Nombre del algoritmo	QuickSort
Mejor caso	El pivote está en el centro de la lista. Dividiéndola en dos sublistas de igual tamaño
Complejidad en el mejor caso	O(nlog n)
Peor caso	El pivote es el mayor o menor elemento de la lista.
Complejidad en el peor caso	O(n^2)
Algoritmo inplace	Si
Algoritmo Adaptativo	No
Algoritmo Estable	No
Nombre del algoritmo	ShellSort
Mejor caso	Cuando los datos están casi organizados.
Complejidad en el mejor caso	O(nlog n)
Peor caso	Cuando los datos están organizados de forma aleatorio.
Complejidad en el peor caso	O(n^2)
Algoritmo inplace	Si
Algoritmo Adaptativo	No
Algoritmo Estable	No
Nombre del algoritmo	MergeSort
Mejor caso	No tiene ya que se comporta de igual manera para cualquier tipo de datos.
Complejidad en el mejor caso	O(n log n)
Peor caso	No tiene ya que se comporta de igual manera para cualquier tipo de datos.
Complejidad en el peor caso	O(n log n)
Algoritmo inplace	No
Algoritmo Adaptativo	Si

Algoritmo Estable	Si

	ShellSort(mseg)	MergeSort(mseg)	QuickSort(mseg)
Tiempo Ejecución 1	6,556	5,679	7,30
Tiempo Ejecución 2	6,518	5,154	5,871
Tiempo Ejecución 3	6,177	5,224	6,193
Tiempo Promedio(mseg):	6,417	5,35233333333333	6,45466666666667

Conclusión: Por el tiempo promedio de ejecución, para el caso general, el algoritmo más eficiente es MergeSort. El siguiente algoritmo en eficiencia es ShellSort debido que los resultados obtenidos tienen poca varianza entre ellos. El algoritmo menos eficiente es QuickSort debido a su inestabilidad y la gran varianza entre sus datos.