1. Compare the trees in Test A and Test B.
The Test A tree is built from a sorted list, ascending.
The Test B tree is built form a sorted list, descending.
Which tree was built faster and had faster processing? Why do you think that is?
Hint: Try drawing a small tree to see what is going on. Perhaps a tree built from 1, 1, 2, 2, 3, 4, 4, 5 and from 5, 4, 4, 3, 2, 2, 1.

Tree A was built in 615 sec and Tree B was built in 7529 msec so clearly Tree A was built more quickly. The main reason for the performance difference is that Tree B is significantly higher: Tree A 4787 height vs. Tree B 39999. In Tree B, because we default to adding duplicates down to the left, ALL nodes connect from the left side and to traverse this tree to add the next node is a O(n) operation, whereas Tree A is better balanced. We see further evidence for this in the “Tree Time” data where Tree A operations occur too quickly to measure and Tree B operations are slower.

2. Compare the trees in Test A and Test C.
The Test A tree is built from a sorted list, ascending.
The Test C tree is built from a shuffled list.
Which tree was built faster? What does the printed "tree info" tell you about these two trees?

Tree A was built in 615 msec and Tree C was built in 19 msec. Tree C was significantly faster. The Tree Info data tells me that Tree C was built much more efficiently and resulted in a much more balanced tree. Tree A: Tree Height 3 (left) 4787 (right) vs Tree C: 39 (left) 161 (right). Tree C had nearly as many nodes on the left of root vs. the right of root. Tree A had nearly all nodes on the right of root.

3. Compare the tree in Test C to the shuffled list in Test C.
The Test C tree is built from the shuffled list.
Which processing was faster: the tree or the list?
How would you describe the big-o of the processing of these two structures?

The tree processing was significantly faster vs. list. Because the list is unsorted, the entire list must be searched to find reports for a given date or O(n). An efficient tree like Tree C, search will be O(log n).

4. What is the main characteristic of a binary search tree that affects its efficiency?

The balance of the tree, meaning the even distribution of values on the left and right side of a node or that the height of the tree on the left side is approximately the same as the height on the right.