Data Structures

- Linear
  - Array
    - Static Array
      - `class Array`
    - Dynamic Array
      - `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
      - By Structural Property
        - Full Binary Tree
        - Complete Binary Tree
        - Perfect Binary Tree
      - By Functional Purpose
        - Binary Search Tree (BST)
          - Balanced BST
            - AVL Tree
            - Red-Black Tree
          - Unbalanced BST
        - Heap
          - Min-Heap
          - Max-Heap

            - Huffman Tree
            - Segment Tree
            - KD-Tree
        - Multiway Tree
            - Multiway Search Trees
                - B-Tree
                - B+ Tree
                - 2-3 Tree
                - 2-3-4 Tree
            - Trie Tree
                - Standard Trie
                - Compressed Trie
                - Suffix Tree
            - Space Partitioning Trees
                - Quadtree
                - Octree

    - Graph

        - Directed / Undirected
        - Weighted / Unweighted
        - Representation
            - Adjacency Matrix
            - Adjacency List
            - Adjacency Multilist
            - Orthogonal List
        - Special Graphs
            - DAG
            - Tree Graph
            - Bipartite

- Set

    - `class HashSet<T>`
    - `class SortedSet<T>`

- Hash

    - `class Hashtable`
    - `class Dictionary<TKey,TValue>`
    - `class OrderedDictionary`

- Specialized / Utility

    - Sorted
        - `class SortedList`
        - `class SortedDictionary<TKey,TValue>`
        - `class SortedList<TKey,TValue>`
    - 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>`