26120 Algorithms and Data Structures (Sem1)

Big-O (O) is *tight upper bound* - the most amount of time required (worst case performance).
Little-o (o) notation is used to describe an *loose upper bound* that cannot be tight.

Big Theta notation (Θ) is the *tightest bound* - the best of all the worst-case times.

Big Omega (Ω) is *tight lower bound* - the least amount of time required (best case).
Little Omega (ω) notation is used to describe an *loose lower bound* that cannot be tight.

Iterative Method calculate the complicity using definitions, by identifying all operations in the algorithm and then add them together. This may be time-costly, so quicker methods may be preferred.

Substitution Method verify the experience-guessed complexity by first guessing a solution and verify it by substituting the supposed answer into the last level result part into the formula.

Master Method calculate the complexity through pre-concluded formulas. $f(n)$ must be asymptotically positive (always positive for sufficiently large n).

$$T(n) = a \times T(n/b) + f(n) = \begin{cases} \Theta\left(n^{\log_b a}\right) & f(n) = O\left(n^{\log_b a - \varepsilon}\right) \\ \Theta\left(n^{\log_b a} \times \log n\right) & f(n) = \Theta\left(n^{\log_b a}\right) \\ \Theta\left(f(n)\right) & f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right) \textbf{ AND } af\left(n/b\right) \le cf(n) \text{ for large } n \end{cases} \quad \begin{matrix} a \ge 1 \\ b > 1 \\ c < 1 \\ \varepsilon > 0 \end{matrix}$$

Disjoint Sets are collections of distinguish entities that have no elements in common. A group of disjoint sets could be merged. Actions on these sets are: $add(x)$ create a new set with x; $find(x)$ find the set that contains x; and $union(x, y)$ that merge the sets x with y.

Union of Two Disjoint Sets attaches the root, or eventual parent, of one node to that of another as a child. The root of set with more nodes will become the root of the new, unionised disjoint set.

Path Splitting of Disjoint Set changes the parent node of the current node to, where possible, the grandparent node (the parent of their parent). Preformed along the route of finding, after the change the finding operation continues with the *old parent node*.

Path Halving of Disjoint Set changes the parent node of the current node to, where possible, the grandparent node (the parent of their parent). Preformed along the route of finding, after the change the finding operation continues with the *new parent node* (the old grandparent node).

Path Compression of Disjoint Set sets all the node along the finding route to the eventual parent. It is preformed during find operations, and nodes that was not in the trail remains *unaffected*.

Amortised Cost is the effort needed to preform each single action in a larger set, with total time divided by actions. It is not a magic tool, sometimes it does not make a difference or is not applicable, and it does not reflect the worst-case situation, nor does it reflect the total time needed as a whole.

Quick Sort work by a greedy divide-and-conquer. It separates the larger array into two smaller ones based on a (potentially) randomly selected divider.

Min-Heap is *essentially a binary tree* (not a search tree) where all children nodes have values larger than their sole parent. The smallest node will always be the root node. It is usually stored in an array, with children being 2n and 2n+1 (first location being 1 than 0). Heap is sorted by exchanging larger parents with its smallest child. Popping out an element will let the last value in array be filled in the first place and sort the heap.

Skip List is a data structure that has $O(\log n)$ complexity for searching. It is a multi-layer structure with higher layers having lower number of nodes (e.g. 50%) and serve as index by linking to the corresponding nodes in lower layers. Similar time complexity with AVL but it is simpler, faster and utilises less space.

Height-Balance Property exists when number of nodes on the left branch for each parent node is *at most* 1 node larger or smaller than that of the right branch. This property exists for all AVL trees or balanced trees. Do not necessarily exist with min-heap have depth higher than 2.
