# Insertion Sort

## Defination:

In the insertion sort we select one element and suppose that the array at the left of the selected element is sorted and then we insert the selected element in the sorted array that is on the left side according to the sorted array. In this sort we compare the selected element to the previous at check if it is less than the previous the move one more to the left so on at the position where selected element become large and place the element at that point.

## Pseudo Code:

Key

for i=2 to n:

key=array[0]

index=i-1

while key<array[index] and index>=0:

array[index+1]=array[index]

index--

array[index+1]=key

## Code:

def InsertionSort(a):

    for i in range(1,len(a)):

        key=a[i]

        j=i-1

        while j>=0 and key<a[j]:

            a[j+1]=a[j]

            j-=1

        a[j+1]=key

## Time Complexity:

This algorithm takes O(n2)

## Strong Points:

1. If the array is already sorted than this algorithm takes O(n) times.
2. This sorting is stable.
3. If we have small input than this sorting algorithm is most efficient.

## Weak Points:

1. For the average case this sorting algorithm takes O(n2).
2. If we have the input in the reverse sorted order than this sorting algorithm takes maximum time because the inner loop execute to the zero index every time.
3. This sorting algorithm is not so efficient for the large number of input.