ALGORITMOS DE ORDENACIÓN

Vamos a analizar y comparar los distintos tiempos de distintos algoritmos de ordenación. En este caso estudiaremos los algoritmos de **Insertion Sort**, **Selection Sort**, **Bubble Sort**, **Merge Sort**, **Quicksort** y **ordenamiento por rango**. Para analizarlos hemos hecho pruebas de tiempo con N elementos variando N de 12000 a 24000 elementos con pasos de 2000. A parte lo hemos hecho para varios casos:

- Caso general: ordenar una lista de 0-99 con números generados aleatoriamente.
- Mejor caso: ordenar una lista, ya ordenada.
- Peor caso: ordenar una lista ordena al revés, es decir que tiene que cambiar de posición todos los valores.

Los datos obtenidos en las pruebas de tiempo han sido los siguientes:

INSERTION SORT

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.001	0.075	0.134
14000	0.002	0.102	0.180
16000	0.003	0.134	0.237
18000	0.004	0.167	0.299
20000	0.004	0.210	0.368
22000	0.005	0.251	0.447
24000	0.006	0.301	0.533

SELECTION SORT

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.163	0.168	0.139
14000	0.222	0.229	0.188
16000	0.291	0.301	0.245
18000	0.373	0.381	0.325
20000	0.459	0.470	0.389
22000	0.562	0.591	0.472
24000	0.657	0.678	0.565

Raúl Sancho y Néstor Pérez

BUBBLE SORT

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.297	0.518	0.326
14000	0.405	0.705	0.447
16000	0.527	0.924	0.581
18000	0.694	1.191	0.752
20000	0.825	1.466	0.910
22000	0.999	1.768	1.109
24000	1.203	2.118	1.319

MERGE SORT

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.056	0.052	0.058
14000	0.070	0.065	0.072
16000	0.080	0.076	0.085
18000	0.104	0.096	0.106
20000	0.121	0.109	0.122
22000	0.135	0.127	0.135
24000	0.143	0.142	0.156

QUICKSORT

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.000	0.001	0.001
14000	0.000	0.001	0.001
16000	0.000	0.001	0.001
18000	0.000	0.001	0.001
20000	0.000	0.002	0.001
22000	0.000	0.002	0.001
24000	0.000	0.002	0.000

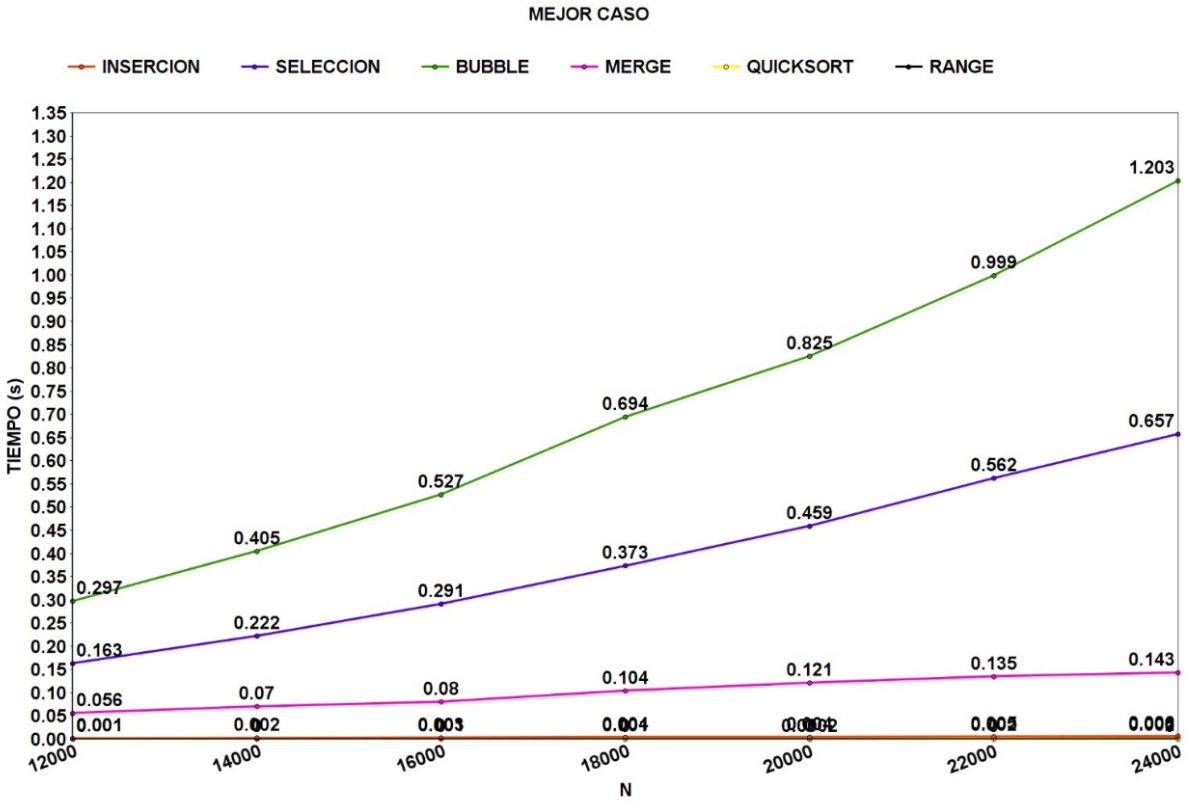
ORDENACIÓN POR RANGO

N	MEJOR CASO	CASO GENEREAL	PEOR CASO
12000	0.001	0.001	0.001
14000	0.000	0.001	0.002
16000	0.001	0.001	0.001
18000	0.001	0.002	0.002
20000	0.002	0.002	0.002
22000	0.002	0.002	0.002
24000	0.002	0.003	0.002

Raúl Sancho y Néstor Pérez

Con estos resultados podemos hacer una gráfica que compare que algoritmos es más rápido para cada caso y el número de elementos introducidos.

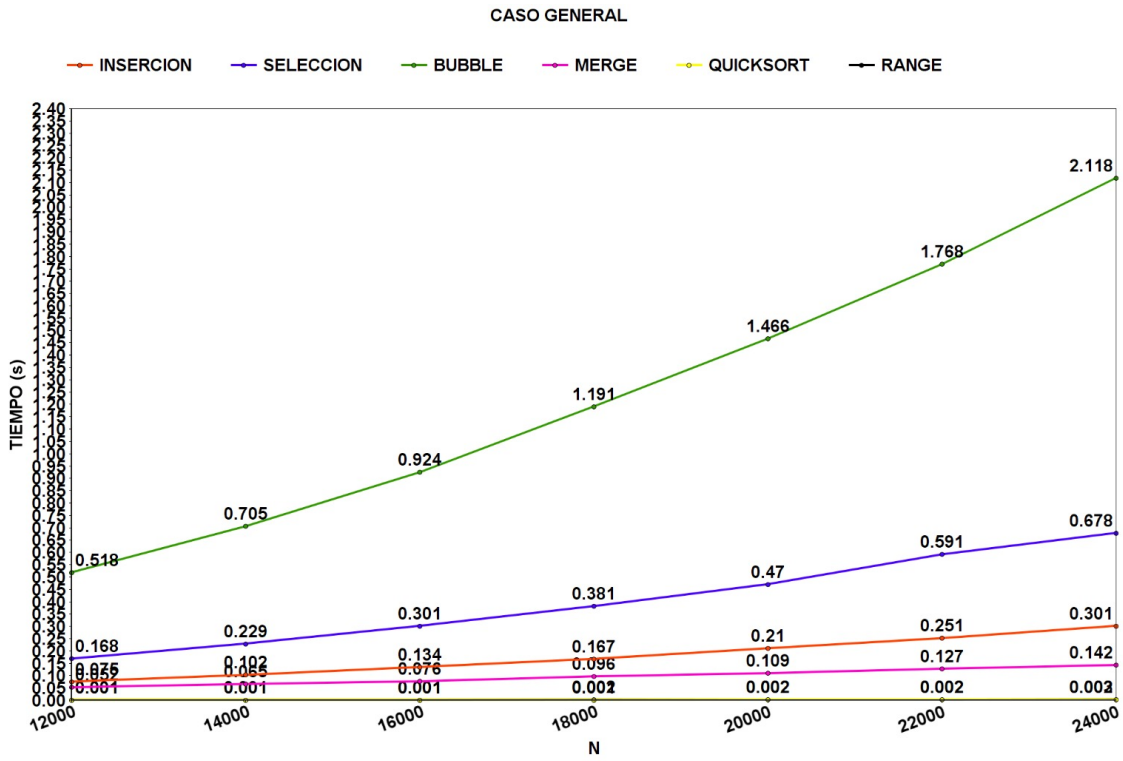
MEJOR CASO



- **Inserción**: según la teoría, la ordenación por inserción debería tener una forma de $0(n)$ en el mejor caso. Esto, a pesar de que en la gráfica se vea muy poco debido al poco tiempo que tarda no se aprecia muy bien, pero si vamos a su tabla vemos que cada vez que aumentamos en 200 elementos, su tiempo se incrementa en 0.001 segundos.
- **Selección**: según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$ en el mejor caso. En la gráfica se aprecia claramente como cada vez que añadimos 2000 elementos el tiempo que tarda se incrementa cada vez más.
- **Burbuja:** según la teoría, la ordenación por selección debería tener una forma de $0(n)$ en el mejor caso. En la gráfica se aprecia claramente es una subida lineal, excepto en el caso de 18000 elementos que tiene un pequeño pico.
- **Merge sort:** según la teoría, la ordenación por selección debería tener una forma de $0(n.\log(n))$ en el mejor caso. En la gráfica no se puede apreciar demasiado bien, ya que esta función para muchos elementos se parece a una gráfica de x = y, y sólo se notaría la diferencia para pocos elementos, donde esta función sería más rápida ya que contra más grande sea más llamadas recursivas debe hacer.

- **Quicksort:** según la teoría, la ordenación por selección debería tener una forma de $0(n.\log(n))$ en el mejor caso. En la gráfica no se puede apreciar debido a que Quicksort es muy rápido para este número de elementos y su tiempo casi permanece constante.
- **Ordenación por rango:** según la teoría, la ordenación por selección debería tener una forma de $0(n)$ en el mejor caso. En la gráfica no se puede apreciar ya que la ordenación por rango es muy eficiente y rápida, lo malo de esta, es que debe cumplirse una serie de condiciones como el rango de números.

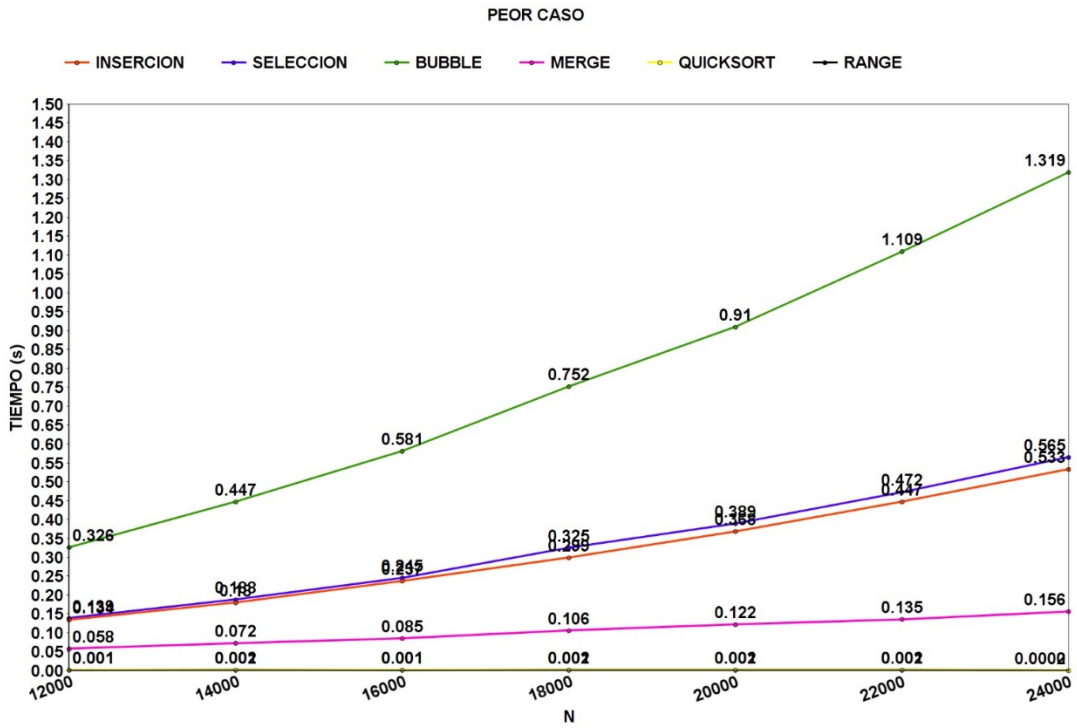
CASO GENERAL



- **Inserción**: según la teoría, la ordenación por inserción debería tener una forma de $0(n^2)$ en el caso general, lo que se puede ver perfectamente en la gráfica como cada vez su tiempo incrementa más.
- **Selección**: según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$ en el caso general que también se aprecia perfectamente.
- **Burbuja:** según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$ en el caso general. En la gráfica es el que mejor se aprecia debido a su diferencia de velocidad.
- **Merge sort:** según la teoría, la ordenación por selección debería tener una forma de $0(n.\log(n))$ en el caso general. En la gráfica no se puede apreciar ya que debido a los valores introducidos la gráfica aparece constante.

- **Quicksort:** según la teoría, la ordenación por selección debería tener una forma de $0(n.\log(n))$ en el caso general. Al igual que antes, su eficiencia nos impide ver su gráfica claramente
- **Ordenación por rango:** según la teoría, la ordenación por selección debería tener una forma de $0(n)$ en el caso general.

PEOR CASO



- **Inserción**: según la teoría, la ordenación por inserción debería tener una forma de $0(n^2)$ en el peor, en la gráfica vemos que está a la par del método de selección.
- **Selección**: según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$.
- **Burbuja:** según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$ en el por caso.
- **Merge sort:** según la teoría, la ordenación por selección debería tener una forma de $0(n.\log(n))$ en el peor caso.
- **Quicksort:** según la teoría, la ordenación por selección debería tener una forma de $0(n^2)$ en el caso general. Al igual que antes, su eficiencia nos impide ver su gráfica claramente
- **Ordenación por rango:** según la teoría, la ordenación por selección debería tener una forma de $0(n)$ en el peor caso.

En conclusión, vemos que en cualquier caso el algoritmo más lento es el BUBBLE SORT, mientras que los más rápidos son el QUICKSORT y la ordenación por RANGO. Esto se debe a la cantidad de veces que tiene que hacer el bucle que vaya comparando cada pareja de números, mientras que el QUICKSORT es más rápido debido al bajo número de operaciones que realiza. La ordenación por RANGO es muy eficiente ya que en todos sus casos es $0(n)$, lo único malo es que tiene cumplir ciertas condiciones, como que sus valores estén en un rango determinado de números.

El método de INSERCIÓN es más rápido que el de SELECCIÓN, ya que en el mejor caso INSERCIÓN $0(n)$ mientras SELECCIÓN es $0(n^2)$. Vemos como esto cambia en el resto de casos ya que el de INSERCIÓN pasa a tener $0(n^2)$. y en el peor caso vemos como sus gráficas son prácticamente iguales.

Por último MERGE SORT, en todos sus casos es $0(n.\log(n))$ por lo que sus gráficas son prácticamente iguales en todos los casos.