I hereby pledge on my honor that I will strictly adhere to academic integrity codes and the work done or this examination is solely my own and I will not recieve/give any help from/to anybody or source during this examination.

Berke Bilgin
BB

1-

```
public void bfs(int depth, Node node){          (initial value is 1)
        if(node == null) return;
        if(node == root) System.out.println(node.data.toString() + " " + depth);
        if(node.left != null) System.out.println(node.left.data.toString() + " " +
                                                                   (depth + 1));
        if(node.right != null) System.out.println(node.right.data.
                                                   toString() + " " +
                                                   (depth + 1));
        bfs(depth +1, node.left);
        bfs(depth+1, node.right);
}
```

$O(n) = \Omega(n) = \Theta(n)$ (Traverses every element once)

2-

a) Best-Case $\rightarrow \Omega(n\log n)$
Avg-Case $\rightarrow \Theta(n\log n)$
Worst-Case $\rightarrow O(n^2)$

(• "It is still ongoing Open Research, the time Complexity of QuickSort in Computer Science"
• Average Case of Quick Sort can be $n^{3/2}$ or $n^{5/4}$ depending on the situation)

(I saw both in the slides. I picked $n\log n$ one.)

b)

$\Omega(n\log n)$

c)

1- QuickSort $\Omega(n\log n)$
2- MergeSort $O(n\log n)$
3- Merge Sort

Scanned by TapScanner