## 0.1 Which of the following statements are true? (check all that apply)

1. The height of a binary search tree with $n$ nodes cannot be smaller than $\Theta(\log n)$.
2. All the operations supported by a binary search tree (except OUTPUT-SORTED) run in $O(\log n)$ time.
3. The heap property is a special case of the search tree property.
4. Balanced binary search trees are always preferable to sorted arrays.

Only (1) is correct. Let $h$ be the height of the tree. the number of nodes in the tree are at most

$$\begin{aligned} 1 + 2 + 3 \ldots + 2^h &= 2^{h+1} - 1 \\ &\leq 2^{h+1} \end{aligned}$$

hence the number of nodes $n$ has the equality

$$\begin{aligned} n &\leq 2^{h+1} \\ \log n &\leq h + 1 \\ \log n &\leq 2h \\ \frac{\log n}{2} &\leq h \end{aligned}$$

from this we see the height $h$ is at least order of $\log n$, thus $h = \Omega(\log n)$.

## 0.2 You are given a binary tree with $n$ nodes. Each node of the tree has a size field, but these fields have not been filled in yet. How much time is necessary and sufficient to compute the correct value for all the size fields?

1. $\Theta(height)$
2. $\Theta(n)$
3. $\Theta(n \log n)$
4. $\Theta(n^2)$

(2) is correct. Let $n$ be the number of nodes in the tree. Since we are required to compute the size field for every node we must visit all $n$ nodes. Hence we have order of n, thus $\Theta(n)$.