

**1. (8 points) True or False**

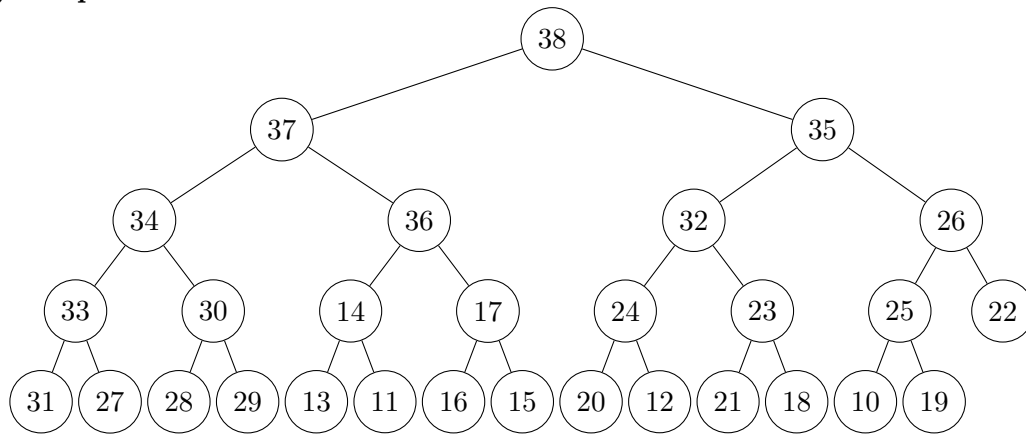
Determine whether the following statements are true or false.

- (a) (1') To print the hierarchy of a folder, we should apply BFS on the folder tree. ☐ True ☒ False
- (b) (1') If the pre-order traversal and in-order traversal of two binary trees are equal respectively, then the two binary trees are exactly the same. ☒ True ☐ False
- (c) (1') Every complete binary tree is also a perfect binary tree. ☐ True ☒ False
- (d) (1') A Huffman Coding Tree is a full binary tree. ☒ True ☐ False
- (e) (1') Heap sort is a stable sorting algorithm. ☐ True ☒ False
- (f) (1') The time complexity of heap sort is  $O(n \log(n))$  in both the best-case and the worst-case. ☒ True ☐ False
- (g) (1') In a BST, there may be a subtree that is not a BST. ☐ True ☒ False
- (h) (1') Suppose there are  $n$  nodes in an AVL tree, its minimum possible height is given by  $\lceil \log_2(n+1) \rceil - 1$ . ☒ True ☐ False

**2. (13 points) Fill in the blanks**

- (a) (2') A full binary tree has 2025 nodes. The number of its leaf nodes is 1013.
- (b) (2') In a binary tree with  $n$  nodes, every node has an odd number of descendants. Then the number of nodes in the tree that have exactly one child is 0.
- (c) (6') In a binary max-heap with  $n$  elements and duplicated elements are not allowed, the  $6^{th}$  largest element can be found in time  $O(\text{log } n)$  if we can only access the top of the heap. And the  $6^{th}$  largest element can be found in time  $O(1)$  if we can access the array storing the heap. And the  $6^{th}$  smallest element can be found in time  $O(n)$  if we can access the array storing the heap. (Give the precise and simplified upper bound)
- (d) (3') Elements [13, 21, 3, 17, 24, 26, 40, 9, 16] are inserted sequentially into an empty BST. Then if element  $x$  is inserted into the BST as the right child of element 9, then all possible integer values for  $x$  could be 10, 11, 12 (all elements are distinct).

3. (9 points) Heap



- (a) (1') Is this heap a max-heap or a min-heap?

**Solution:** max-heap

- (b) (3') Suppose that you pop the key from the heap above. Write down all the involved swapping operations in order. (for example,  $a - b, c - d$  to indicate swapping  $a, b$  and then swapping  $c, d$ )

**Solution:** (19-38,) 19-37, 19-36

- (c) (2') Suppose that you insert a key  $k$  into the heap above, and then it is moved to the place of current 26. Write down the possible value range of  $k$ .

**Solution:** Between 26 and 35.

- (d) (3') Suppose that inserting the key  $x$  was the last operation performed in the binary heap in the figure. That is, after inserting  $x$ , the heap is shown in the figure above. Write down all possible values of  $x$ .

**Solution:** 19,25,26.

To insert a node in a binary heap, we place it in the next available leaf node and swim it up. Thus, 19,25,26,35,38 are the only keys that we might move. But, the last inserted key could not have been 35 and its ancestors, because, then, 26 would have been at the place of current 35 (which would violate heap order because the left child of 26 is 32).