EXERCISE FOR CSE202 – WEEK 6

This exercise considers the case when many keys are actually duplicates (which is a common situation, for instance when sorting an array of people by their age).

Question 1. *Show that if, in the partitioning procedure, one of the* `A[i]>=p` *or* `A[j]<=p` *was replaced by a* `>` *or* `<` *test, then quicksort would have quadratic complexity for all arrays with just a constant number of distinct keys.*

Solution : Assume first that all keys are equal. Then, if either of the large inequalities is replaced by a strict one, the corresponding loop runs till the other extremity of the subarray and the pivot is put at the corresponding end. Thus the recursive call will enter with a subarray of length $n-1$, which is the worst case for quicksort.

When there are k distinct keys, at least one of them occurs at least n/k times. At one point during the recursive calls, an array of size at least $n/(2k)$ containing only copies of this key will be given as input and the previous analysis shows a complexity in $O(n^2/(4k^2)) = O(n^2)$ comparisons in that case, since k is considered fixed. □

Question 2. *Assuming that 3-way partitioning can be done in $n-1$ comparisons of keys, show that on an array where the keys can only take two distinct values, the number of comparisons of keys performed by quicksort becomes* linear *in n.*

Solution : If the array contains only two distinct types of keys, then after 3-way partitioning, it is sorted! □

Question 3. *How is that compatible with the $n\log_2 n$ lower bound for sorting?*

Solution : The lower bound is the worst-case over all $n!$ possible permutations of n elements. It is obtained as the height of a perfectly balanced tree with that number of leaves. Here, we are considering the worst-case over a much more restricted set of inputs, and given more prior information (the distinct number of keys). So the complexity can be better than the general worst-case.
Note. In order to estimate a lower bound in this restricted case, the tree that has to be constructed does not need to distinguish all permutations anymore, since a transposition of two duplicate keys has no impact on the final ordering. So with two distinct possible keys, we have 2^n possible inputs. Another, more minor difference, is that we have to consider a ternary tree since the outcome of a comparison is now one of $<,=,>$. In any case, we thus obtain that at most n comparisons are needed and this is consistent with the complexity achieved by quicksort.

□