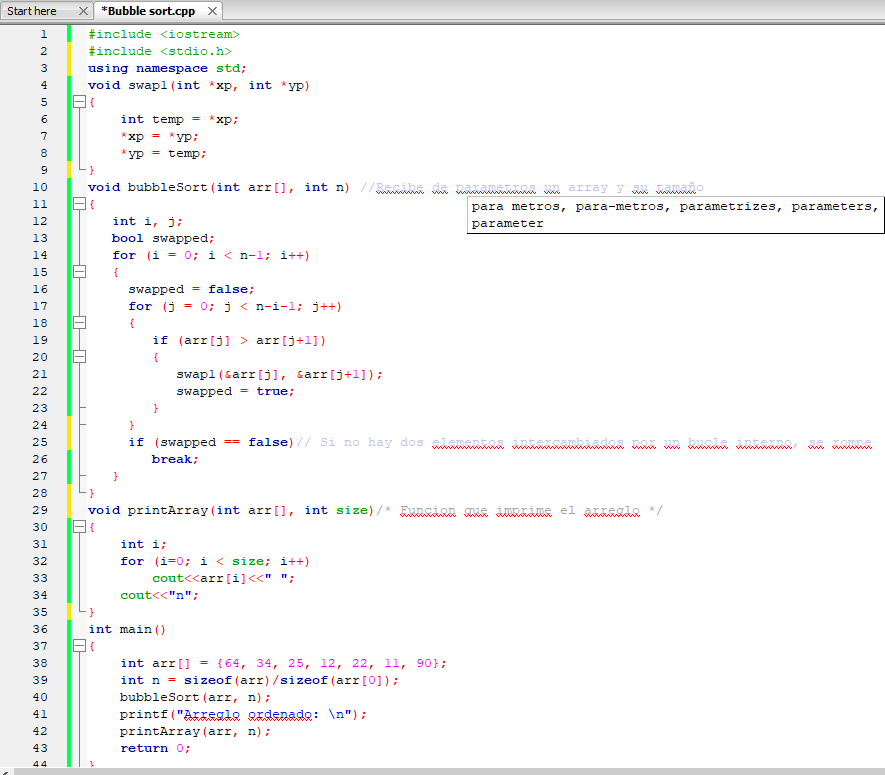
**ALGORITMOS DE ORDEMANIENTO**

**Ordenamiento de Burbuja (Bubble Sort)**

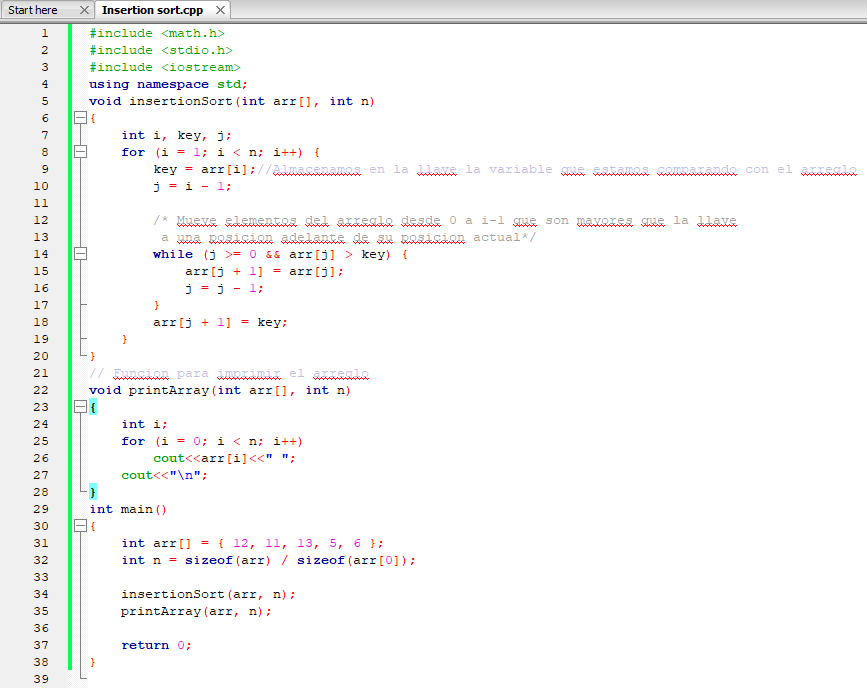
Es un sencillo algoritmo de ordenamiento. Funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada. Este algoritmo obtiene su nombre de la forma con la que suben por la lista los elementos durante los intercambios, como si fueran pequeñas "burbujas". También es conocido como el método del intercambio directo. Dado que solo usa comparaciones para operar elementos, se lo considera un algoritmo de comparación, siendo uno de los más sencillos de implementar.



**Ordenamiento de Inserción (Insertion Sort)**

El ordenamiento por inserción (insertion sort en inglés) es una manera muy natural de ordenar para un ser humano, y puede usarse fácilmente para ordenar un mazo de cartas numeradas en forma arbitraria. Requiere O(n²) operaciones para ordenar una lista de n elementos.

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 desplazarse los demás elementos.



**Ordenamiento por Mezcla (Merge Sort)**

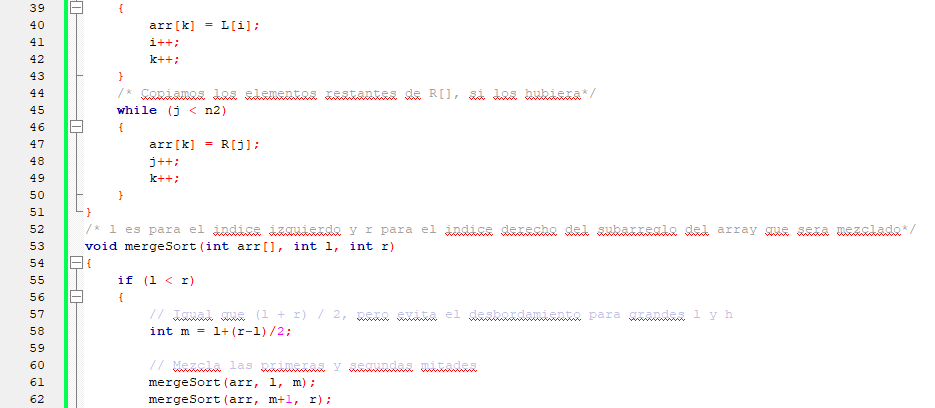
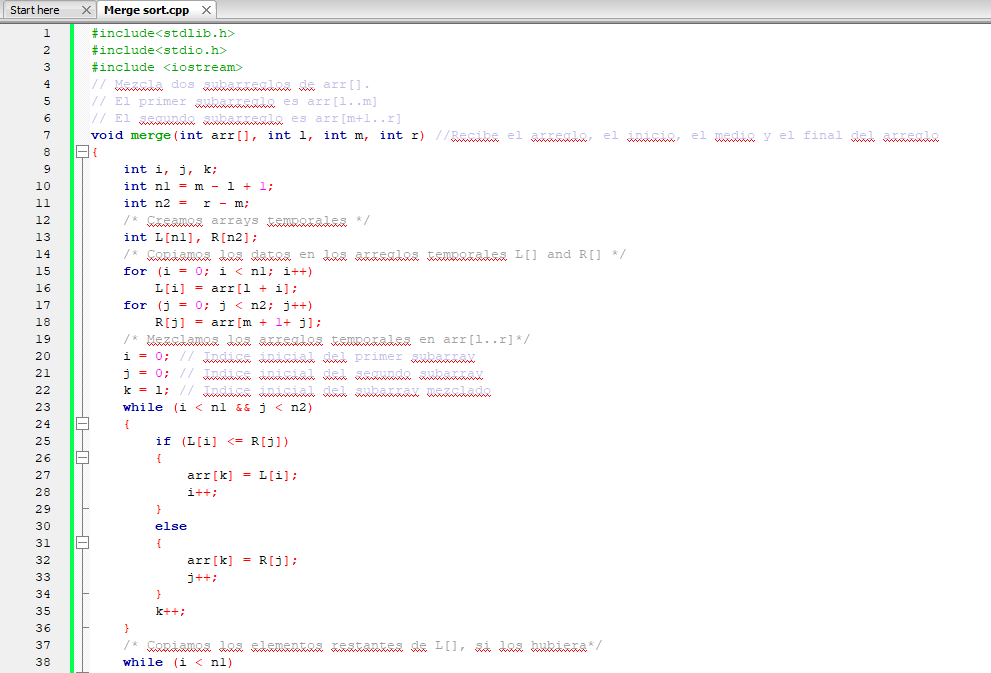
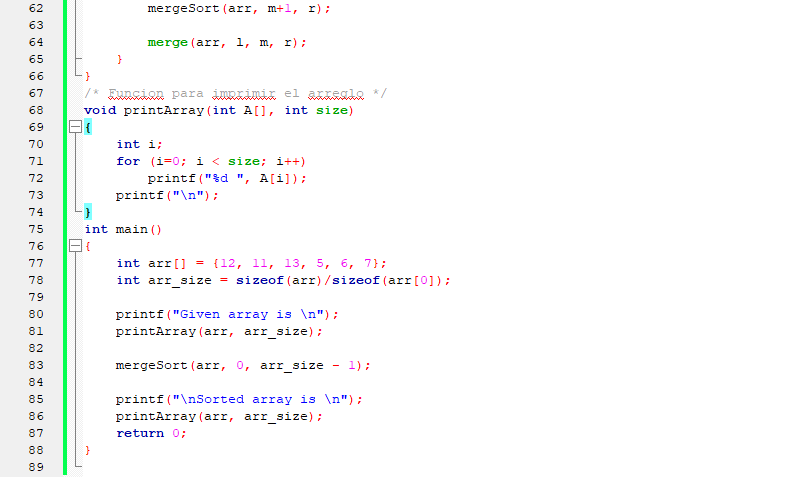
El algoritmo de ordenamiento por mezcla (*merge sort* en inglés) es un algoritmo de ordenamiento externo estable basado en la técnica divide y vencerás. Es de complejidad [O](https://es.wikipedia.org/wiki/Cota_superior_asint%C3%B3tica)(*n* log *n*).

Conceptualmente, el ordenamiento por mezcla funciona de la siguiente manera:

1. Si la longitud de la lista es 0 ó 1, entonces ya está ordenada. En otro caso:
2. Dividir la lista desordenada en dos sublistas de aproximadamente la mitad del tamaño.
3. Ordenar cada sublista recursivamente aplicando el ordenamiento por mezcla.
4. [Mezclar](https://es.wikipedia.org/w/index.php?title=Algoritmo_de_mezcla&action=edit&redlink=1) las dos sublistas en una sola lista ordenada.

El ordenamiento por mezcla incorpora dos ideas principales para mejorar su tiempo de ejecución:

1. Una lista pequeña necesitará menos pasos para ordenarse que una lista grande.
2. Se necesitan menos pasos para construir una lista ordenada a partir de dos listas también ordenadas, que a partir de dos listas desordenadas. Por ejemplo, sólo será necesario entrelazar cada lista una vez que están ordenadas.



**Ordenamiento Rápido (Quick Sort)**

El algoritmo trabaja de la siguiente forma:

* Elegir un elemento del arreglo de elementos a ordenar, al que llamaremos **pivote**.
* Resituar los demás elementos de la lista a cada lado del pivote, de manera que a un lado queden todos los menores que él, y al otro los mayores. Los elementos iguales al pivote pueden ser colocados tanto a su derecha como a su izquierda, dependiendo de la implementación deseada. En este momento, el pivote ocupa exactamente el lugar que le corresponderá en la lista ordenada.
* La lista queda separada en dos sublistas, una formada por los elementos a la izquierda del pivote, y otra por los elementos a su derecha.
* Repetir este proceso de forma recursiva para cada sublista mientras éstas contengan más de un elemento. Una vez terminado este proceso todos los elementos estarán ordenados.

Como se puede suponer, la [eficiencia del algoritmo](https://es.wikipedia.org/wiki/Eficiencia_Algor%C3%ADtmica) depende de la posición en la que termine el pivote elegido.

* En el mejor caso, el pivote termina en el centro de la lista, dividiéndola en dos sublistas de igual tamaño. En este caso, el orden de [complejidad del algoritmo](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_la_complejidad_computacional) es [**O**](https://es.wikipedia.org/wiki/Cota_superior_asint%C3%B3tica)**(n·log n)**.
* En el peor caso, el pivote termina en un extremo de la lista. El orden de complejidad del algoritmo es entonces de **O(n²)**. El peor caso dependerá de la implementación del algoritmo, aunque habitualmente ocurre en listas que se encuentran ordenadas, o casi ordenadas. Pero principalmente depende del pivote, si por ejemplo el algoritmo implementado toma como pivote siempre el primer elemento del [array](https://es.wikipedia.org/wiki/Array" \o "Array), y el array que le pasamos está ordenado, siempre va a generar a su izquierda un array vacío, lo que es ineficiente.
* En el caso promedio, el orden es **O(n·log n)**.

No es extraño, pues, que la mayoría de optimizaciones que se aplican al algoritmo se centren en la elección del **pivote**.

