**DAA ASSSIGNMENT**

**ID - 10663651**

**Insertion Sort**

Insertion sort is a simple sorting algorithm that builds the final sorted array one item at a time and keeps a prefix of the array sorted. At each step, this prefix is grown by inserting the next value into it at the correct place. Eventually, the prefix is

the entire array, which is therefore sorted. Every repetition of insertion sort removes an element from the input data, inserting it into the correct position in the already-sorted list, until no input elements remain. The choice of which element to remove from the input is arbitrary, and can be made using almost any choice

algorithm.

Sorting is typically done in-place. The resulting array after k iterations has the property where the first k + 1 entries are sorted. In each iteration the first remaining entry of the input is removed, inserted into the result at the correct position, thus extending the result.

**code for insertion sort:**

void InsertionSort(int a[], int n)

{

for(int i=1; i<n; i++) // i is length of sorted prefix

{

int val = array[i]; // take new value out of array

int j = i; // j will be location to place val

while((j > 0) && (array[j-1] > val)) // set j; shift others

{

array[j] = array[j-1];

j--;

}

array[j] = val; //put value into its correct place

}

**Advantages of Insertion Sort**

1. Efficient for small sets of data

2. Simple to implement

3. Passes through the array only once.

4. They are adaptive; efficient for data sets that are

already sorted.

**Disadvantage of Insertion Sort**

Less efficient on larger list and arrays

Best case: the array is already sorted

Worst case: elements are completely backwards