CS375 Assignment 8

Due October 18th 10PM ET

A general note for CS375: When writing up your homework, please write neatly and explain your answers clearly, giving all details needed to make your answers easy to understand. Graders may not award full credit to incomplete or illegible solutions.

You are encouraged to discuss the problems with your classmates, **but your submission should be your own.** When writing your solutions up, you should not be looking at any other student's submission.

Clear communication **is the point**, on every assignment. In general in CS375, unless explicitly specified otherwise, answers should be accompanied by explanations. Answers without explanations may not receive full credit. Please feel free to ask me any questions about explanations that might come up!

Necessary acknowledgements: many of these questions are reproductions or adaptations of questions of Eric Aaron's devising. Others are from our course texts as listed on the syllabus.

1. In class, we developed a linear-time algorithm for SELECTION: given an array A of n distinct numbers and an integer k, find the kth-smallest number in A. We first considered groups of three, which led to a recurrence with an $n \log n$ solution, and then switched to groups of five. This problem walks through why groups of five give us a linear-time algorithm.

   Our algorithm is the following. When we recursively select a median, use the lower median if a set has even size.

```
def Selection(A[1, ..., n], k):
    """Return the kth-smallest element of A."""
    if n <= 5:
        sort A and return A[k]

    Divide A into groups of 5
    M = the array containing the median of each group
    pivot = Selection(M, len(M) / 2)  # the median of the group medians

    r = Partition(A, the index of pivot)
    # Partition puts the pivot in position r, with every
    # smaller element before it and every larger element after it.

    if k == r:
        return A[r]
    elif k < r:
        return Selection(A[1, ..., r-1], k)
    else: # if k > r
        return Selection(A[r+1, ..., n], k-r)
```

   (a) Explain why the base case, where $n \leq 5$, takes $\Theta(1)$ time and correctly returns the kth-smallest element.

   (b) We divide the array into groups of five and find the median of each group. Explain why finding all these medians takes $\Theta(n)$ time.

   (c) There are $n/5$ group medians. We then find their median recursively by a call to SELECTION on an input of size $n/5$. What is the purpose of choosing the median of the group medians, rather than simply choosing one of the group medians arbitrarily?

   (d) We now need to show that the pivot allows us to discard a substantial fraction of the array. Consider the groups whose medians are less than or equal to the pivot. At least

half of the group medians are less than or equal to the pivot. Explain why, in each such group at least three elements are less than or equal to the pivot. What does this tell us about the number of elements less than or equal to the pivot?

(e) Use a symmetric argument to show that at least approximately $3n/10$ elements are greater than or equal to the pivot as well. Explain why, after partitioning, the recursive call that continues searching for the answer has size at most $7n/10$ (ignoring constant-size rounding effects).

(f) Use induction to show that the algorithm correctly returns the kth-smallest element. Your explanation should discuss the three possibilities after partitioning: the pivot is in position k, the desired element is before the pivot, or the desired element is after the pivot.

2. We will now analyze the runtime. Let $T(n)$ be the runtime of SELECTION on an array of size n.

(a) Using your answers above, explain why the runtime satisfies

$$T(n) \leq T(n/5) + T(7n/10) + \Theta(n)$$

(b) Draw the recursion tree for this instance. Explain why the work at level i is $\Theta\left(\left(\frac{9}{10}\right)^i n\right)$.

(c) Explain why the height of the recursion tree is $\Theta(\log n)$.

(d) Use the geometric-series identity (which you proved using induction wayyyy back when)

$$\sum_{i=0}^{h} r^i = \frac{1 - r^{h+1}}{1 - r}$$

to show that the total nonrecursive work over all levels is at most $\Theta(n)$.

(e) Why does this establish an $O(n)$ upper bound for $T(n)$, including the work done at the leaves?

(f) Finally, explain why $T(n) = \Omega(n)$ (thereby showing that $T(n) = \Theta(n)$).