CSE 551 Assignment 2

14^{th} September, 2022

Submission Instructions: Deadline is **11:59pm on 09/20/2022**. Late submissions will be penalized, therefore please ensure that you submit (file upload is completed) before the deadline. Additionally, you can download the submitted file to verify if the file was uploaded correctly. **Please TYPE UP YOUR SOLUTIONS and submit a PDF** electronically, via *Canvas*.
Furthermore, please note that the graders will grade 2 out of the 4 questions randomly. Therefore, if the grader decides to check questions 1 and 4,and you haven't answered question 4, you'll lose points for question 4. Hence, please answer all the questions.

1. The Fibonacci series can be computed as follows,

$$F(n) = F(n-1) + F(n-2) \tag{1}$$

In class, we showed how this can be done in $\mathcal{O}(log\ n)$ computation time. Now suppose that the definition is changed in the following way,

$$F'(n) = F'(n-1) + F'(n-2) + F'(n-3) + F'(n-4) \tag{2}$$

Can $F'(n)$ be computed in $\mathcal{O}(log\ n)$? If yes, please show how it can be done. If no, show a counterexample where this fails. Please provide your rationale for both. Assume that $F'(0) = 0, F'(1) = 1, F'(2) = 1, F'(3) = 1$. **[25 Points]**.

2. In class we have discussed the Quick Sort algorithm. Answer the following questions regarding quick sort.

   - When does the worst case for Quick Sort occur?
   - What would be the worst case recurrence for Quick Sort? Solve the recurrence and get its time complexity in worst case.
   - What would be the best case recurrence for Quick Sort? Solve the recurrence and get its time complexity in best case.

- Suppose you have an algorithm which can give the median of a set on numbers in O(n) time. How can this be used to improve the worst case time complexity of Quick sort?

- In the execution of Quick sort suppose every time the $n/5^{th}$ element is picked as the pivot element among n elements, what would be the recurrence for quick sort? What would be its time complexity?

3. If k is a non-negative constant, then prove that the recurrence: **[25 points]**

$$T(n) = k \text{ , for } n = 1 \text{ and}$$
$$T(n) = 3T(\tfrac{n}{2}) + kn, \text{ for } n > 1$$

has the following solution (for n a power of 2):

$$T(n) = 3kn^{log_2 3} - 2kn$$

4. Design an algorithm to compute the 2nd smallest number in an unordered (unsorted) sequence of numbers $\{a_1, a_2, ..., a_n\}$ in $n + \lceil log_2(n) \rceil - 2$ comparisons in the worst case. If you think such an algorithm can be designed, then show how it can be done. If your answer is no, then explain why it cannot be done. **[25 points]**