
	Search	O(log n)	O(log n)	O(log n)
Traversal		O(n)	O(n)	O(n)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Depth-First Search (In-order, pre-order, and post-order traversal)	O(n)	O(n)	O(n)	O(n)
Breadth-First Search (Level-order traversal)	O(n)	O(n)	O(n)	O(n)
Tree Sort	O(n^2)	O(n log n)	O(n log n)	O(n)
Splaysort	O(n log n)	O(n log n)	O(n)	O(n)
Cartesian Tree Sort	O(n log n)	O(n log n)	O(n)	O(n)

Graphs

(Space-time complexity)

Time Complexity:

		Worst Case Scenario	Average Case Scenario	Best Case Scenario
Insert Vertex	Adjacency List	O(1)	O(1)	O(1)
	Adjacency Matrix	$O(V^2)$	$O(V^2)$	$O(V^2)$
	Incidence Matrix	O(V*E)	O(V*E)	O(V*E)
Remove Vertex	Adjacency List	O(E)	O(E)	O(E)
	Adjacency Matrix	$O(V^2)$	$O(V^2)$	$O(V^2)$
	Incidence Matrix	O(V*E)	O(V*E)	O(V*E)
Insert Edge	Adjacency List	O(1)	O(1)	O(1)
	Adjacency Matrix	O(1)	O(1)	O(1)
	Incidence Matrix	O(V*E)	O(V*E)	O(V*E)
Remove Edge	Adjacency List	O(V)	O(V)	O(V)
	Adjacency Matrix	O(1)	O(1)	O(1)
	Incidence Matrix	O(V*E)	O(V*E)	O(V*E)
Check if Vertices Adjacent	Adjacency List	O(V)	O(V)	O(V)
	Adjacency Matrix	O(1)	O(1)	O(1)
	Incidence Matrix	O(E)	O(E)	O(E)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Breadth-First Search	O(V+E)	O(V+E)	O(V+E)	O(V)
Depth-First Search	O(V+E)	O(V+E)	O(V+E)	O(V)
A* Search	O(E)	O(E)	O(E)	O(V)
Dijkstra’s algorithm	$O(V^2)$	O(E * log(V))	O(E * log(V))	O(V)
Floyd–Warshall	$O(V^3)$	$O(V^3)$	$O(V^3)$	$O(V^2)$

Maps

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Updating an element	O(n)	O(1)	O(1)
Inserting an element	O(n)	O(1)	O(1)
Deleting an element	O(n)	O(1)	O(1)
Searching for an element	O(n)	O(1)	O(1)
Insert (TreeMap)	O(log n)	O(log n)	O(1)
Delete (TreeMap)	O(log n)	O(log n)	O(1)
Search (TreeMap)	O(log n)	O(log n)	O(1)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Bucket Sort (k = buckets)	O(n^2)	O(n + k)	O(n + k)	O(n)
Insertion Sort	O(n^2)	O(n^2)	O(n)	O(1)
Selection Sort	O(n^2)	O(n^2)	O(n^2)	O(1)
Heapsort	O(n log(n))	O(n log(n))	O(n log(n))	O(1)
Hash-based Search	O(n)	O(1)	O(1)	O(1)
Binary Search	O(log(n))	O(log(n))	O(1)	O(1)
Linear Search	O(n)	O(n)	O(1)	O(1)
Rabin-Karp Algorithm	O(m*(n-m+1))	O(n + m)	O(m)	O(m)

Heaps

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Insert	O(log n)	O(logn)	O(1)
Delete	O(log n)	O(log n)	O(1)
Find min/max	O(1)	O(1)	O(1)
Search	O(n)	O(n)	O(1)
Insert (Fibonacci/Binomial)	O(log n)	O(1)	O(1)
Increase/Decrease key	O(log n)	O(log n)	O(1)
Extract min/max	O(log n)	O(log n)	O(log n)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Heapsort	O(n log(n))	O(n log(n))	O(n log(n))	O(1)
Smoothsort	O(n log(n))	O(n log(n))	O(n)	O(n)
Quick select	O(n^2)	O(n)	O(n)	O(1)
Linear Search	O(n)	O(n)	O(1)	O(1)
Dijkstra’s shortest path	O(V^2)	O(E * log(V))	O(E * log(V))	O(V)