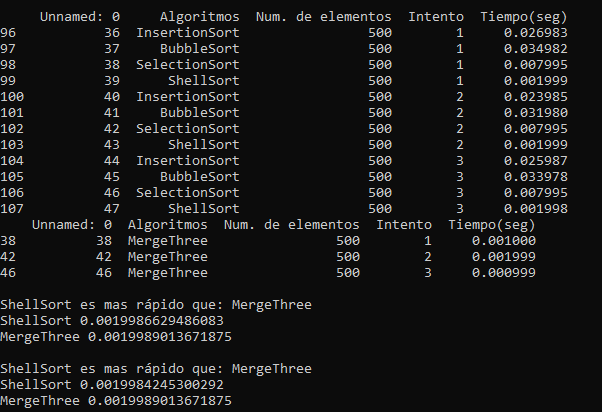
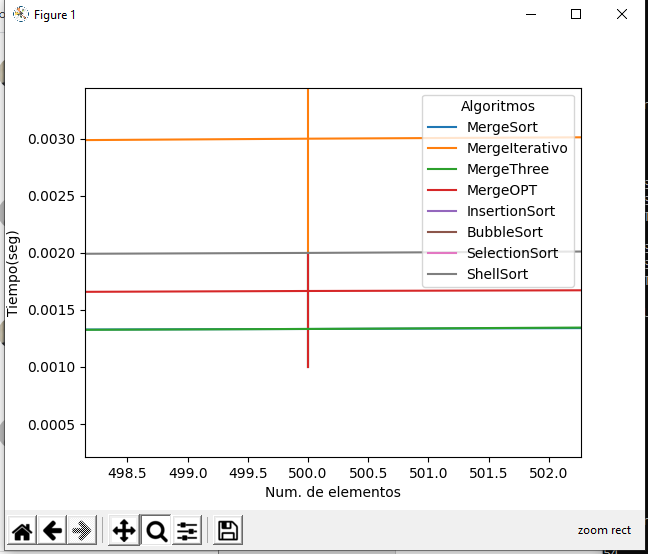
**Reporte: Juan Reyna #10- 10883**

**Actividad 3:** En eta actividad se requiere determinar si hay una secuencia de tamaño **n,** suficientemente pequeño, para el cual alguno de los algoritmos de ordenamientos simples (insertion sort, selection sort, bubble sort,Shell sort) es más rápido que mergesort three. Para determinar este **n**, fue diseñado un programa de comparación llamado: **comparación\_on2\_nlgn.py,** que contrasta los tiempos entre mergesort three y los algoritmos de ordenamiento simple, comparándolo según el tamaño de la secuencia y mostrando su gráfica. El tipo de secuencia que se usa para determinar el tamaño fueron secuencias de tipo sreal y se hicieron tres intentos para hacer la aplicación de dichos algoritmos. Haciendo diversas pruebas el programa **comparación\_on2\_nlgn.py,** determina de manera consistente que para **n = 500** shellsort ordena más rápido las secuencias que se les suministra. Por ser **n = 500** un numero bajo, gráficamente es imperceptible determinar dicho resultado cuando se indican diversos tamaños de secuencias muy grandes.

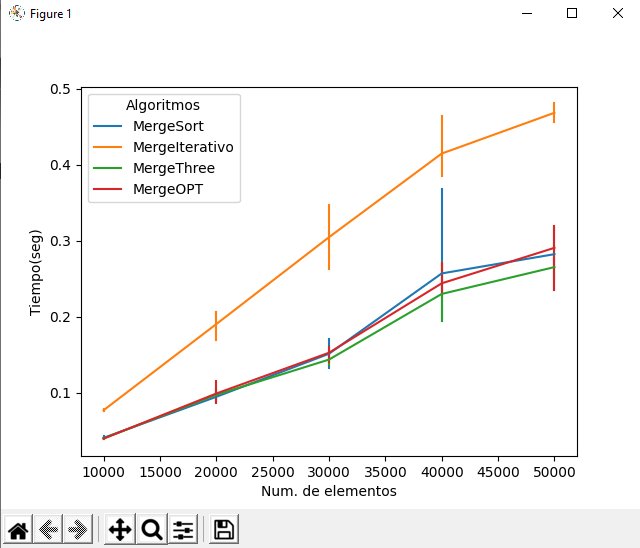


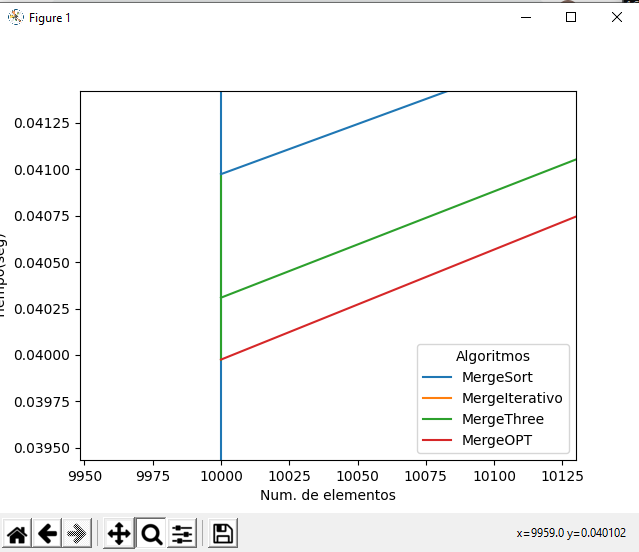
**Figura 1. Comparación**

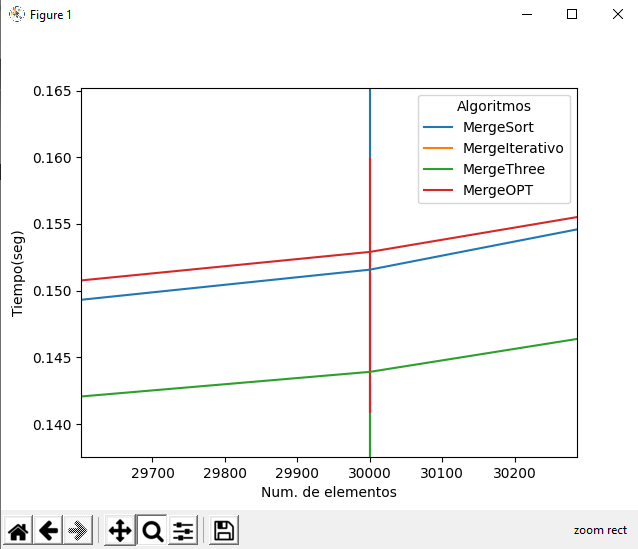


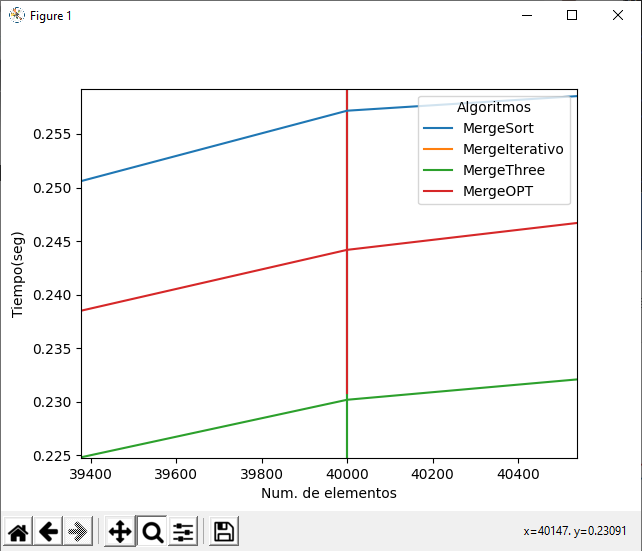
**Figura 2. Gráfica de comparación**

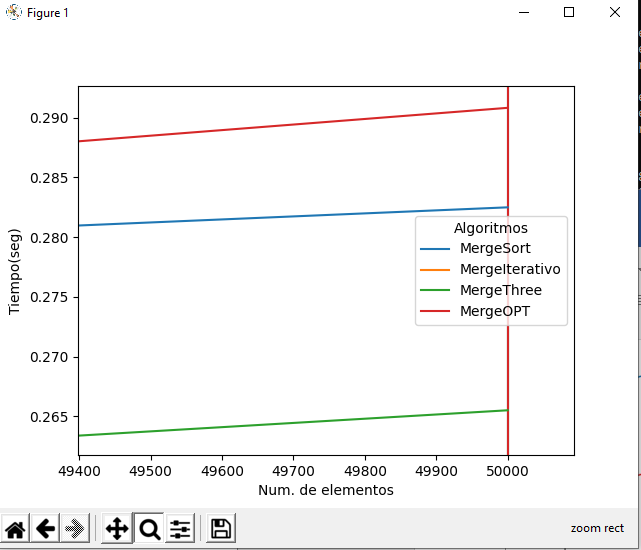
**Actividad 7:** Modificando el programa **comparación\_on2\_nlgn.py**, para analizar el desempeño de mergesortopt nos fijamos que este es mucho más rápido que los todos los algoritmos de ordenamiento para los parámetros exigidos en la **actividad 6.**











 Una ventaja del Shellsort es su eficiencia para secuencias de longitud suficientemente pequeña. Shellsort es el algoritmo más rápido de todos aquellos de complejidad O(n^2), como es el caso del Bubble sort y el Insertion sort. Combinado con merge sort (mergesortopt) es un algoritmo más rápido que los otros involucrados en esta actividad.