1. For each of the following recurrences, give an expression for the runtime $T(n)$ if the recurrence can be solved with the Master Theorem. Otherwise, indicate that the Master Theorem does not apply.

a. $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log(n)}$
b. $T(n) = \frac{1}{2}T\left(\frac{n}{2}\right) + n$
c. $T(n) = T\left(\frac{3n}{4}\right) + 1$
d. $T(n) = 7T\left(\frac{n}{3}\right) + n^2$
e. $T(n) = 2T\left(\frac{n}{2}\right) + n\log(n)$
f. $T(n) = T\left(\frac{n}{2}\right) + n(2.\cos(n))$

2. Two students delay the time to return some books to the university library.

a. The first student is a bit busy. This student should do 4 math problems for his class before returning the books. Each problem requires one day with probability of $\frac{3}{4}$ and two days with probability of $\frac{1}{4}$. If A is the number of days the students delays returning the books, what is E[A]?
   Example: The first and fourth math problems require 1 day, and the second and third math problems require 2 days. $\mathrm{A} = 6$.

b. The second student is a bit relaxed. He rolls a dice in the morning. If he rolls a 6, then he returns the books immediately (zero days of delay). Otherwise, he delays for one day and repeats this process the next morning. If B is the number of days the student delays returning the books, what is E[B]?
   Example: The student rolls a 4 the first morning, a 2 the second morning, and a 6 in the third morning. $\mathrm{B} = 2$.

3. Arrange the following functions in increasing asymptotic order.

$$n^{\log(\log(n))}, n^3, n \times 2^n, n^{\frac{1}{\log(n)}}, (\log(n))!, \log(\log(n)), 8^{\log(n)}$$

4. Suppose that we are given n points in a plane, each given by a pair of real numbers. We want to find which pair of points has the shortest distance between them. We can do this in $\theta(n^2)$ time by computing the distance for each of the pairs and taking the minimum of all these distances. But we want to use a recursive algorithm to solve this problem in time $O(n\log(n))$.

5. Let $f(n)$ and $g(n)$ be asymptotically positive functions. Prove or disapprove following conjectures.

a. $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$

b. $f(n) + o(f(n)) = \theta(f(n))$

c. $f(n) = \theta\left(f\left(\frac{n}{2}\right)\right)$

6. Describe a $\theta(n\log(n))$ - time algorithm that, given a set S of n integers and another integer x, determines whether or not there exists two elements in S whose sum is exactly x.

7. Observe that the **while** loop of lines 5–7 of the `INSERTION-SORT`, uses a linear search to scan (backward) through the sorted subarray A[1..j − 1]. Can we use a binary search instead to improve the overall worst-case running time of insertion sort to $\theta(n\log(n))$.

```
INSERTION-SORT(A)
1 for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1..j-1].
4      i = j-1
5      while i > 0 and A[i] > key
6             A[i+1] = A[i]
7             i = i-1
8      A+1 = key
```

8. If $T(n)$ indicates number of stars printed with `Mystery(n)` function, write its appropriate recurrence relation.

```
Mystery(n)
if n >= 2
     Mystery(n-1)
     print ***
     Mystery(n-2)
     print ****
     Mystery(n-1)
```

9. Six pirates must divide $300 dollars among themselves. The division is to proceed as follows. The senior pirate proposes a way to divide the money. Then the pirates vote. If the senior pirate gets at least half the votes he wins, and that division remains. If he doesn't, he is killed and then the next senior-most pirate gets a chance to do the division. Now you have to tell what will happen and why (i.e. , how many pirates survive and how the division is done)? All the pirates are intelligent and the first priority is to stay alive and the next priority is to get as much money as possible.

10. Reconsider the pirate problem above, where only one indivisible dollar is to be divided. Who gets the dollar and how many are killed?

11. Suppose we start with n companies that eventually merge into one big company. How many different ways are there for them to merge?

(TA)

12. Compute 2101×1130 by applying the divide-and-conquer algorithm. (Multiplication of Large Integers)

13. Solve the recurrence for the number of additions required by Strassen's algorithm. Assume that n is a power of 2.

(Strassen's Matrix Multiplication)