For this problem, the running-time of quicksort is measured by the number of comparisons between array entries. Since for the input array, it has all distinct entries and is sorted in decreasing order, this means that each time the Partition algorithm is ran, it must produce one subproblem with n – 1 elements and one with 0 elements. And this means that every time the algorithm Partition(A, p, r) is called, it would perform (n – 1) comparisons within the for loop, where n = r – p + 1. Since Partition is called within Quicksort, Quicksort will keep calling itself as long as p < r, which means there exists more than one element in the array that need to be sorted. Although, within each Quicksort recursive calls, it would always perform one comparison in the beginning, but this comparison is not between array entries but is between indexes. Hence, for this algorithm, the lower bound for the number of comparisons between array entries is
(n – 1) + ((n – 1) – 1) + ((n – 1 – 1) – 1) + … + 2 + 1
= (n – 1) + (n – 2) + (n – 3) + … + 2 + 1
= $\frac{n\,(n-1)}{2} = \frac{n^2-n}{2}$ as the algorithm must perform this many comparisons in order to finish. Hence, the running-time of Quicksort as measured by the number of comparisons between array entries is $\Omega(n^2)$.