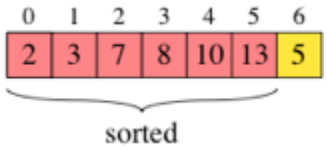
ORDENAMIENTO POR INSERCIÓN

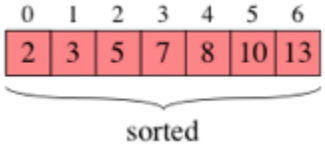
Inicialmente se tiene un solo elemento, que obviamente es un conjunto ordenado. Después, cuando hay **K** elementos ordenados de menor a mayor, se toma el elemento **K+1** y se compara con todos los elementos ya ordenados, deteniéndose cuando se encuentra un elemento menor (todos los elementos mayores han sido desplazados una posición a la derecha) o cuando ya no se encuentran elementos (todos los elementos fueron desplazados y este es el más pequeño). En este punto se *inserta* el elemento **K+1** debiendo

n términos de arreglos, imagina que el subarreglo con índices del 0 al 555 ya está ordenado, y queremos insertar el elemento que por ahora está en el índice 6 en este subarreglo ordenado, de manera que el subarreglo con los índices del 0 al 6 esté ordenado. Aquí está cómo empezamos:



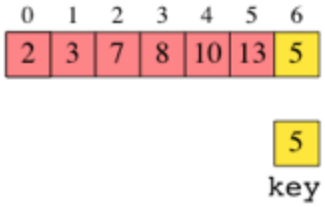
Inserción

Y aquí está cómo se debe ver el subarreglo cuando hayamos terminado:



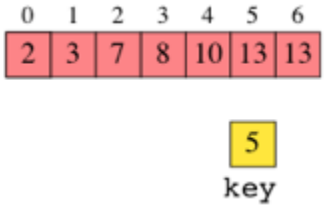
Inserción

Para insertar el elemento que está en la posición 6 en el subarreglo a su izquierda, lo comparamos repetidamente con los elementos a su izquierda, moviéndonos de derecha a izquierda. Llamemos al elemento en la posición 6 la **llave**. Cada vez que veamos que la llave es menor que un elemento a su izquierda, desplazamos ese elemento una posición a la derecha, ya que sabemos que la llave tendrá que quedar a la izquierda de ese elemento. Vamos a necesitar hacer dos cosas para lograr que esta idea funcione: necesitamos tener una operación llamada **desplazar** que desplaza un elemento una posición hacia la derecha, y necesitamos almacenar el valor de la llave en un sitio separado (de manera que el elemento que queda inmediatamente a su izquierda no la reemplace). En nuestro ejemplo, vamos a jalar el elemento en el índice 6 a una variable llamada `key` (llave):



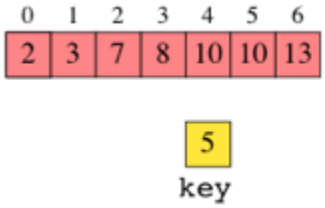
Inserción

Ahora comparamos `key` con el elemento en la posición 5. Encontramos que `key` (5) es menor que el elemento en la posición 5 (13), así que desplazamos este elemento a la posición 6:



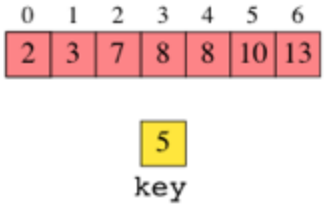
Inserción

Observa que la operación de desplazar simplemente copia el elemento una posición a la derecha. A continuación, comparamos `key` con el elemento en la posición 4. Encontramos que `key` (5) es menor que el elemento en la posición 4 (10), y desplazamos este elemento a la derecha:



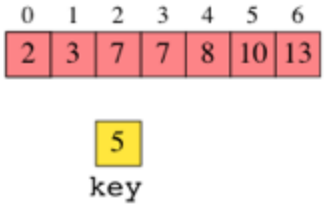
Inserción

A continuación, comparamos `key` con el elemento en la posición 3 y desplazamos ese elemento a la derecha:



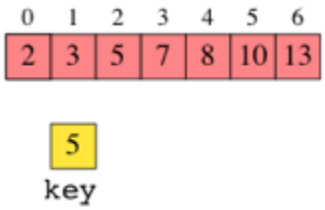
Inserción

Sucede lo mismo con el elemento en la posición 2:



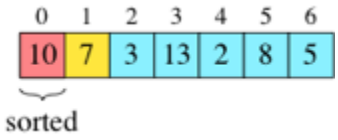
Inserción

Ahora llegamos al elemento en la posición 1, que tiene el valor 3. Este elemento es menor que `key`, así que *no* lo desplazamos hacia la derecha. En lugar de eso, ponemos `key` en la posición inmediata a la derecha de este elemento (es decir, en la posición 2), cuyo elemento fue el último en desplazarse a la derecha. El resultado es que el subarreglo con índices del 0 al 6 ha quedado ordenado:



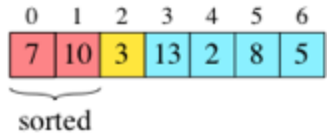
Inserción

El ordenamiento por inserción inserta repetidamente un elemento en el subarreglo ordenado que está a su izquierda. Al inicio, podemos decir que el subarreglo que solo contiene al índice 0 está ordenado, ya que contiene un solo elemento, y ¿cómo un solo elemento podría *no* estar ordenado con respecto a sí msimo? Debe estar ordenado. Trabajemos un ejemplo. Aquí está nuestro arreglo inicial:



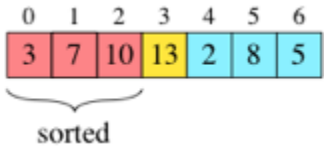
Ordenamiento por inserción

Dado que el subarreglo que solo contiene el índice 0 es nuestro subarreglo ordenado inicial, la primera llave está en el índice 1. (Vamos a mostrar el subarreglo ordenado en rojo, la llave en amarillo y la porción del arreglo con la que aún tenemos que trabajar en azul). Insertamos la llave en el subarreglo ordenado que está a su izquierda:



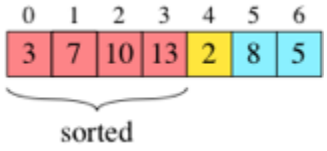
Ordenamiento por inserción

Ahora el subarreglo ordenado va del índice 0 al 1, y la nueva llave está en el índice 2. La insertamos en el subarreglo ordenado a su izquierda:

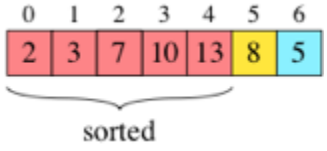


Ordenamiento por inserción

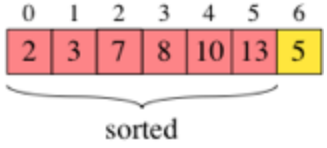
Seguimos adelante, considerando a cada elemento del arreglo en turno como la llave, e insertándolo en el subarreglo ordenado a su izquierda:



Ordenamiento por inserción

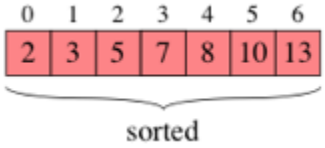


Ordenamiento por inserción



Ordenamiento por inserción

Una vez que hayamos insertado el elemento hasta la derecha en el arreglo, habremos ordenado todo el arreglo:



Pseudocódigo

```
PARTITION(A, p, r)
1  x = A[r]
2  i = p − 1
3  for j = p to r − 1
4       if A[j] ≤ x
5            i = i + 1
6            exchange A[i] with A[j]
7  exchange A[i + 1] with A[r]
8  return i + 1
```