Complejidad temporal

- **Burbuja**

```
procedimiento DeLaBurbuja (a0, a1, a2, ..., a(n-1))
    para i ← 1 hasta n − 1 hacer
        para j ← 0 hasta n − i − 1 hacer
            si a(j) > a(j+1) entonces
                aux ← a(j)
                a(j) ← a(j+1)
                a(j+1) ← aux
            fin si
        fin para
    fin para
fin procedimiento
```

n = tamaño del arreglo

Mejor caso	Peor caso
n n n 0 0 0 0	n (n-1) + (n-2) + ... + 3+2+1+1 = n(n-1)/2 + 1 n(n-1)/2 n(n-1)/2 n(n-1)/2 n(n-1)/2 n(n-1)/2
3n = O(n)	$3n^2$ -2n + 1 = O(n^2)

- **Inserción**

```
public void InsercionDirecta()
{
    int auxili;
    int j;
    for (int i = 0; i < vector.Length; i++)
    {
        auxili = vector[i];
        j = i - 1;
        while (j >= 0 && vector[j] > auxili)
        {
            vector[j + 1] = vector[j];
            j--;
        }
        vector[j + 1] = auxili;
    }
}
```

n = tamaño del arreglo

Mejor caso	Peor caso
1 1 n+1 n n n 0 0 n	1 1 n n n 1 + 2 + ... + n -2 + n - 1 + 1 = n(n-1)/2 + 1 n(n-1)/2 n(n-1)/2 n
5n + 3 = O(n)	$3/2n^2$ + 5/2n + 3 = O(n^2)

Complejidad espacial

- **Burbuja**

```
procedimiento DeLaBurbuja (a_0, a_1, a_2, ..., a_(n-1))
    para i ← 1 hasta n − 1 hacer
        para j ← 0 hasta n − i − 1 hacer
            si a_(j) > a_(j+1) entonces
                aux ← a_(j)
                a_(j) ← a_(j+1)
                a_(j+1) ← aux
            fin si
        fin para
    fin para
fin procedimiento
```

Considerado como ordenamiento in place
n = tamaño del arreglo

Mejor caso	Peor caso
1*(4 bytes) 1*(4 bytes) 0 0 0 0	1*(4 bytes) 1*(4 bytes) 0 1*(4 bytes) (crea una variable que luego se desecha antes de la siguiente iteración) 0 0
4 bytes = O(1)	12 bytes = O(1)

Si se tiene en cuenta el arreglo O(1) + O(n) = O(n)

- **Inserción**

```
public void InsercionDirecta()
{
    int auxili;
    int j;
    for (int i = 0; i < vector.Length; i++)
    {
        auxili = vector[i];
        j = i - 1;
        while (j >= 0 && vector[j] > auxili)
        {
            vector[j + 1] = vector[j];
            j--;
        }
        vector[j + 1] = auxili;
    }
}
```

Considerado como ordenamiento in place
n = tamaño del arreglo

Mejor caso	Peor caso
1*(4 bytes) 1*(4 bytes) 1*(4 bytes) 0 0 0 0 0 0	1*(4 bytes) 1*(4 bytes) 1*(4 bytes) 0 0 0 0 0 0
12 bytes = O(1)	12 bytes = O(1)

Si se tiene en cuenta el arreglo O(1) + O(n) = O(n)