$= n + (n+1) + (n+2) + \ldots\ldots 2$

$= \frac{n(n+1)}{2} - 1 = \frac{n^2+n}{2} - 1 = \frac{n^2+n-2}{2}$

$= O(n^2)$, ignoring the lower order terms

Hence the time complexity of selection sort algorithm is $O(n^2)$

3.5 analysis of sorting algorithm - Insertion Sorting

In the Insertion sort algorithm, the second data element is compared with the first. If the second element is lesser than the first, then it is inserted before the first data element. Next the third data element is compared with the second one. If it is lesser than the second element, then it is compared first. If it is lesser than the first element then it is inserted before the first data element, else it is inserted between the first and the second data elements.

This process is repeated for all the n-1 elements. After the n-1 iterations, all the data elements get sorted. The algorithm is shown in Figure 13

```
BEGIN
for i ← 2 to length(Arr)
 j ← i-1 v=A[i]
 while j >= 1 and v <= A[j]
     A[j+1] = A[j] , j ← j - 1
 end while
 a[j+1] = v
end for
```

FIGURE 5 : INSERTION SORT

Let's now determine the time complexity of the insertion sort algorithm.
The basic operations done inside the inner loop are one comparison and one swapping. One swapping operation done inside the while loop will get executed = (i – 1-1) + 1 = i-1 times (because j varies from i-1 to 1). Hence, the number of comparisons in the while loop is 2*(i-1)

The while loop performs 2 comparisons and one swapping = 3*(i-1)

‹ qsort() Library Function Demo (/?q=MULNCourseBook/listDetailedCBLearningContents/178099/cidb/full/view/153784/431550/431535/39731)