Insertion sort

Implementation:

```
void insertion(int ara[],int n)
{
   int i,j,x;
   for(i=0;i<n;i++)
   {
      x=ara[i];
      j=i-1;
      while(j>=0 && ara[j]>x)
         ara[j+1]=ara[j],j--;
      ara[j+1]=x;
   }
}
```

Analysis:

We should analyze the algorithm from it's source code:

		Cost	Time
1.	void insertion(int ara[],int n)		
2.	{		
3.	int i,j,x;		
4.	for(i=0;i<n;i++)	c1	n
5.	{		
6.	x=ara[i];	c2	n-1
7.	j=i-1;	c3	n-1
8.	while(j>=0 && ara[j]>x)	c4	$\frac{n(n+1)}{2}$
9.	ara[j+1]=ara[j],j--;	c5	$\frac{n(n+1)}{2}$
10.	ara[j+1]=x;	c6	n-1
11.	}		
12.	}		

Time function will be

f(t) = $c1n + c2(n-1) + c3(n-1) + c4\frac{n(n+1)}{2} + c5\frac{n(n+1)}{2} + c6(n-1)$

Best case occurs when array is already sorted

1	2	3	5	7

In this situation c4 will execute for (n-1) time and c5 will execute for 0 time

So function will be,

f(t) = $c1 * n + c2(n-1) + c3(n-1) + c4(n-1) + c5 * 0 + c6(n-1)$

= $n(c1 + c2 + c3 + c4 + c6) - (c2 + c3 + c4 + c6)$

Which look like $y = an - b$, it is a linear equation

Worst case occurs when the array is reversely sorted,

7	5	3	2	1

Then time function will be,

f(t) = $c1n + c2(n-1) + c3(n-1) + c4\frac{n(n+1)}{2} + c5\frac{n(n+1)}{2} + c6(n-1)$

=$\frac{c1+c2}{2}n^2 + \frac{2(c1+c2+c3+c6)+c4+c5}{2}n - (c2 + c3 + c6)$

Which look like $y = an^2 + bn + c$, it is a quadratic equation.

Time complexity:

Best Case:

We know for best case, $y = an - b$

So, Ω(n)

Worst Case:

For worst case , $y = an^2 + bn + c$

So, $O(n^2)$