10665006

Insertion sort algorithm

SUMMARY INSERTION SORT

To determine the average efficiency of insertion sort consider the number of times that the inner loop iterates. As with other loops featuring nested loops, the number of iterations follows a familiar pattern: 1 + 2 + ... + (*n* - 2) + (*n* - 1) = *n*(*n* - 1) = *O*(*n*2). Conceptually, the above pattern is caused by the sorted sublist that is built throughout the insertion sort algorithm. It takes one iteration to build a sorted sublist of length 1, 2 iterations to build a sorted sublist of length two and finally n-1 iterations to build the final list. To determine whether there are any best or worst cases for the sort, we can examine the algorithm to find data sets that would behave differently from the average case with random data. Because the average case identified above locally sorts each sublist there is no arrangement of the aggregate data set that is significantly worse for insertion sort. The nature of the sorting algorithm does however lend itself to perform more efficiently on certain data. In the case where the data is already sorted, insertion sort won't have to do any shifting because the local sublist will already be sorted. That is, the first element will already be sorted, the first two will already be sorted, the first three, and so on. In this case, insertion sort will iterate once through the list, and, finding no elements out of order, will not shift any of the data around. The best case for insertion sort is on a sorted list where it runs is *O*(*n*).

The insertion sort algorithm is the sort unknowingly used by most card players when sorting the cards in their hands. When holding a hand of cards, players will often scan their cards from left to right, looking for the first card that is out of place. For example, if the first three cards of a player's hand are 4, 5, 2, he will often be satisfied that the 4 and the 5 are in order relative to each other, but, upon getting to the 2, desires to place it before the 4 and the 5. In that case, the player typically removes the 2 from the list, shifts the 4 and the 5 one spot to the right, and then places the 2 into the first slot on the left. This is insertion sort. Unlike other simple sorts like selection sort and bubble sort which rely primarily on comparing and swapping, the insertion sort achieves a sorted data set by identifying an element that out of order relative to the elements around it, removing it from the list, shifting elements up one place and then placing the removed element in its correct location.