Análisis de Algoritmos 2018/2019

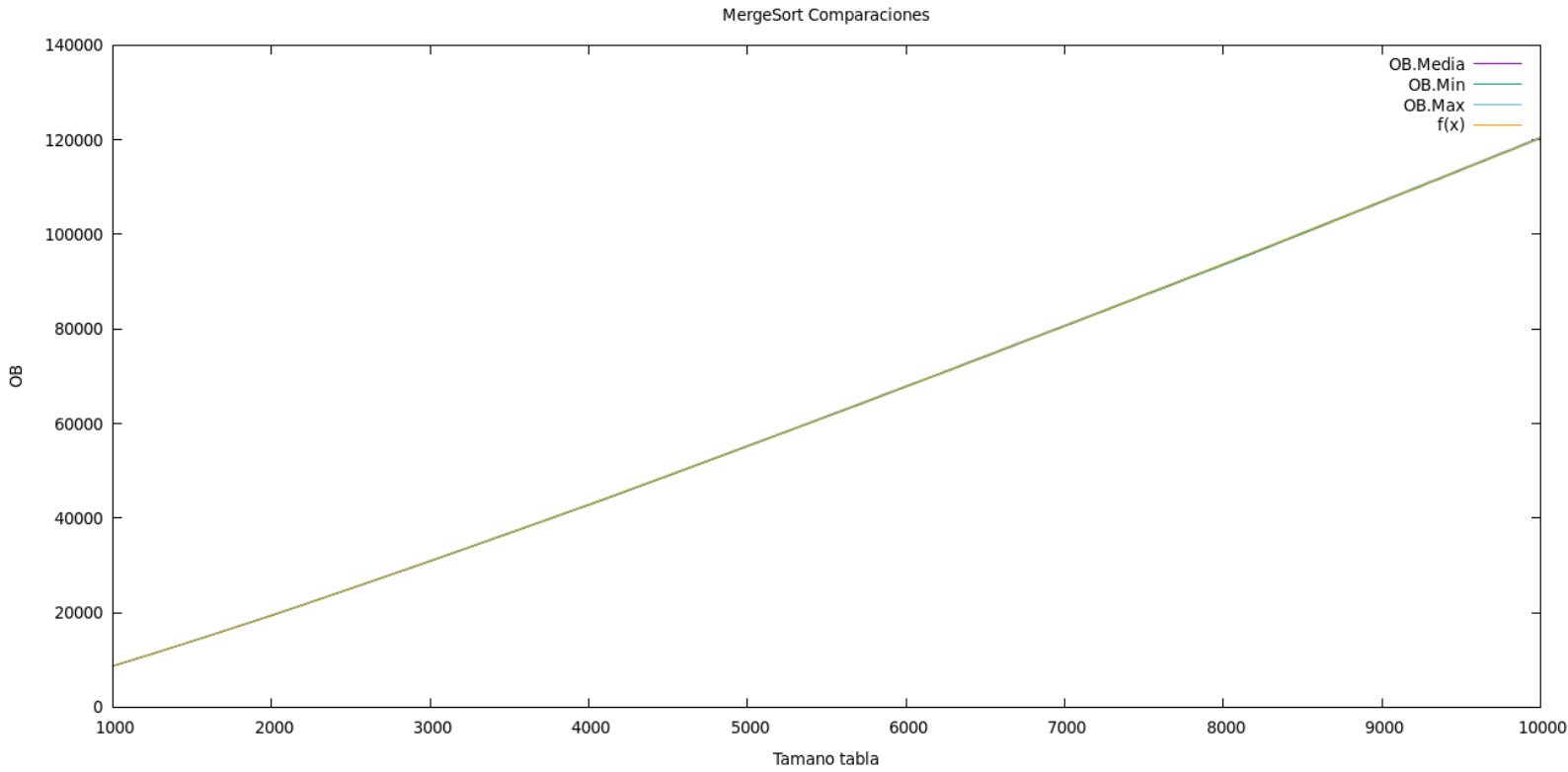
Práctica 2

Miguel Ángel Luque López y Nicolás Serrano Salas. Grupo. 1212

Código	Gráficas	Memoria	Total

5. Resultados, Gráficas

5.2 Apartado 2



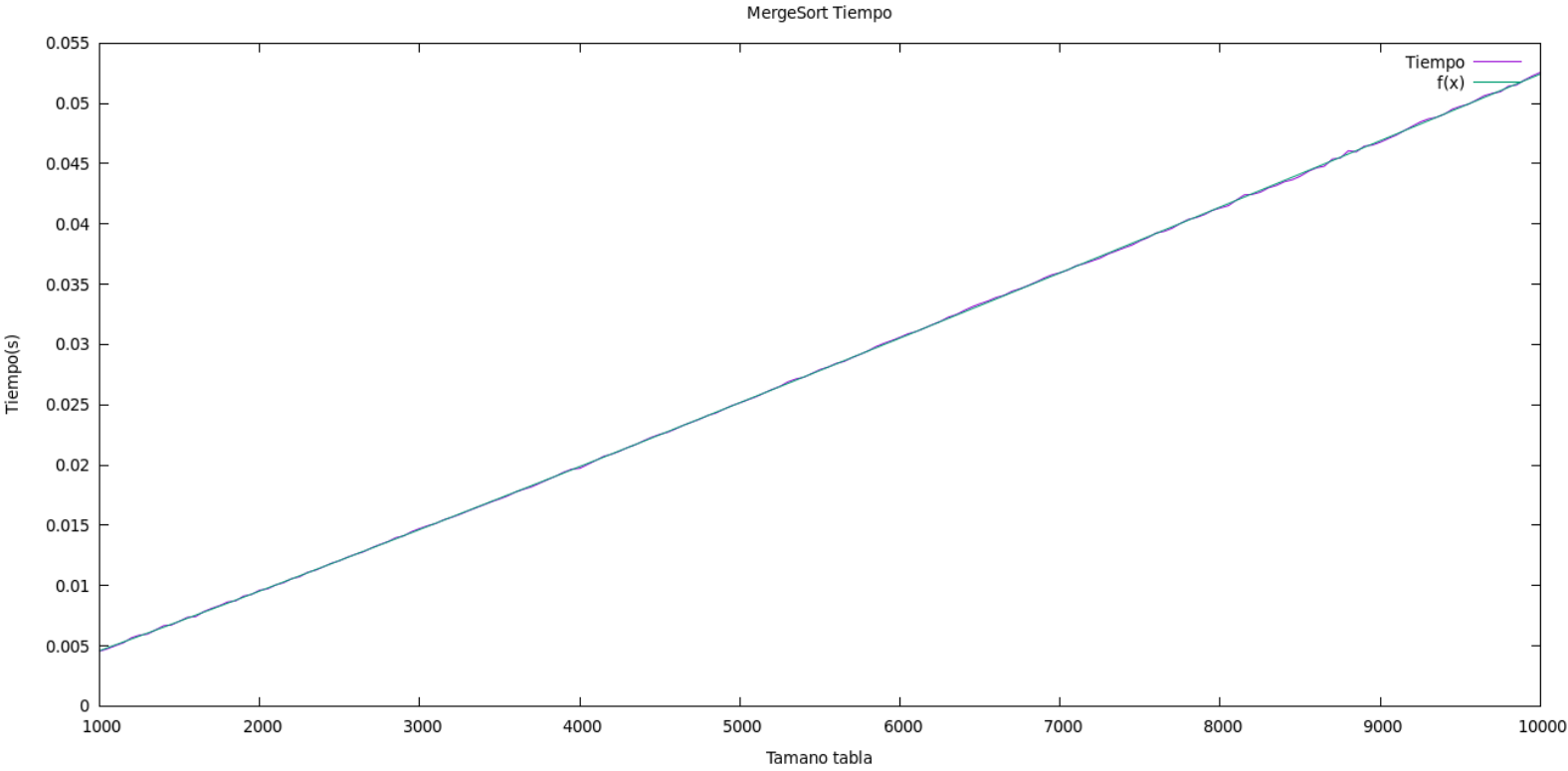
Podemos observar que el caso mejor, peor y medio son muy similares pues todas son

$W_{MS}(N) \leq Nlg(N)+O(N)$

$B_{MS}(N) \geq (1/2)Nlg(N)$

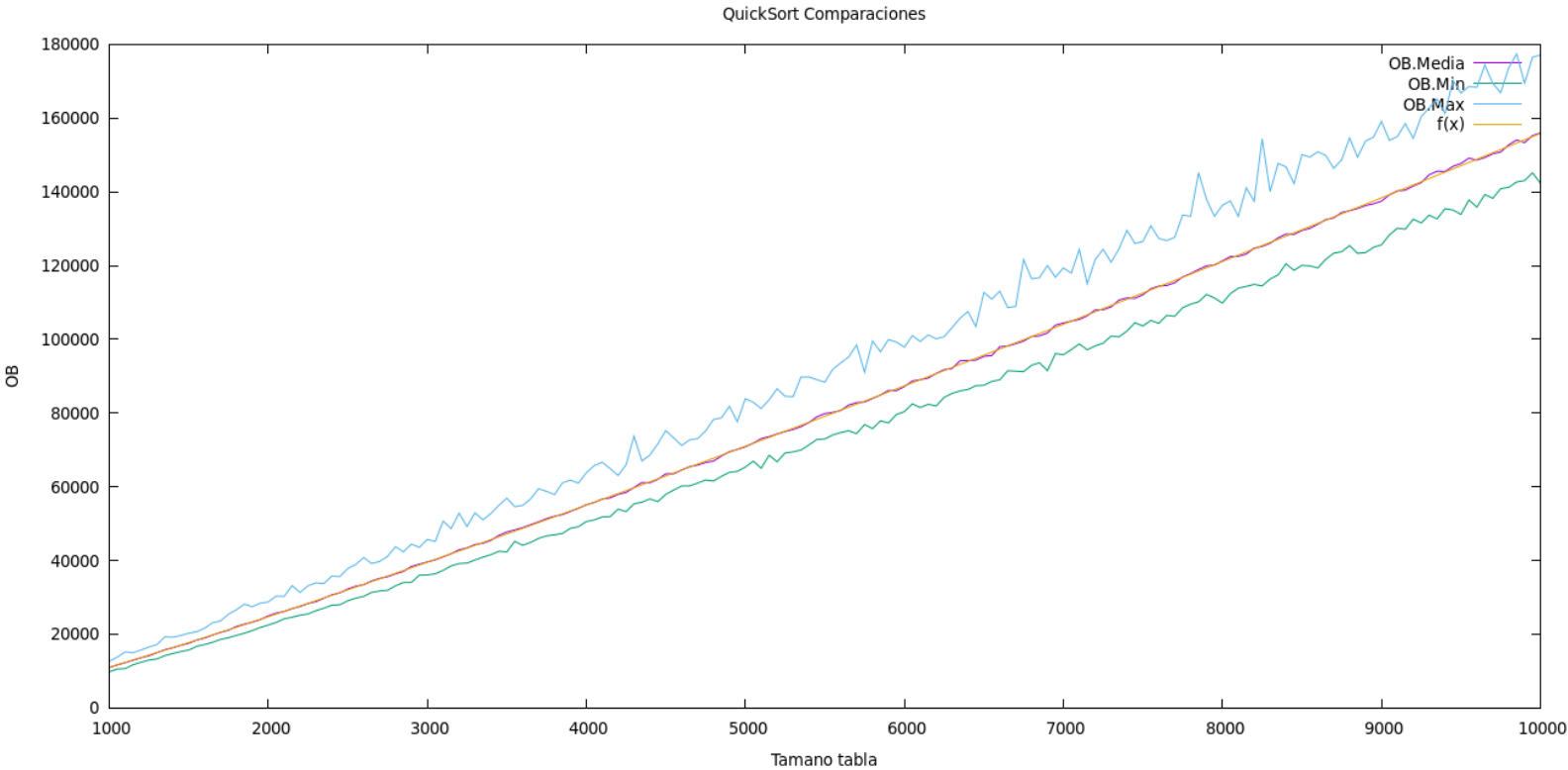
$A_{MS}(N) = \Theta(Nlg(N))$

Y al no realizar N! permutaciomes y además ser estas aleatorias, nos quedan valores cercanos al A_{MS} siendo estos del orden de $Nlg(N) \pm O(N)$



Observamos que la función del tiempo medio es una función del orden a*x*log(x) +b*x+c que hemos ajustado con el comando fit.

5.4 Apartado 4



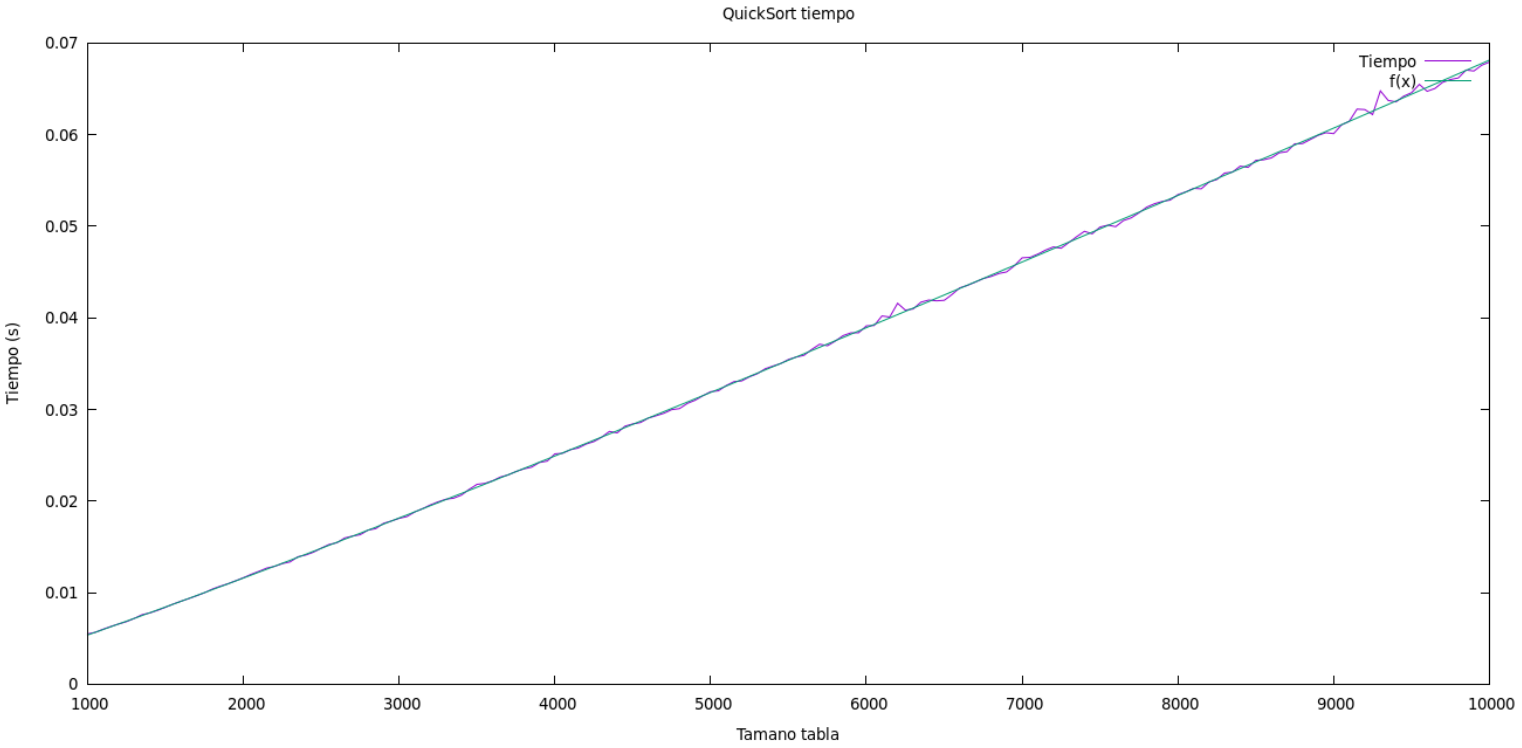
Podemos observar que el caso mejor, peor y medio son tan similares como en MergeSort pero no los esperados pues son

$W_{QS}(N) = N^2/2-N/2$

$B_{MS}(N) = \Theta(Nlg(N))$

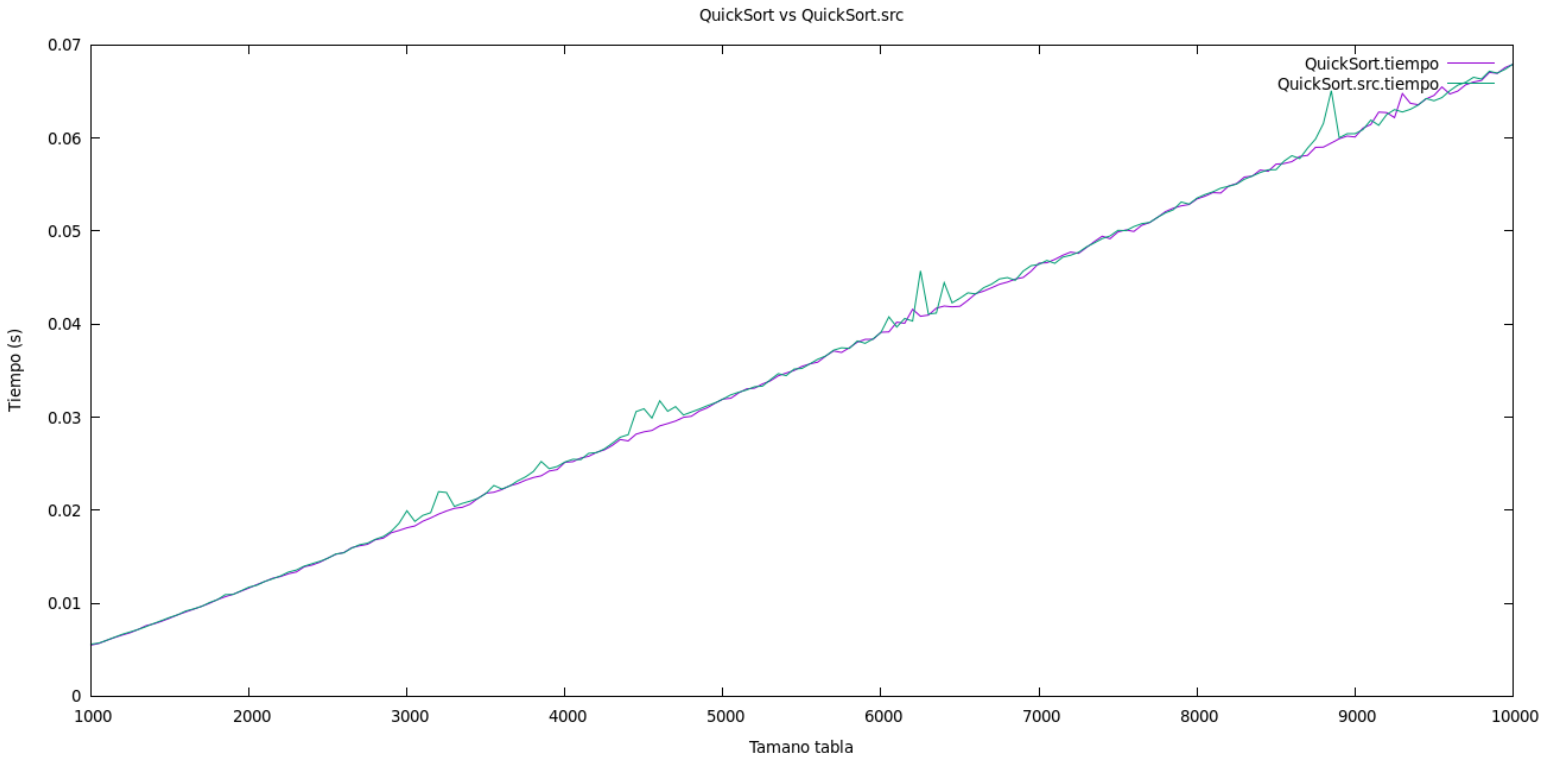
$A_{MS}(N) = \Theta(Nlg(N))$

Y al no realizar N! permutaciomes y además ser estas aleatorias, nos quedan valores cercanos al A_{MS} siendo estos del orden de $Nlg(N) \pm O(N)$



Observamos que la función del tiempo medio es una función del orden ax*log(x)+bx+c que hemos ajustado con el comando fit.

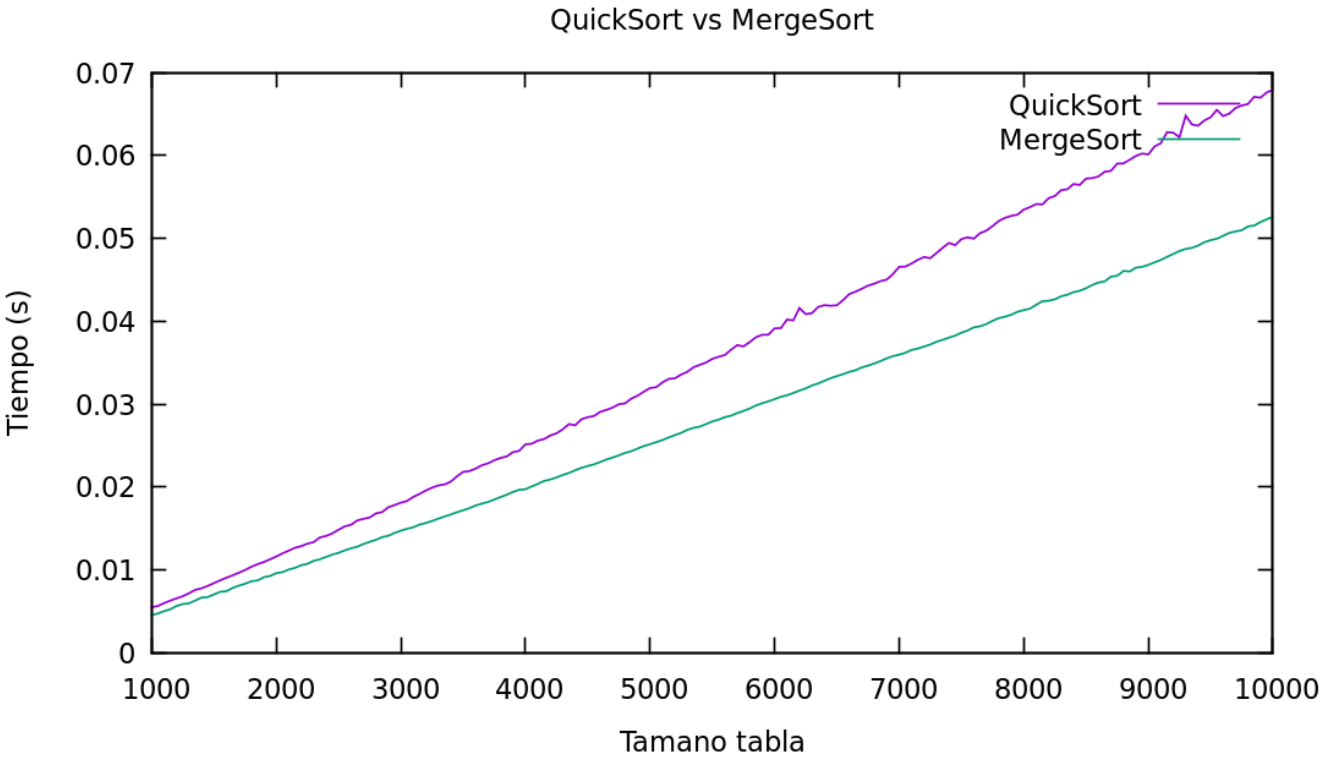
5.5 Apartado 5



Observamos que el tiempo medio de QuickSort y de QuickSort_src es practicamente el mismo salvo cuestiones de azar al ser calculadas con permutaciones completamente aleatorias y observamos algunos picos donde diverge esta igualdad que han podido ser por estas permutaciones aleatorias o errores causados por el procesador

5. Respuesta a las preguntas teóricas.

5.1 Pregunta 1



Observamos en esta gráfica que el rendimiento de MergeSort es mejor que el de QuickSort y además su recta es menos picuda aunque estos picos pueden ser causados bien por el ordenador al realizar la simulación o por la aleatoriedad de la permutación en la utilización de los algoritmos de ordenación.

5.3 Pregunta 3

$B_{MS}=(1/2)Nlog(N)$ y $W_{MS}=Nlog(N)+O(N)$

$B_{QS}=O(Nlog(N))$ y $W_{QS}=N^2/2-N/2$

Para calcular el caso mejor, peor y medio tendríamos que introducir la mejor permutación,para el caso mejor; la peor permutación, para el caso peor y N! permutaciones para calcular más eficientemente el tiempo medio.

5.4 Pregunta 4

Observamos en las gráficas de tiempo y de operaciones básicas que MergeSort tarda menos que QuickSort, esto es porque MergeSort realiza entre $B_{MS}=(1/2)Nlog(N)$ y $W_{MS}=Nlog(N)+O(N)$ con un tiempo medio de $A_{MS}=\Theta(Nlog(N))$ y QuickSort ya su caso medio es de $A_{QS}=2Nlog(N)+O(N)$ siendo $A_{QS}>W_{MS}$

Por otro lado QuickSort es más eficiente que MergeSort desde el punto de vista de gestión de memoria pues no utiliza memoria auxiliar para guardar partes de la tabla como sí lo hace MergeSort.

5.5 Pregunta 5

Podemos observar en la gráfica 5 QuickSort vs QuickSort_src que ambos algoritmos no muestran ninguna diferencia significante como bien explicamos en ese apartado comparandolas. Aunque observamos algunos picos donde diverge esta igualdad que han podido ser causado por el uso de permutaciones aleatorias o errores causados por el procesador