

Data Structures

- Linear
 - Array
 - Static Array
 - `class Array`
 - Dynamic Array
 - `class BitArray`
 - `class ArrayList`
 - `class List<T>`
 - Linked List
 - Singly Linked List
 - Doubly Linked List
 - `class LinkedList<T>`
 - Circular Linked List
 - Skip List
 - Stack
 - Array-based Stack
 - `class Stack<T>`
 - Linked List-based Stack
 - Queue
 - Simple Queue
 - `class Queue<T>`
 - Deque (Double-Ended Queue)
 - `class LinkedList<T>`
 - Priority Queue
 - Circular Queue
- Non-Linear
 - Tree
 - Binary Tree
 - Full Binary Tree
 - Complete Binary Tree
 - Perfect Binary Tree
 - Binary Search Tree (BST)
 - AVL Tree
 - Red-Black Tree
 - Heap
 - Min-Heap
 - Max-Heap
 - Trie Tree
 - Prefix Tree
 - Multiway Tree
 - B-Tree

- B+ Tree
 - Quadtree
 - Octree
- Graph
 - Directed / Undirected
 - Weighted / Unweighted
 - Representation
 - Adjacency Matrix
 - Adjacency List
 - Adjacency Multilist
 - Orthogonal List
 - Special Graphs
 - DAG
 - Tree Graph
 - Bipartite
- Set
 - Hash Set
 - `class HashSet<T>`
 - Sorted Set
 - `class SortedSet<T>`
 - Multiset/Bag
 - Bit Set
- Hash/Dictionary
 - Hash Table
 - `class Hashtable`
 - `class Dictionary<TKey,TValue>`
 - Sorted Map
 - `class SortedList`
 - `class SortedList<TKey,TValue>`
 - `class SortedDictionary<TKey,TValue>`
 - Bi-directional Map
 - MultiMap
- Specialized / Utility
 - Ordered
 - `class OrderedDictionary`
 - ReadOnly
 - `class ReadOnlyCollection<T>`
 - `class ReadOnlyDictionary<TKey,TValue>`
 - `interface IReadOnlyList<T>`
 - `interface IReadOnlyDictionary<TKey,TValue>`
 - Data Binding
 - `class ObservableCollection<T>`

- String
 - `class StringCollection`
 - `class StringDictionary`
- Bit Operation
 - `class BitArray`
 - `class BitVector32`
- Hybrid
 - `class HybridDictionary`
 - `class NameValueCollection`
- Immutable
 - `class ImmutableArray<T>`
 - `class ImmutableList<T>`
 - `class ImmutableHashSet<T>`
 - `class ImmutableDictionary<TKey,TValue>`
 - `class ImmutableSortedDictionary<TKey,TValue>`
- Disjoint Set / Union-Find
- Bloom Filter
- Counting Filter
- Linear Probing Table
- Concurrent
 - `class ConcurrentQueue<T>`
 - `class ConcurrentStack<T>`
 - `class ConcurrentBag<T>`
 - `class ConcurrentDictionary<TKey,TValue>`
 - `class BlockingCollection<T>`