Problem Set 1

2021-2022, 1st Semester **Due:** 20:00, 10-14

Please answer each of the following problems. You can discuss the problems with your classmates, but do't just simply copy and paste. Don't plagiarize! （可以使用中文回答问题。）

flawed in at least one place. Which of their steps are faulty and why?

1. **Different-sized sub-problems.** (10 points)

   Solve the following recurrence relation.

   $$T(n) = T(n/2) + 2T(n/8) + n,$$

   where $T(1) = 1$.

   [We are expecting a formal proof with substitute method. For example, what is the value range of the parameters in your assumption? You may state your final running time with $O(\cdot)$ notation, but do not use it in your proof. Pay attention to $T(1) = 1$, which should be satisfied in your results.]

2. **Asymptotic bound with Maximum operator.** (15 points)

   Let $f(n)$ and $g(n)$ be asymptotically non-negative functions. Using the basic definition of the Θ notation, prove that $max(f(n), g(n)) = \Theta(f(n) + g(n))$.

3. **Proof of correctness.** (15 points)

   Consider the following Selection Sort algorithm that is supposed to sort an array of integers. Provide a proof that this algorithm is correct. (**Hint**: you may need to use more than one loop invariant.)

```
# Sorts an array of integers.
Sort(array A):
    for i = 1 to A.length:
        minIndex = i
        for j = i + 1 to A.length:
            if A[j] < A[minIndex]:
                minIndex = j
                Swap(A[i], A[minIndex])

# Swaps two elements of the array.  You may assume this function is correct.
Swap(array A, int x, int y):
    tmp = A[x]
    A[x] = A[y]
    A[y] = tmp
```

4. **COVID-19 Risk Detection.** (20 points)

Each of n customers spends some time in a cafe shop. For each $i = 1, \ldots, n$, user i enters the shop at time a_i and leaves at time $b_i \geq a_i$. You are interested in the question: how many distinct pairs of customers are ever in the shop at the same time? (Here, the pair (i, j) is the same as the pair (j, i)).

Example: Suppose there are 5 customers with the following entering and leaving times:

Customer	Enter time	Leave time
1	1	4
2	2	5
3	7	8
4	9	10
5	6	10

Then, the number of distinct pairs of customers who are in the shop at the same time is three: these pairs are $(1, 2)$, $(4, 5)$, $(3, 5)$.

(a) (5 pts) Suppose a customer c_i is reported as a suspected case, can you give an algorithm to retrieve all the potential inflicted customers in $O(n)$-time?

(b) (7 pts) Given input $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$ as above, there is a straightforward algorithm that takes about[1] n^2 time to compute the number of pairs of customers who are ever in the shop at the same time. Give this algorithm and explain why it takes time about n^2.

(c) (8 pts) Give an $O(n \log(n))$-time algorithm to do the same task and analyze its running time. (**Hint:** consider sorting relevant events by time).

5. **SELECT-7 algorithm.** (20 pts, 3 pts for each of (a), (b), (d), (e); 4 pts for each of (c) and (f))

In the `SELECT` algorithm from class, in order to find a pivot, we break up our array into blocks of length 5. Why 5? In this question, we explore `SELECT-7`, in which we break up our array into blocks of length 7. In addition, to simplify your logic, you should assume throughout this problem that your array is a power of 7.

(a) Consider the pseudocode for `SELECT` and `MEDIAN_OF_MEDIANS` in ours slides of lecture 3. In the pseudocode, we break up our array into blocks of size 5. What change(s) would you need to make to this pseudocode in order to write `SELECT-7` and `MEDIAN_OF_MEDIANS-7`? [**We are expecting a one or two sentence description of changes made.**]

(b) Prove that the recursive call to `SELECT-7` inside of `MEDIAN_OF_MEDIANS-7` is on an array of size at most $n/7$. You should assume that your array is a power of 7. [**We are expecting a rigorous proof.**]

(c) Prove that the recursive call inside of `SELECT-7` is on an array of size at most $5n/7 + 7$. You should again assume that your array is a power of 7. (Hint: it might be helpful to note that $\lceil x \rceil \leq x + 1$.) [**We are expecting a rigorous proof.**]

[1] Formally, "about" here means $\Theta(n^2)$, but you can be informal about this.

(d) Explain why the work done within a single call to `SELECT-7` and `MEDIAN_OF_MEDIANS-7` on an array of size n is $\Theta(n)$. **[We are expecting a few sentences of explanation. You do *not* need to do a formal proof with the definition of Θ.]**

(e) Using what you proved in (b), (c), and (d), write down a recurrence relation for the runtime of `SELECT-7`. (You can do this problem even if you did not complete all of (b), (c), and (d).) **[We are expecting a one-line answer with your recurrence relation.]**

(f) Is `SELECT-7` $O(n)$? Justify your answer. **[We are not expecting a formal proof, but you should describe clear reasoning, such as analyzing a tree, unraveling the recurrence relation to get a summation, or attempting the substitution method. (Note that succeeding at the substitution method would prove select-7 is O(n), but failing at the substitution method does not prove select-7 is not O(n). 代入法成功可以证明 select-7 的复杂度是 O(n) 的，但是代入法不成功不能说明 select-3 的复杂度不是 O(n)。)]**

6. **Find out honest monkeys.** (20 pts, 5 pts for each)

On a mountain, there are honest monkeys and unreliable monkeys. The honest monkeys always tell the truth; the unreliable monkeys may lie or may tell the truth. The monkeys themselves can tell who is honest and who is unreliable, but an outsider can't tell the difference: they all just look like monkeys.

You arrive at this mountain, and are asked to find out honest monkeys. You can pair up the monkeys and ask them evaluate each other. We refer to one of these evaluations as a "monkey-to-monkey comparison". The outcomes of comparing monkeys A and B are as follows:

Monkey A	Monkey B	A says about B	B says about A
honest	honest	honest	honest
honest	unreliable	unreliable	either
unreliable	honest	either	unreliable
unreliable	unreliable	either	either

Suppose that there are n monkeys on the mountain, and that there are strictly more than $n/2$ honest monkeys.

(a) Suppose that n is even. Show that, using **at most** $n/2$ **monkey-to-monkey comparisons**, you can reduce the problem to the same problem with less than half the size. That is, give a procedure that does the following:

- **Input**: A population of n monkeys, where n is even, so that there are strictly more than $n/2$ honest monkeys in the population.
- **Output**: A population of m monkeys, for $0 < m \leq n/2$, so that there are strictly more than $m/2$ honest monkeys in the population.

[We are expecting a pseudocode.]

(b) Using the procedure from part (a), design a recursive algorithm that uses $O(n)$ monkey-to-monkey comparisons to find a single honest monkey. **[We are expecting a pseudocode. Here we assume that the procedure in (a) can work when n is either even or odd]**

(c) Prove formally, using induction, that your answer to part (b) is correct. **[We are expecting: A formal argument by induction. Make sure you explicitly state the inductive hypothesis, base case, inductive step, and conclusion.]**

(d) Give a procedure to find out all honest monkeys using $O(n)$ monkey-to-monkey comparisons. **[We are expecting a pseudocode or an informal description of the procedure.]**