1. Let $0 < \alpha \leq 1/2$. Suppose Quicksort takes a path such that each segment of length $m$ removes a pivot and produces two segments of length $\alpha m - 1$ and $(1 - \alpha)m$. Considering the top level of the recursion tree to be level 0, find an expression for $k$ = deepest level at which a leaf (segment with one element) may appear. Ignore round-ups and round-downs by assuming that segments produced by the $\alpha$-expressions are integers.

2. Show that the instruction count along such a path is $O(n \lg n)$. That is, consider an input permutation that causes the subproblem size evolution in the preceding question, and show that the time complexity is $O(n \lg n)$ for that permutation.

3. Show that the instruction count along such a path is $\Theta(n \lg n)$. That is, consider an input permutation that causes the subproblem size evolution in the preceding question, and show that the time complexity is $\Omega(n \lg n)$ for that permutation.
Solutions:

1. Let $0 < \alpha \leq 1/2$. Suppose Quicksort takes a path such that each segment of length $m$ removes a pivot and produces two segments of length $\alpha m - 1$ and $(1 - \alpha)m$. Considering the top level of the recursion tree to be level 0, find an expression for $k = \text{deepest level at which a leaf (segment with one element) may appear}$. Ignore round-ups and round-downs by assuming that segments produced by the $\alpha$-expressions are integers.

The deepest level leaf will occur at the end of a path that passes through the larger of the two child segments at each step. Assuming an input of size $n$, the root at level 0 splits and the path takes the larger child, which has size $(1 - \alpha)n$. The path continues into a segment of size $(1 - \alpha)^2n$, and so forth. It terminates, at level $k$ when the segment has a single element. That is,

$$1 \leq (1 - \alpha)^k < 2$$

$$\frac{1}{n} \leq (1 - \alpha)^k < \frac{2}{n}$$

$$\frac{\lg \left( \frac{1}{n} \right)}{\lg(1/(1-\alpha))} \leq k = \frac{2}{\lg(1/(1-\alpha))} \leq \frac{1}{\lg(1/(1-\alpha))} \lg n.$$

2. Show that the instruction count along such a path is $O(n \lg n)$. That is, consider an input permutation that causes the subproblem size evolution in the preceding question, and show that the time complexity is $O(n \lg n)$ for that permutation.

We know that the instructions in the Lomuto partition, plus its invocation and the two recursive calls, are bounded above by $K_2n$, where $n$ is the length of the segment to be sorted. At the top level, we have an upper bound of $K_2n$ on the instruction count, including the recursive calls but not the work done in those calls. At the next level, the upper bound is $K_2(n - 1)$, since one pivot element was removed at the root level. The next level has a bound of $K_2(n - 3)$, since three pivots have been removed above it. In general, the work at a given level is bounded by $K_2$ times the total length of the segments at that level. This total length, in turn, is the starting length $n$ minus the number of pivots removed in previous levels. As the shorter subtrees terminate, we cannot continue with the maximum number of removed pivots, so we denote the actual number of pivots removed prior to level $d$ as $n_d$. We then have

$$T(n) \leq K_2[n + (n - n_1) + (n - n_2) + \ldots (n - n_{k-1})] + n.$$

Here $k$ is the level where the lowest leaf appears, which implies that the last partitioning work occurs at level $k - 1$. The final $n$ in the expression accounts for processing the leaves, each of which incurs one instruction – the if-test in Quicksort that enables the recursion (or not). Substituting an upper bound for $k$ from the preceding problem,

$$T(n) \leq K_2[kn - \sum_{j=1}^{k-1} n_j] + n \leq K_2 kn + n$$

$$\leq \frac{K_2n \lg n}{\lg(1/(1-\alpha))} + n$$

$$\frac{T(n)}{n \lg n} \leq \frac{K_2}{\lg(1/(1-\alpha))} \left[ 1 + \frac{\lg((1/(1-\alpha)))}{K_2 \lg n} \right] \rightarrow \frac{K_2}{\lg(1/(1-\alpha))}.$$
3. Show that the instruction count along such a path is $\Theta(n \lg n)$. That is, consider an input permutation that causes the subproblem size evolution in the preceding question, and show that the time complexity is $\Omega(n \lg n)$ for that permutation.

We know that the best case instruction count is $\Omega(n \lg n)$. Since this particular path cannot achieve a smaller count than the best possible, we conclude that it is also $\Omega(n \lg n)$. Combining this observation with the result of the preceding problem, we have $T(n) = \Theta(n \lg n)$ for a split that induces the imbalance $\alpha m - 1, (1 - \alpha)m$ at each subdivision.