Problem 1. *Modify the QUICKSORT LOMUTO PARTITION algorithm so that the largest element is always selected as the pivot. What is the running time? (CODE QUESTION) run a test case*

Problem 2. *Compare Merge Sort and QuickSort in terms of space complexity and cache performance.*

Problem 3. *Illustrate the operation of HOARE PARTITION on the array (No sorting required)*

$$A = \langle 13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11 \rangle$$

Problem 4. *Illustrate the operation of LOMUTO PARTITION on the array (No sorting required)*

$$A = \langle 13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11 \rangle$$

Problem 5. *You are given the following array of numbers:*

$$A = \langle 50, 51, 52, 53, 54, 55, 1, 2, 3, 4, 5, 6 \rangle$$

You need to sort this array efficiently using either QuickSort with the Lomuto partition scheme or MergeSort. Which algorithm should you choose and why?

Problem 6. *Assume that QuickSort is applied to an array of size n, but the partitioning process is highly unbalanced such that one partition always contains (0.995)n elements and the other contains only (0.005)n elements.*

1. *Write the recurrence relation for this partitioning scenario.*
2. *Find the approximate height of the recursion tree.*
3. *Compare this unbalanced partitioning to the ideal case (when the pivot splits the array into two halves) and discuss the impact on overall sorting performance.*

Problem 7. *For the set of* $\{1, 4, 5, 10, 16, 17, 21\}$ *of keys, draw binary search trees of heights 2, 3, 4, 5, and 6.*

Problem 8. *A concatenate operation takes two sets* S_1 *and* S_2*, where every key in S1 is smaller than any key in* S_2*, and merges them together. Give an algorithm to concatenate two binary search trees into one binary search tree. The worst-case running time should be* $O(h)$*, where* h *is the maximal height of the two trees.*

Problem 9. *Explain why a Red-Black Tree ensures that the longest path is at most twice the shortest path.*

Problem 10. *Why does an AVL tree always maintain a balance factor of -1, 0, or 1? What happens if this condition is violated?*

Problem 11. *AVL trees maintain strict balance by performing rotations, but does this result in additional memory usage? explain your answer.*

Problem 12. *Insert the following list in an* ***AVL*** *tree, and start with an empty tree*

$$A = \langle 10, 20, 15, 25, 30, 16, 18, 5 \rangle$$

you always need to check if the tree is balanced after every new element insertion.

Problem 13. *Show that an n-element heap has height* $\lfloor log_2 n \rfloor$
If you can not show it mathematically you can give an example to illustrate your answer

Problem 14. *Show that there are at most* $\lceil \frac{n}{2^{h+1}} \rceil$ *nodes of height h in any n-element heap.*

Problem 15.

$$A = \langle 5, 13, 2, 25, 7, 17 \rangle$$

- *Use the previous list to build a Red-Black tree, show steps of insertion by drawing a new tree for every step, use two different pen colors or shade the black node.*

Problem 16. *Illustrate the operation of HEAP-SORT on list A.*

$$A = \langle 5, 13, 2, 25, 7, 17 \rangle$$

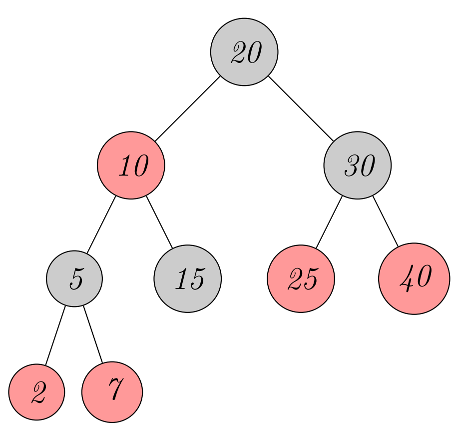
Problem 17. *Use the following formula to argue the run-time of Max-Heap:*

$$\sum_{h=0}^{\lfloor log_2 n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil * O(h)$$

Problem 18. *What are the minimum and maximum numbers of elements in a heap of height h? If you are performing Max-heap - on a complete binary tree- which node should you start with? support your answer with the correct formulas.*

Problem 19. *Where in a max-heap might the smallest element reside, assuming that all elements are distinct?*

Problem 20. *Consider the following Red-Black Tree (RBT):*



Insert the key **6** *into the RBT and:*

1. *Show the tree after insertion (before fixing violations).*
2. *Perform* ***color flips*** *and/or* ***rotations*** *to restore RBT properties. Draw the final tree.*
3. *Justify each step by referencing the RBT invariants.*

Why are Red-Black Trees preferred over AVL trees in certain applications? Discuss time/space tradeoffs.