Module 4 Creativity Exercise

5.7.24)

This is a good candidate for a BFS of a min heap utilizing a queue. Start with your root node and check if the root node val is <= k. If it is, add to your ans array. Then check the left child and if the val is <= k add to the queue and do the same with the right child. Continue the loop until the queue is empty meaning there are no other values that could be less than or equal to k (give the property of a min heap). This will have an O(k) time complexity since the queue will be empty once the BFS reaches a point where all values left in the heap are greater than the search value of k.

#start code, assume root node is given and node class has properties val, leftChild(), rightChild()

```
def find_all_values_less_than_k(root, k):
    queue ← [root]
    ans ← []
    while queue:
        root ← queue.dequeue()
        if root.val <= k:
            ans.append(root)
        if root.leftChild() is not null and root.leftChild().val <=k:
            queue.enqueue(root.leftChild())
        if root.rightChild() is not null and root.rightChild().val <= k:
            queue.enqueue(root.rightChild())
    return ans
```

Time Complexity: O(k).

Algorithm will only be enqueueing nodes with values less than or equal to k.