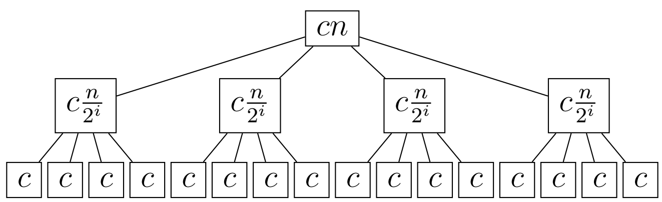
Problem Set 2

Name: Your Name

Collaborators: Name1, Name2

Problem 2-1.

(a) $T(n) = 4T(\frac{n}{2}) + O(n)$



$T(n) = \sum_{i=0}^{logn} 4^i \frac{n}{2^i} = n \sum_{i=0}^{logn} 2^i = n(2n-1) = \Theta(n^2)$
In another hand, there are $4^{log_2 n} = n^2$ leaves. And $T(1) = \Theta(1)$, we induct that $T(n) = \Omega(n^2)$. As consequence, $T(n) = \Theta(n^2)$

(b)

$$T(n) = 3T(\frac{n}{\sqrt{2}}) + O(n^4) = \sum_{i=0}^{\log_{\sqrt{2}} n} 3^i (\frac{n^4}{4^i}) \tag{1}$$

$$= 4n^4(1 - (\frac{3}{4})^{log_{\sqrt{2}} n+1}) = O(n^4) \tag{2}$$

(c)

$$T(n) = 2T(\frac{n}{2}) + 5nlogn = \sum_{i=0}^{logn} 2^i \frac{5nlogn}{2^i} \tag{3}$$

$$= 5nlogn(logn + 1) = O(nlog^2 n) \tag{4}$$

Problem 2-2.

(a) The problem requires the algorithm to be in-place, which excludes the merge sort. $D.set\_at(i, x)$ costs $\Theta(nlogn)$ and each swap operation calls $D.set\_at(i, x)$ twice, which is definitely inefficient. So we want to choose **the algorithm which needs fewer swaps**. And the answer for that **is selection sort**, which needs $\Theta(n)$ swaps at worst. In this case, $T(n) = O(n(n + nlogn)) = O(n^2 logn)$. In contrast, insertion sort will perform $\Theta(n^2)$ times swaps in worst case, which leads $T(n) = O(n^3 logn)$

(b) In this case, comparison is an expensive operation. As a result, we want to choose **the algorithm which needs fewer comparisons**. And the answer for that is **merge sort**. For selection sort and insertion sort, they need n^2 times comparisons. For merge sort, it needs $nlog_2 n$ times comparisons.

(c) In this case, the array is basiclly sorted because even for $n = 10^9, loglogn \simeq 5$. In addition, swaps are adjacent. So the answer is absolutely insertion sort. In this case, $T(n) = \Theta(n)$, which is linear time.

Problem 2-3. Pass

Problem 2-4.

Problem 2-5.

(a)

(b)

(c) Submit your implementation to `alg.mit.edu`.