**ANALYSIS OF SORTING ALGORITHMS**

**Insertion Sort:**

**Best Case: Best case occurs in insertion sort when the list is in ascending order.**

**In that case the outer loop for(int i =1;i<input.length;i++) will execute for n-1 one times and inner loop will execute only once per outer loop.**

**Hence the position will keep on just incrementing and no swapping will be required.**

**Hence,**

**Position =2 comparisons =1**

**Position =3 comparisons =2**

**Therefore sum of comparisons = n-1**

**Hence the time complexity will be O(n).**

**Worst Case:**

**Worst case occurs when the list is in descending order, as the outer loop will run for N-1 times the inner while loop to will run till n-1 times**

**Hence the sum of comparisons and swapping will be AP i.e the time complexity will be O(n^2)**

**Quick Sort:**

**Best Case:**

**Time Complexity for best case will be O(nlogn).**

**Let’s assume we have n elements and we consider a pivot to be left + right/2 than there will be n/2 partitions.**

**As quick is recursive algorithm it will further divide the partitions till the size 1.**

**So sum of total comparisons made will be n\* log n(to base 2) ie O(nlogn).**

**Worst Case:**

**It will occur when in the partition we have one result on left or right side and whole list needs to be partition again, this will generally happen when pivot is either the largest or smallest in the list.**

**So in the worst case we will have**

**n-1 – First pass**

**n-2- Second pass**

**n-3 in 3rd pass.**

**So when we take sum of all the comparisons the complexity will be O(N^2).**

**Selection Sort.**

**For selection sort whether its best case, average case or worst case the complexity will always remain O(n^2).**