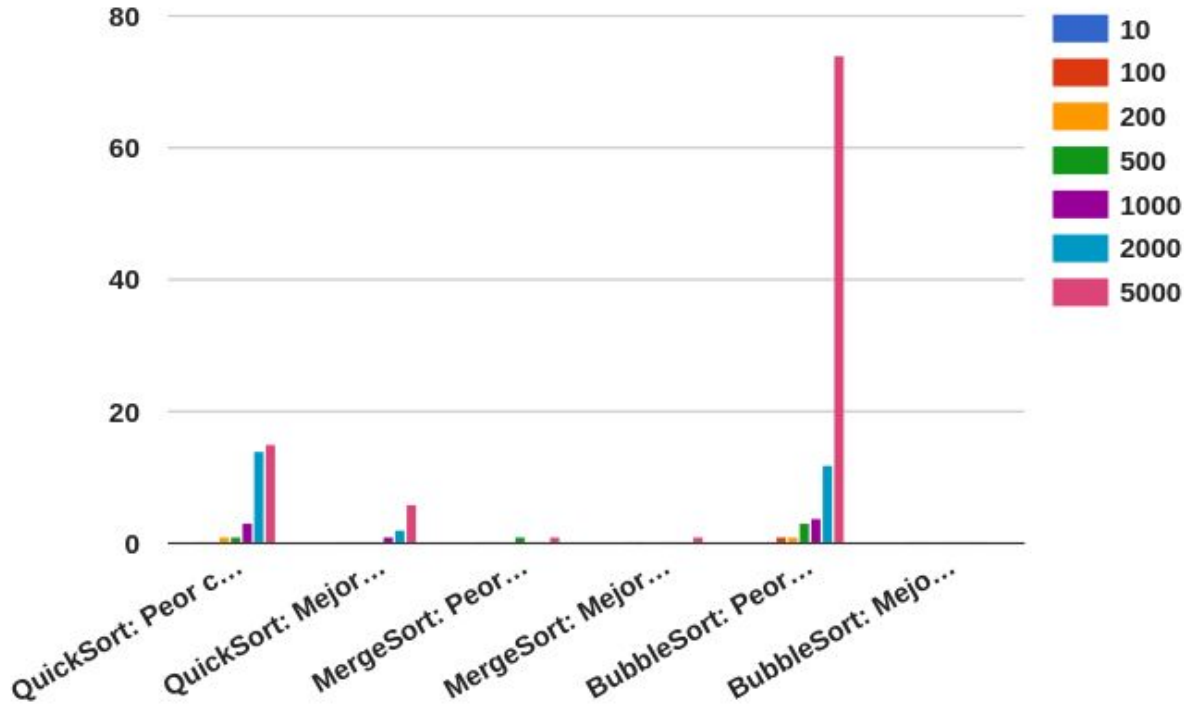

En la gráfica anterior notamos el peor y mejor caso de cada ordenamiento dependiendo del tamaño del ejemplar. Del lado izquierdo se muestra el tiempo que tarda.

- Quick Sort: peor caso.

Sucede cuando la partición, sobre un arreglo de n elementos, nos regresa un subarreglo de tamaño (n-1) cada vez. En la implementación de la práctica el primer elemento del arreglo es el pivote, entonces una secuencia ordenada ya sea ascendentemente o descendentemente lo provoca. Particularmente en la práctica se usó una **secuencia descendente**. Por otra parte este toma $O(n^2)$.

- Quick Sort: mejor caso.

En la implementación que se hizo tomando el primer elemento como pivote, el mejor caso es generado al tener en cada iteración, al elemento que representa la mediana de los elementos de la secuencia en la primera posición.
De esta manera, la función Partición regresará la mitad. Esto significa que cada vez el arreglo es dividido en subarreglos de tamaño similar. En este caso toma $O(nlogn)$.

- Merge Sort: peor caso.

Para este tomaremos como referencia el desempeño computacional del procedimiento mezcla, entonces basta tener una secuencia que mezcla haga el mayor número de comparaciones. En este caso toma $O(nlogn)$.

Para construir este peor caso base se usó una técnica bottom-up.

1. Suponemos que el arreglo ordenado es {0, 1, 2, …, 7}
2. Para el peor caso del arreglo anterior debe ser {0, 2, 4, 6, 1, 3, 5, 7} por los dos subarreglos que generan el peor caso.
3. Usando la misma técnica tenemos {0, 4, 2, 6, 1, 5, 3, 7}.
4. Repitiendo el mismo proceso {4, 0, 6, 2, 5 , 1, 3, 7}.

Se puede ver claramente que se emplea recursividad para ir generando el peor caso. Es decir, dada una secuencia secuencia ordenada se da como resultado el peor caso siguiendo el procedimiento recursivo anterior(este se ve con mayor claridad en la implementación.).

- Merge Sort: mejor caso.

Para este tomaremos como referencia el desempeño computacional del procedimiento mezcla, entonces basta con tener una secuencia ya ordenada para generar el mejor caso. En este caso toma $O(nlogn)$.

- Bubble Sort: mejor caso.

Este se da cuando la secuencia dada ya está ordenada ya que no se tiene que intercambiar ningún elemento. En este caso toma $O(n)$.

- Bubble Sort: peor caso.

Se da cuando la secuencia dada esta orden inverso, ya que se tienen que realizar todos los intercambios para devolver la secuencia en el orden correspondiente. En este caso toma $O(n^2)$.