CS218, Spring 2023, Assignment #1 **Due: 11:59pm, Thursday 04/13, 2023**
Name: Zijin Wan **Student ID:**

1 A Complex Complexity Problem (1.2pts)

Answers:

1. $(\sqrt{3})^{\log n} = o(n)$
   Explain briefly:
   $$(\sqrt{3})^{\log n} = n^{\log \sqrt{3}} = o(n)$$
2. $\log \log n = o(\sqrt{\log n})$
   Explain briefly:
   $$\log \log n = \log((\sqrt{\log n})^2) = 2 \log \sqrt{\log n} = o(\sqrt{\log n})$$
3. $\log(n!) = \Theta(n \log n)$
   Explain briefly:
   1.
   $$\log(n!) = \log(n * (n-1) * ... * 1) = \log n + \log(n-1) + ... + \log 1 = O(n \log n)$$

   2.

   $\log(n!) = \log n + \log(n-1) + ... + \log 1 \geq \log n + \log(n-1) + \log(n/2) \geq (n/2) log(n/2) = \Omega(n \log n)$

   Thus, $\log(n!) = \Theta(n \log n)$
4. $2^n = o(3^n)$
   Explain briefly:
   $$\lim_{n \to \infty} \frac{3^n}{2^n} = \lim_{n \to \infty} 1.5^n = \infty$$

2 Solve Recurrences (0.6pts)

1. $T(n) = T(n/2) + n \log n$
   Answer: Let $y = \log_b a = \log_2 1 = 0$, since $f(n) = n \log n = \Omega(n^y) = \Omega(1)$, $T(n) = \Theta(n \log n)$.
2. $T(n) = 2T(n/4) + \sqrt{n}$
   Answer: Let $y = \log_b a = \log_4 2 = 0.5$, since $f(n) = \sqrt{n} = \Theta(n^y)$, $T(n) = \Theta(\sqrt{n} \log n)$.
3. $T(n) = 4T(n/4) + n^{1/2}$
   Answer: Let $y = \log_b a = \log_4 4 = 1$, since $n^{1/2} = O(n^y) = O(n)$, $T(n) = \Theta(n)$.

3 Test the candies

1. (0.3pts) Your boss told you that there is only **one bad candy**. In that case, can you show an algorithm that uses $\lceil \log_2 n \rceil$ (note: this is not in big-O!) dollars to find this bad candy?
   Answer: To find the bad candy among a group of candies, the candies can be evenly divided into two piles and placed on one side of the scale. The pile with the smaller weight contains the bad candy. This process can be repeated on this pile until the bad candy is found. If the number of candies is odd, one candy can be arbitrarily selected and set aside. If the scale is balanced, the selected candy is the bad candy. Otherwise, the divide-and-conquer process can be continued. In either case, this algorithm requires at most $\lceil \log_2 n \rceil$ steps.

1. First, evenly divide the candies into two piles and put them on the scale separately. After that, we know the pile with less weight contains the bad candy. Then we unload the scale and repeat the first step with that pile. As in each round this algorithm reduces the candidate candies to a half, it will finish after $\lceil \log_2 n \rceil$ rounds while each round costs one dollar.
   Specifically, if in some rounds, the number of candies is odd, we arbitrary put one candy aside. If the scale is balanced, this candy is the bad one; otherwise, we continue doing the algorithm. In either case, the bound $\lceil \log_2 n \rceil$ still holds.

2. First, similar to the previous algorithm, we equally divide the candies but into three piles. We select two of them put on the scale and see if one has less weight. In this case, we repeat the first step with the less weighted pile since it contains the bad candy. Otherwise, we repeat the first step with the unselected pile. Since this algorithm reduces the candidates down to one third in each round, the cost is $\lceil \log_3 n \rceil$ therefore.
   In some rounds, if the amount of candies divided by three is one, we put this one to the unselected pile. If that divided by three is two, we put one of the remained candies on each side of the scale.

3. In the decision tree, there are n leaves and any of them could possibly be the bad one. Since a balance scale can generate at most three result in each use (left, right, even), there are three decisions made at most each time. With three branches on each node, the decision tree has the worst depth of at least $\lceil \log_3 n \rceil$, which is the lower bound of the algorithm. We cannot use fewer dollars to guarantee to find the correct answer.

4. We can divide the candies into two piles and repeat the step with the less weighted one until the two piles have the same weight. This operation costs at most $\Theta(\log_2 n)$ dollars. Once the piles get balanced, we know that each of them have one bad candy and they can be solved within extra $2 \log_2 n$ dollars via the algorithm in (1). Therefore, this algorithm can be done using $O(\log n)$ dollars.

5. First, we divide the candies into two piles and the lighter one contains at least $\lceil k/2 \rceil$ bad candies. In this way, we repeat the first step with the lighter pile (if the two are balance, use arbitrary one). In the worst case, after at most $\lceil \log k \rceil$ rounds, there must be a pile containing exact one bad candy and we can find it in $O(\log n)$. That means we could distinguish one bad candy in $O(\log n)$.
   Then, we remove this bad one and repeat all the steps above until all bad candies are found. It can be done in $k \cdot O(\log n)$. As k is a constant, $k \cdot O(\log n) = O(k \log n) = O(\log n)$. Alternatively, we can split the candies into $k + 1$ piles, so the heaviest one has no bad candies. We can work on the piles with bad candies and use the all-good pile as a reference. This can also give you a desired solution.

6. Similar to the part (4), we use the decision tree to prove the lower bound. In each round, the same, we divide the candies into three piles and can make at most three decisions. Since the decision tree has $2n$ leaves now (n candies and each can be either lighter or heavier) while each node has up to three branches, the worst depth is at least $\lceil \log_3 2n \rceil$.

7. After using the scale the first time, we split the candies into three piles, denoted as A, B, and C. Assuming we have put A and B on the scale, in the case that the scale was balanced, we know the bad one is in C but yet we have no weight relationships between candies in it. Hence, we need $\lceil \log_3(2|C|) \rceil$ dollars to find that one so $|C| \leq 4$ as we only have two-dollar budget left.
   In the other hand, if the scale was unbalanced in the first time, there are $|A| + |B|$ possibilities because either candies in A may be lighter or the ones in B may be heavier. Using the decision tree, we know at least $\lceil \log_3(|A| + |B|) \rceil$ dollars are needed to figure it out so $|A| + |B| \leq 9$.
   Since $|A| + |B| + |C| = 13$, the only valid assignment is $|A| + |B| = 9$ and $|C| = 4$.
   To compare the weights of candies on the scale, it needs $|A| = |B|$ so we add the reference candy to make their total size even. We label the unknown candies in A from 1 to 5, and those in B from 6 to 9. The reference is put in B.

Case 1: A and B are unbalanced. Without loss of generality, we assume A is lighter. We list the 9 possibilities in the table below and show how to figure it out by two more uses. In the second use, we put candies labeled 136 and 257 on the two sides of the scale, respectively. While in the third use, we put on 238 and 159. We use the symbol $\circ$ to represent the one lighter and use $\bullet$ to represent the one heavier. According to the comparison results, the bad candy is decided by looking up the table.

	2nd use		3rd use	
possibilities	136	257	238	159
1 $\circ$	$<$		$>$	
2 $\circ$	$>$		$<$	
3 $\circ$	$<$		$<$	
4 $\circ$	$=$		$=$	
5 $\circ$	$>$		$>$	
6 $\bullet$	$>$		$=$	
7 $\bullet$	$<$		$=$	
8 $\bullet$	$=$		$>$	
9 $\bullet$	$=$		$<$	

Case 2: A and B are balanced so the bad one is in C. We label the unknown candies in C from 1 to 4 and denote the reference as R. In the second round, we put 12 and 3R on the two sides. If they are unbalanced, we compare 1 and 2 in the last use; otherwise, we compare 4 and R. The bad candy can be decided by the table below.

	2nd use		3rd use		3rd use	
possibilities	12	3R	1	2	4	R
1 $\circ$	$<$		$<$			
2 $\circ$	$<$		$>$			
3 $\circ$	$>$		$=$			
4 $\circ$	$=$				$<$	
1 $\bullet$	$>$		$>$			
2 $\bullet$	$>$		$<$			
3 $\bullet$	$<$		$=$			
4 $\bullet$	$=$				$>$	