Information you might need.

1. Master Formula

Suppose T(n) satisfied

$$T(n)=\begin{cases} d & if\ n=1 \\ aT(\left\lceil \frac{n}{b} \right\rceil)+cn^k & otherwise \end{cases}$$

Where k is none negative integer and a, b, c, d are constants with a>0, b>1, c>0, d>=0 then

$$T(n)=\begin{cases} \Theta(n^k) & if\ a<b^k \\ \Theta(n^k \log n) & if\ a=b^k \\ \Theta(n^{\log_b^a}) & if\ a>b^k \end{cases}$$

2. $x=b^y \Longrightarrow Log_b x=y$

3. $\sum_{i=0}^{n-1} i=\frac{n(n-1)}{2}$

Q1) (14 points) Complexity Analysis

1. [4 points] What is the worst case running time in Big-Oh notation for the following functions? Show your work for partial credit.

a.
```
void foo(int n )
{
   int count = 0;
   for (int i = 0; i < n; i++)
      for (int j = i; j^2 > 0; j--)
         count = count + 1;
  return count;
}
```

b.
```
void foo(int n)
  {
     m = 0;
     while(n>=2)
      {
         n = n/3;
         m = m+1;
         System.out.println(m);
      }
     return m
  }
```

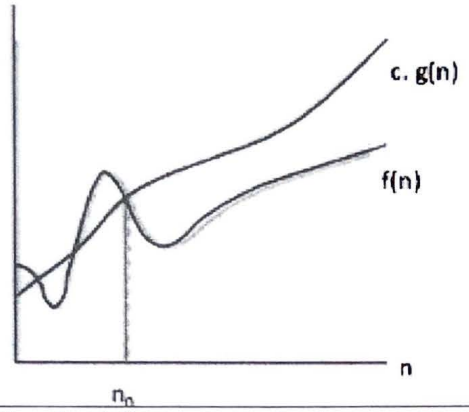
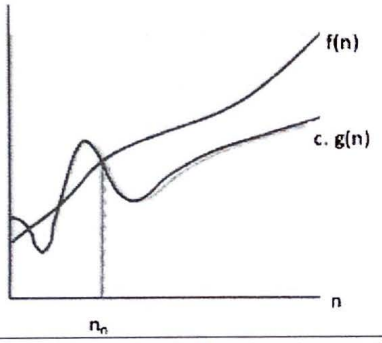
2. [4 points] Discuss whether the next statements are true for the given function $f(n) = 2n^2 + n\log n$

a) $f(n) = O((n^2 \log n)$

b) $f(n) = \Theta(n^2 \log n)$

3. [6 points] Multiple Choice Questions:

1.	What is the tightest asymptotic bound for the following f(n) $f(n) = 2n^n + 2^{100}$	✓a. $O(n^n)$ b. $O(2^{100})$ c. $O(2n^n)$
2.	What is the tightest asymptotic bound for the following f(n) $f(n) = n^2 \log(n^5) + 5n \log(n^5) + n^3$	a. $O(n^2 \log(n^5))$ b. $O(n^2 \log n)$ ✓c. $O(n^3)$ d. $O(5n \log(n^5))$

3.	For the following graph, the asymptotic relationship of f and g functions is  	a. f(n) = Theta (g(n)) ✓b. f(n) = O (g(n)) c. f(n) = Big-Omega (g(n))
4.	For the following graph, the asymptotic relationship of f and g functions is  	a. f(n) = Theta (g(n)) b. f(n) = O (g(n)) ✓c. f(n) = Big-Omega (g(n))
5.	Which of the following is not $O(n^3)$	A. $N^{2.98}$ B. 15^{10} n + 1000 ✓ C. $n^4 / n^{1/2}$ D. 2^{20} n^2
6.	Which of the given options provide the increasing order of asymptotic complexity of functions f1, f2, f3 and f4? f1(n) =2n f2(n) = n! f3(n) = n logn f4(n) = 7^n	a. f3, f2, f4, f1 b. f3, f2, f1, f4 c. f2, f3, f1, f4 ✓d. f1, f3, f4, f2

Q2) (11 points) Analysis

1. [4 points] Give tight asymptotic bound for the following recurrences:
   a. $T(n) = 4T(n/2) + 1$

   b. $T(n) = T(n/2) + n^3$

2. [5 points] Given an array A with n integer elements. Write an algorithm to print the largest number and the smallest number in the array. Propose an algorithm to solve this problem in O(n) worst-case time or better.
   Example: A = {1, 3, 2, 1, 3, 5, 3} the result should be the largest number is 5 and the smallest number is 1.

```
public void search1(int[] input) {
    int maxValue = Integer.MIN_VALUE;
    int minValue = Integer.MAX_VALUE;
    for (int i=0; i < input.length; i++) {
        if(input[i] < minValue)
            minValue = input[i];

        if(input[i] > maxValue)
            maxValue = input[i];
    }
    System.out.println("maxValue: " + maxValue);
    System.out.println("minValue: " + minValue);
}
```

We need to transform this in pseudo-code sintax

3. [2 points] Verify whether the following statement is true or false. The amortized cost of a sequence of n operations from S is O(n), and so the amortized cost of a single operation from S is O(n).
   Given S consists of two operations:
   a. add(x) = inserts String x into next available slot
   b. clear() = replaces all Strings in array with nulls

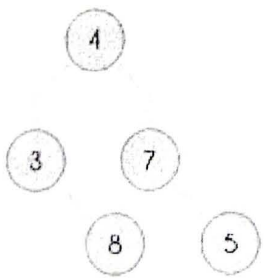
Conclusion. Therefore, the amortized cost of a sequence of n operatons from S is $O(2n) = O(n)$, and so the amortized cost of a single operation from S is $O(n)/n = O(1)$.

Q3) (14 points) Sorting algorithms:

1. [5 points] Answer the following questions by True or False
   × a. It is possible to develop a comparison based sorting algorithm that runs in O (n).
   × b. The best time complexity of Bubble Sort is O(n log n).
   × c. Merge Sort makes more swap operations than Selection-Sort.
   × d. Suppose we have a O(n) time algorithm that finds median of an unsorted array. Consider a QuickSort implementation where we first find median using the above algorithm, then use median as pivot. The worst case time complexity of this modified QuickSort is O(n)
   ✓ e. Insertion-Sort is stable sorting algorithm.

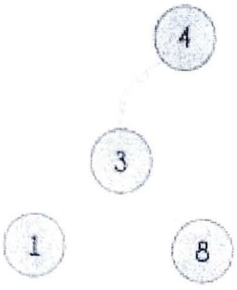
A	B	C	D	E

2. [3 points] Which of the following trees is a heap? Explain your answer for each tree.

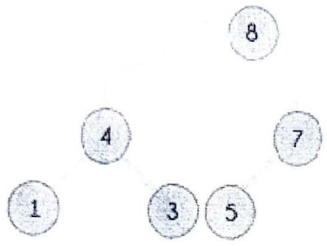


a.

b.



c.



According to my understanding this is a Heap Tree, because this is a Max-Heap. which means the root is the max number of all nodes.

3. [2 points] Given that the running time worst case for merge-sort is better than quicksort, why quick-sort is commonly used?

Comparison With MergeSort

- MergeSort's O(nlog n) worst-case running time makes it reliable, but in practice QuickSort is faster
- Reason for QuickSort's faster speed: MergeSort makes many copies of portions of array, increasing overhead.
- Perspective on QuickSort's worst-case: In sorting 1 thousand arrays of size approx 1 million, the probability that QuickSort will perform sorting less efficiently then O(nlog n) is less than 1/1 billion. More likely system would crash.
- MergeSort's style of writing to memory can be adapted to efficiently handle extremely large sorting jobs, where all data cannot fit into memory (so repeated disk reads are necessary)

4. [4 points] Use Merge Sort Algorithm to sort the following array of integers. Show the merge-sort tree.

30 25 10 80 20 15 99 88