int insertsort(int n,int a[])

{

int i,n;

for (i = 1; i < n; i++)

{

key = arr[i]; ----------> (n-1) times. For example c1 times

j = i - 1; --------------->(n-1) times . For example c1 times

/\* Move elements of arr[0..i-1], that are

greater than key, to one position ahead

of their current position \*/

while (j >= 0 && arr[j] > key)

{

arr[j + 1] = arr[j]; -------> constant times c2

j = j - 1; -----------> constant times c2

}

arr[j + 1] = key; ------>(n-1) times. For example c3 timess

}

}

**Time Complexity:**

**Best Case:**

The statement inside the while loop , how many times they are executed will depends on the input. If we have sorted array then it will not go inside the while loop even once, because condition will not be true. So overall time taken in the best case will be equal to (c1+c3)\*(n-1).

T(n)= (c1+c3)(n-1)

=an+b

=O(n)

**Worst Case:**

For example your input is 5 4 3 2 1 and you want arrange these values in ascending order. Then

All the values are shifted like

When i=1 we will have 1 shift

When 1=2 we will have 2 shift

When 1=3 we will have 3 shifts.....

So

T(n)=(c1+c3)\*(n-1) + (1+2+3+4+....)\*c2;

= an2+bn+c

=O(n2)

**Average Case:**

T(n)=O(n2)

FOS000912912044