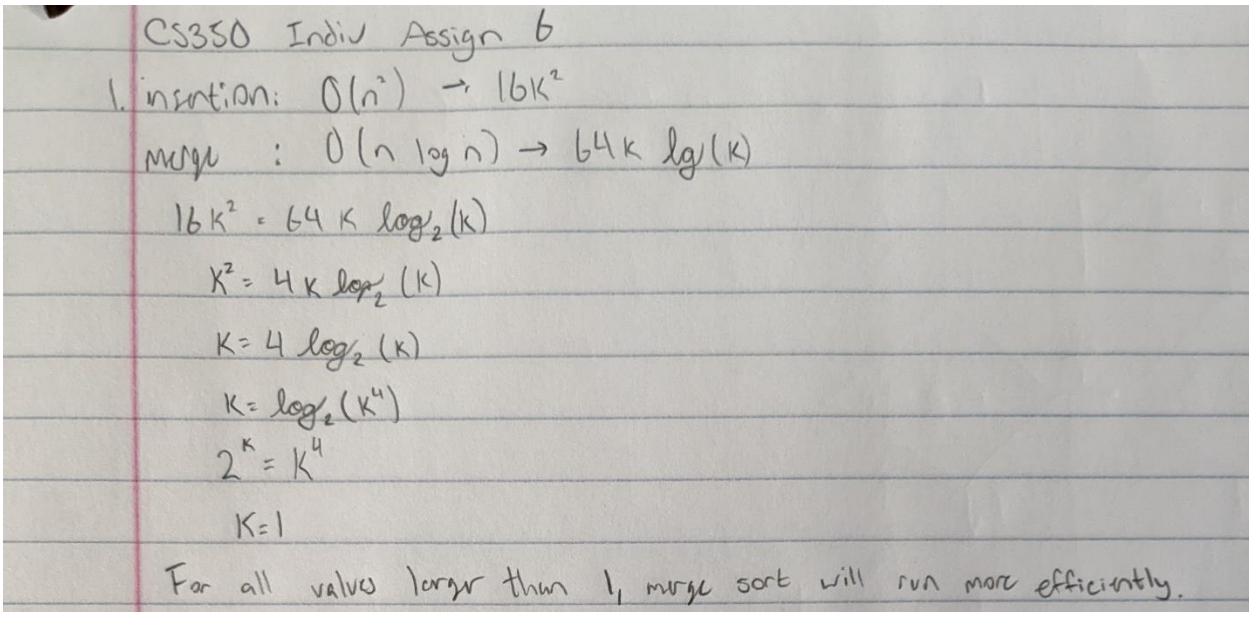
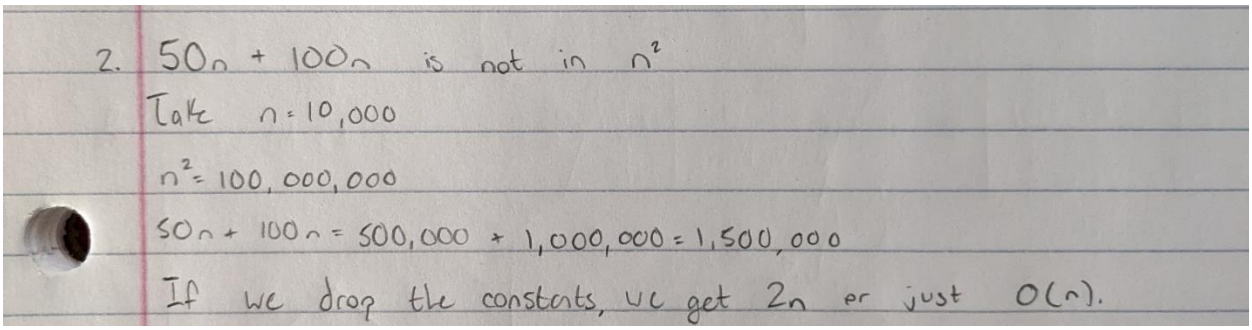
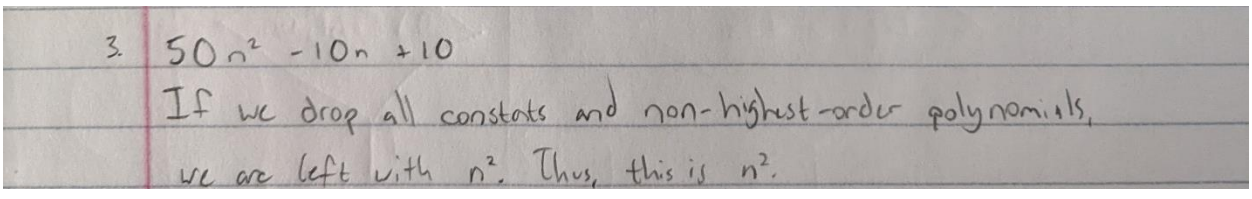
1. **Assume that we are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size k, insertion sort runs in $16k^2$ steps, while merge sort runs in 64k*lg(k) steps. For which values of k does insertion sort beat merge sort?**



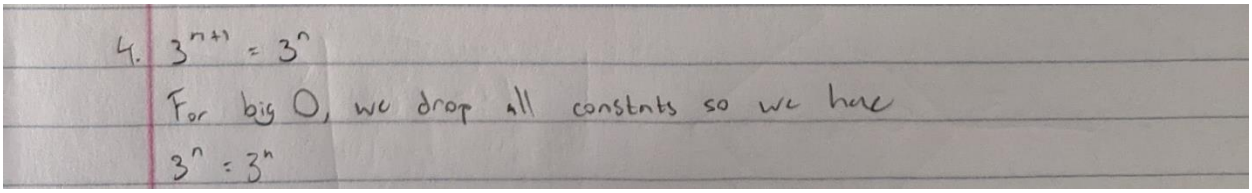
2. **Prove that $50n + 100n$ is in $O(n^2)$.**



3. **Prove that $50 n^2 - 10n + 10$ is in $O(n^2)$.**



4. Is $3^{n+1} = O(3^n)$? please write down your answers with detailed steps.



5. Write Java functions to implement Big-O notations below.

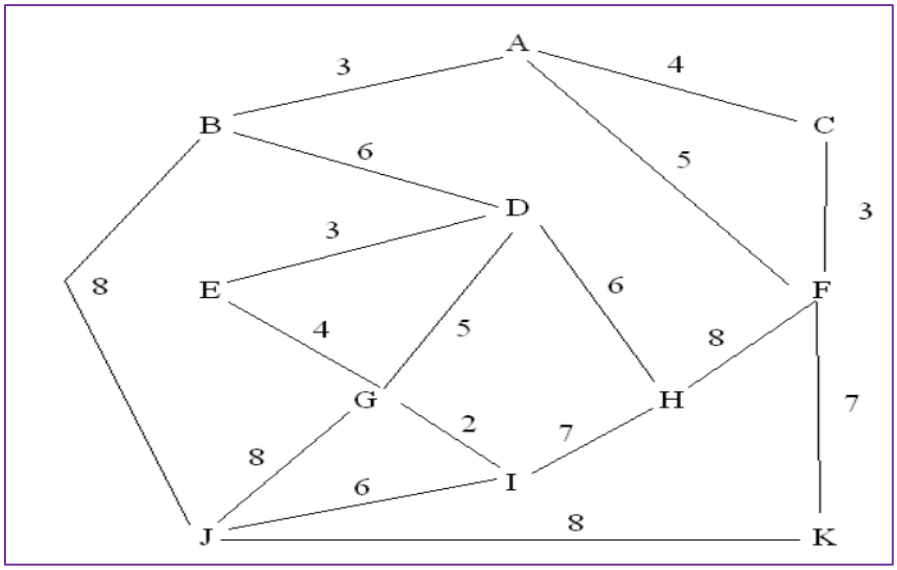
[1] $O(\log n)$ [2] $O(n \log n)$ [3] $O(\sqrt[3]{n})$ [4] $O(\sqrt{n})$ [5] $O(n^3)$

```
public static void main(String[] args) {
    int n = 1000;

    // log n
    for (int i = 0; i < n; i *= 2) {
    }

    // n log n
    for (int i = 0; i < n; i++) {
      for (int j = n; j > 0; j /= 2) {
      }
    }

    // cube root
    for (int i = 0; i < Math.pow(n, 1/3); i++) {
    }

    // sqrt
    for (int i = 0; i < Math.sqrt(n); i++) {
    }

    // cubed
    for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
        for (int k = 0; k < n; k++) {
        }
      }
    }
  }
```

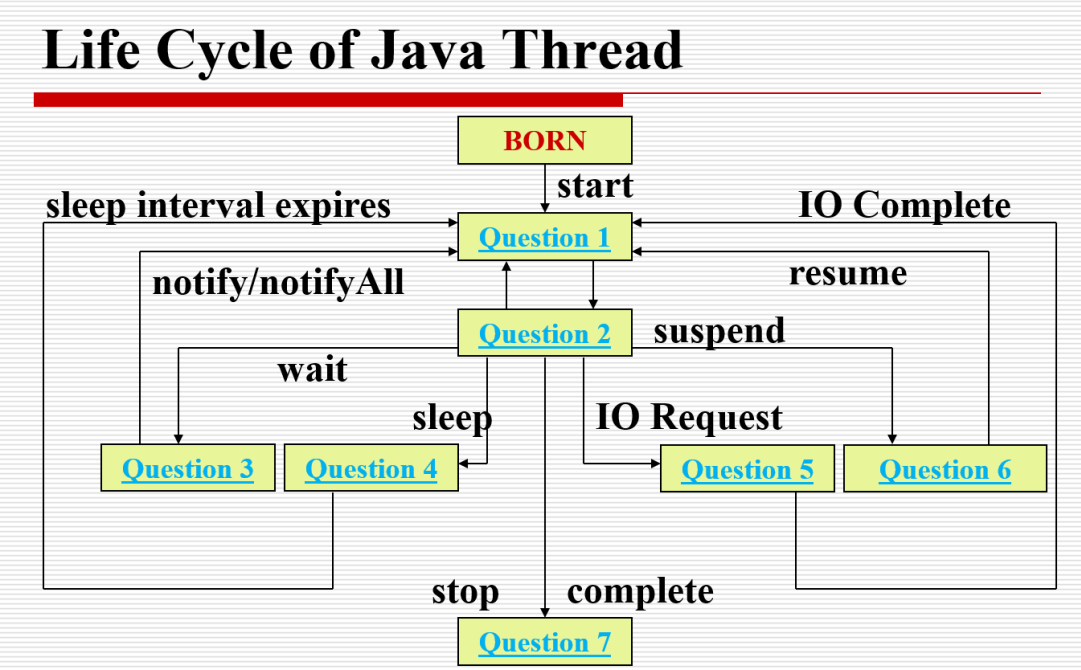
1. **Consider the graph in Figure 1 below. Unless otherwise indicated, always visit adjacent nodes in alphabetical order. (20% points, 10% each)**

2. **DFS algorithm traverses all nodes starting at node D**
   D -> E -> G -> J -> B -> A -> C -> F -> K -> H -> I
3. **BFS algorithm traverses all nodes starting at node G**

   J -> K -> F -> C -> A -> B -> D -> E -> G -> I -> H



2. Consider the graph in Figure 1 below. Please fill in Questions 1-7.



1. READY
2. RUNNING
3. WAITING
4. SLEEPING
5. BLOCKED
6. SUSPENDED
7. DEAD