

## Ordenación por inserción

### Inserción de un elemento en un arreglo ordenado (manteniendo el orden)

Supongamos que debemos insertar un nuevo elemento en un arreglo ordenado. Se supone que el arreglo no ha sido usado en forma completa. Se recorre el arreglo desde la última posición a la primera buscando el lugar en donde insertar el nuevo dato. Mientras esto se hace, se realiza también un “corrimiento” de los elementos del arreglo, para crear un espacio en donde ubicar el nuevo valor.

10	20	30	40				
----	----	----	----	--	--	--	--

U

u marca la posición de la última celda ocupada con información útil.  
Y sea DATO una variable con el valor a insertar.

Supongamos que se desea insertar el valor 25.-

10	20	30	40				
----	----	----	----	--	--	--	--

U



Se recorre desde el final hasta el inicio del vector, mientras el valor en el índice U sea menor que el valor a insertar.

10	20	30	40	40			
----	----	----	----	----	--	--	--

U

Mientras la condición (dato > valor) no se cumpla, se copia el valor a la posición siguiente (U+1)

10	20	30	30	40			
----	----	----	----	----	--	--	--

U

10	20	25	30	40			
----	----	----	----	----	--	--	--

U

Cuando la se cumple, se inserta el valor a la siguiente posición (u+1)

Funcion para insertar:

```

void insertar(int a[8], int u, int dato)
{
    int i = u;
    while (i >= 0 && dato < a[i])
    {
        a[i+1] = a[i];
        i--;
    }
    a[i+1] = dato;
}

```

Arreglo

Ultima posición

Valor a insertar

Mantras dato sea menor

Se copia el valor a la posición siguiente

Resta una posición

Inserta el nuevo dato en la posición correspondiente

### Algoritmo de ordenación por inserción

Supongo un arreglo completo y desordenado.

Con el algoritmo de inserción anterior, y limitando el vector a una “zona a ordenar”, se utiliza el valor siguiente a la posición U [U+1] para insertarlo en la ubicación que le corresponde dentro de la zona a ordenar. De esta forma hasta completar el total del vector.

18	1	12	9	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

1	18	12	9	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

1	12	18	9	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

1	9	12	18	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

1	9	12	18	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

18	1	12	18	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

u

18	1	12	18	19	23	45	33
0	1	2	3	4	5	6	7

Zona a ordenar

u

### Algoritmo de ordenacion

```
void ordenacion_insercion(int a[8])
{
    int u=0;
    while (u < 7) //llega hasta la posición del anteúltimo elemento del arreglo.
    {
        insertar(a, u, a[u+1]);
        u++;
    }
}
```

Valor a insertar

Arreglo

Posición final de la Zona a ordenar

Tener en cuenta que llega hasta la anteultima ubicacion porque el ultimo valor queda ordenado por la pasada anterior.

Por ejemplo el numero "33" de nuestro ejemplo anterior queda ordenado con u=6