This week, we’ll continue exploring the use of randomization and approximation to solve problems.

1. **Finding the Median.** Given a set \( S = \{a_1, \ldots, a_n\} \) of numbers, the median of \( S \), denoted \( \text{med}(S) \), is the \( k \)-th smallest element of \( S \), where \( k = \lfloor \frac{n+1}{2} \rfloor \). In this problem, you will analyze a randomized algorithm to output the median. Consider the following algorithm for finding the median:

\[
\text{FindMedian}(S)
1 \quad \text{Return Select}(S, \lfloor \frac{n+1}{2} \rfloor)
\]

\[
\text{Select}(S, k)
1 \quad \text{Choose pivot } a_i \in S \text{ at random, uniformly}\,
2 \quad \text{Initialize } S^-, S^+ := \{
3 \quad \text{for each } j \neq i
4 \quad \text{if } a_j < a_i \text{ add } a_j \text{ to } S^-
5 \quad \text{if } a_j > a_i \text{ add } a_j \text{ to } S^+
6 \quad \text{if } |S^-| = k - 1 \text{ return } a_i
7 \quad \text{else if } |S^-| > k - 1
8 \quad \quad \text{return Select}(S^-, k)
9 \quad \text{else}
10 \quad \quad \text{return Select}(S^+, k - (1 + |S^-|))
\]

---

An element is chosen *uniformly* if each element is equally likely to be picked.

- First, show that \( \text{FindMedian} \) always returns the median.
- Next, analyze the \( \text{FindMedian} \) running time when the pivot element is chosen uniformly from \( S \). The following structure will help guide you.

Say that the algorithm is in **phase** \( j \) if there are between \( n(3/4)^j \) and \( n(3/4)^{j+1} \) elements in the set \( S \) being considered. So, for example, we are in phase 0 the first time \( \text{Select} \) is called (because \( n(3/4)^0 + 1 \leq |S| \leq n(3/4)^0 \)).

Call an element \( a_i \in S \) **central** to \( S \) if (i) at least \( |S|/4 \) of the elements of \( S \) are less than \( a_i \) and (ii) at least \( |S|/4 \) elements of \( S \) are greater than \( a_i \).

(a) Show that there are \( |S|/2 \) central elements.
(b) Show that if the pivot element is central, the phase ends i.e., the next recursive call that gets made will be in a different phase.
(c) Give an upper bound on the expected number of recursive calls to \( \text{Select} \) before a round ends.
(d) Give an upper bound on the running time of each \( \text{Select} \) call, not including recursive calls.
(e) Give an upper bound on the number of phases that are run before \( \text{FindMedian} \) terminates.
(f) Give an upper bound on the expected runtime of \textsc{FindMedian} when the pivot is chosen uniformly.

2. **Chromatic Number.** Consider the optimization problem \textsc{ChromaticNumber}, defined as follows. Given a graph \( G = (V, E) \) as input, determine the smallest number \( k \) such that it is possible to \( k \)-color the graph.

(a) Prove that \textsc{ChromaticNumber} is NP-hard.
(b) Prove that there is no efficient \( \frac{4}{3} \)-approximation to \textsc{ChromaticNumber} unless \( P = \text{NP} \).