**Name: Agboado Gideon Azasu**

**ID: 10681572**

**Insertion Sort Algorithm**

Insertion sort is a type of sorting algorithm that produces a final sorted array one item at a time from an unsorted array. Insertion sort iterates, taking one input element each repetition, and producing a sorted array list at the end. In each iteration, insertion sort takes one element from the elements in the array, finds the position where it belongs within the sorted list, and then inserts the element there. This process is repeated until all the array list has been sorted.

**Pseudocode For Insertion Algorithm**

rocedure insertionSort( A : array of items )

int holePosition

int valueToInser

for i = 1 to length(A) inclusive do:

valueToInsert = A[i]

holePosition = i

while holePosition > 0 and A[holePosition-1] > valueToInsert do:

A[holePosition] = A[holePosition-1]

holePosition = holePosition -1

end while

A[holePosition] = valueToInsert

end for

end procedure

**Time Complexity Of Insertion Sort**

Examining the insertion sort code, it is clear that every iteration of while loop reduces one inversion. The while loop executes only if i > j and arr[i] < arr[j]. So we can say the number of inversions is same as the total number of while loop iterations for all the values of i . Therefore overall time complexity of the insertion sort is O(n + f(n)) where f(n) is inversion count. The time complexity of insertion sort is O(n), If the inversion count is O(n). In worst case scenario, there can be n\*(n-1)/2 inversions. The worst case occurs as aresult of the array being sorted in a reverse order. So therefore, the worst case time complexity of insertion sort is O(n2).