M2: Divide and Conquer - Recurrence Relations

This homework will ask you to think about Divide and Conquer Algorithms with a focus on recurrence relations (this is module 2 after all!). Please try to follow the guidelines below when writing up your solutions to these problems:

- Algorithm descriptions should be about 1 paragraph long. If you are writing more than one paragraph to describe your algorithm, then it is TOO long.
- There should be enough detail that I could implement the algorithm if I wanted to, but we don't need to see code. If you want to add pseudo-code for clarity, that would be fine.
- For the recurrence relation problems below, make sure to show your work clearly!

1. You are a hacker, trying to gain information on a secret array of size n. This array contains $n-1$ ones and exactly 1 two; you want to determine the index of the two in the array.

   Unfortunately, you don't have access to the array directly; instead, you have access to a function $f(l1, l2)$ that compares the sum of the elements of the secret array whose indices are in $l1$ to those in $l2$. This function returns -1 if the $l1$ sum is larger, 0 if they are equal, and 1 if the sum corresponding to $l2$ is larger.

   For example, if the array is $a = [1, 1, 1, 2, 1, 1]$ and you call $f([0, 2, 4], [1, 3, 5])$ then the return value is 1 because $a[0] + a[2] + a[4] = 3 < 4 = a[1] + a[3] + a[5]$. Design an algorithm to find the index of the 2 in the array using the least number of calls to $f()$. Then, answer the following questions:

   - Describe your algorithm clearly (in a paragraph or so)
   - Give the recurrence relation for the runtime of your algorithm. Make sure to write this in terms of f_r, which we will use to represent the runtime of $f()$.
   - Suppose you discover that $f()$ runs in $\Theta(max(|l1|, |l2|))$, what is the overall runtime of your algorithm in this case?

Directly solve, by unrolling the recurrence, the following relation to find its exact solution. Make sure to show your work.

2. $T(n) = T(n-1) + n$

Use induction to show bounds on the following recurrence relations.

3. Show that $T(n) = 4T(\frac{n}{3}) + n \in O(n^{log_3(4)})$. You'll need to subtract off a lower-order term to make the induction work here.

Use the master theorem to solve the following recurrence relations. State which case of the theorem you are using and why.

4. $T(n) = 2T(\frac{n}{4}) + 1$

5. $T(n) = 2T(\frac{n}{4}) + \sqrt{n}$

6. $T(n) = 2T(\frac{n}{4}) + n$

7. $T(n) = 2T(\frac{n}{4}) + n^2$