Mariana González Velázquez

222966146

Análisis de Algoritmos

Jorge Ernesto López Arce Delgado

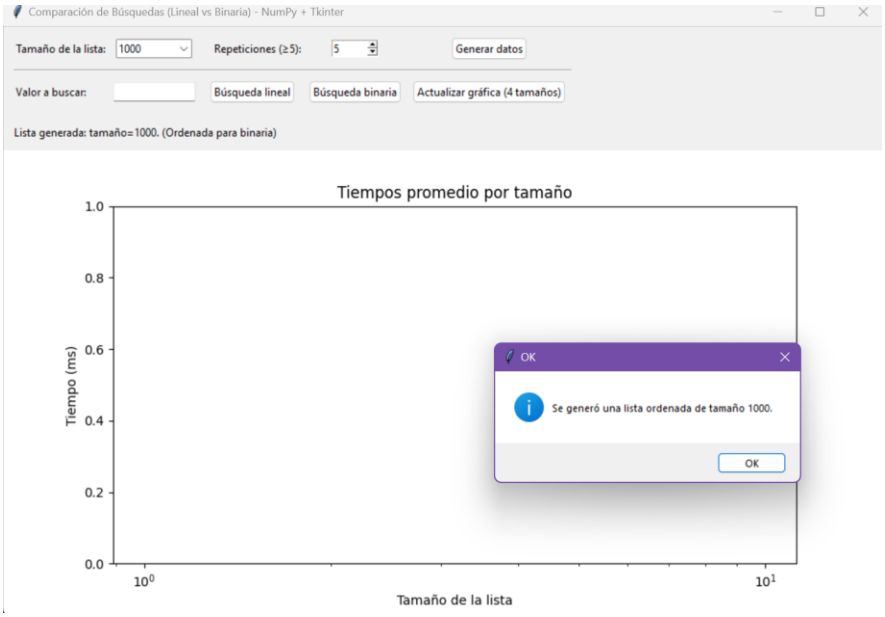
Practica 01

Búsqueda con GUI

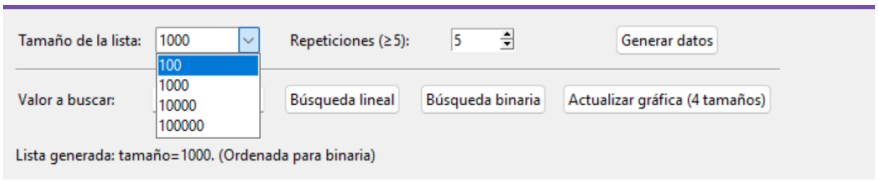
-Descripción

Comparar la eficiencia de la búsqueda lineal y binaria sobre listas de distintos tamaños (100, 1000, 10000, 100000), con repeticiones de <=5

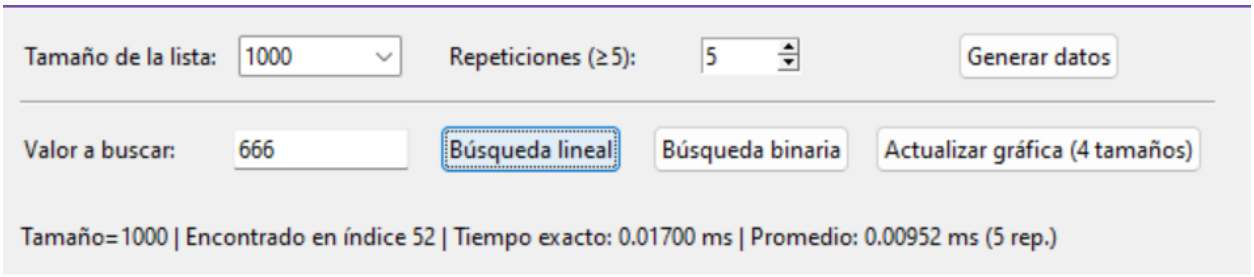
-Lista Generada (tamaño de 1000 y ordenada para la búsqueda binaria)

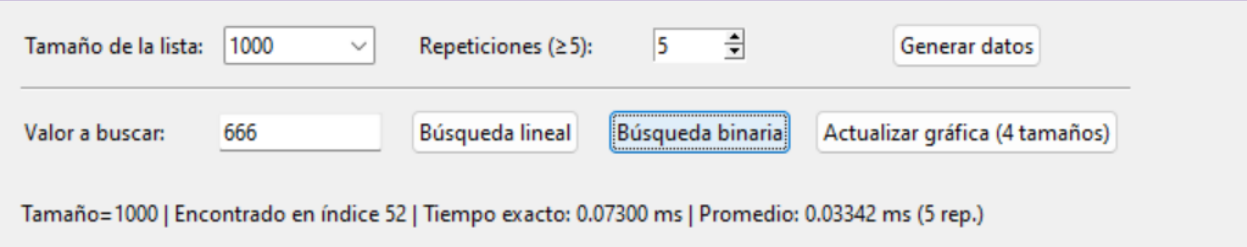


Opciones tamaño para generar la lista

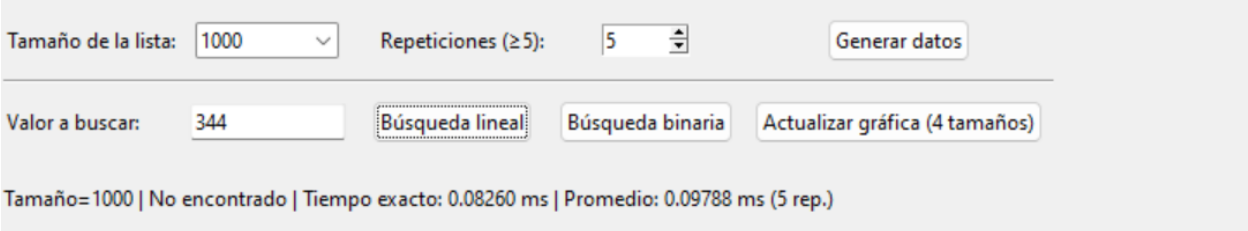


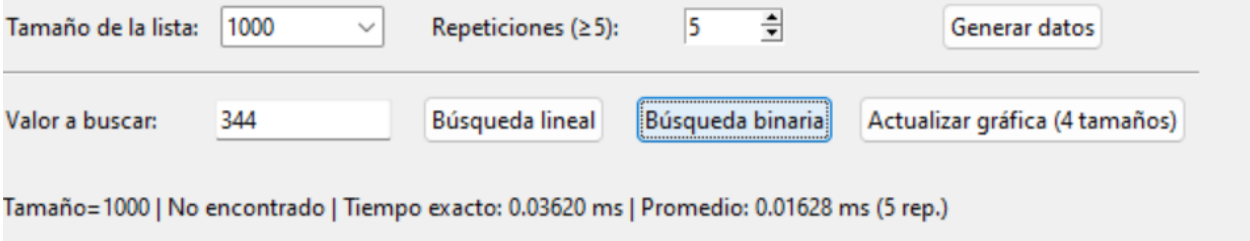
-Búsqueda Lineal y Binaria



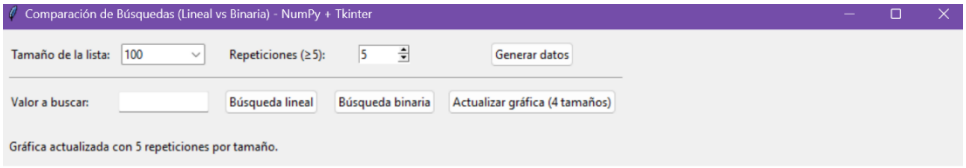


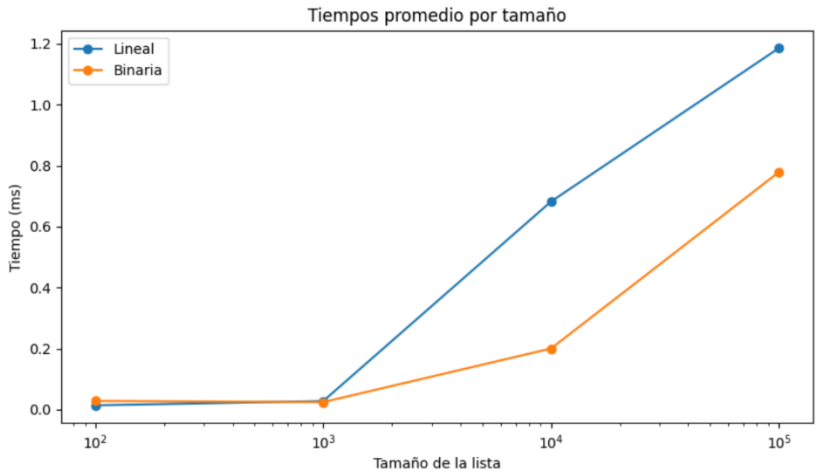
Valor no encontrado





-Gráfica comparativa (todos los tamaños)



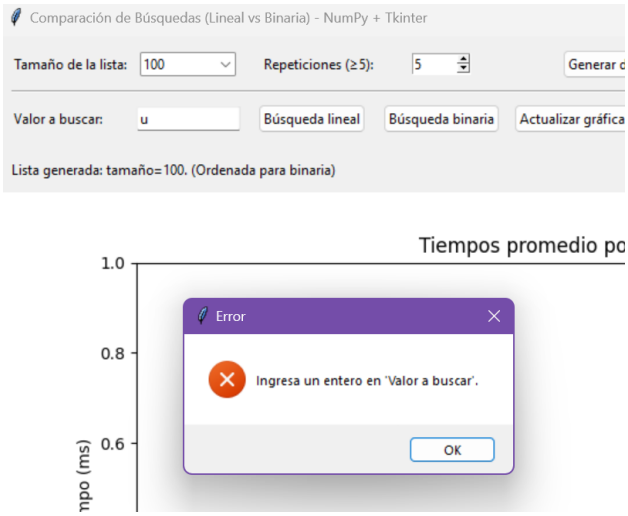


La gráfica refleja que, a medida que la lista aumenta de tamaño, la búsqueda lineal tarda cada vez más, mientras que la binaria se mantiene mucho más rápida. Esto confirma lo que normalmente se explica en teoría sobre la eficiencia de ambos algoritmos

Tamaño de la lista	Búsqueda lineal (ms)	Búsqueda Binaria (ms)
100	0.01	0.03
1000	0.03	0.02
10000	0.68	0.20
100000	1.18	0.78

-Validaciones de la GUI

Sin generar la lista, Manejar entradas inválidas (vacías, no numéricas).



Conclusiones personales

Desarrollando esta práctica pude comprender la relevancia que tiene la eficiencia de los algoritmos al procesar información, comparando la búsqueda lineal con la búsqueda binaria noté que la búsqueda lineal resulta útil en listas pequeñas ya que su rendimiento se reduce de acuerdo al aumento de los datos, por otro lado la búsqueda binaria, aunque requiere que la lista se encuentre ordenada crea un rendimiento mucho más optimo en listas grandes, lo que muestra la importancia de considerar varios factores antes de seleccionar un método de búsqueda. Implementando el último paso que son las validaciones dentro de la interfaz gráfica, pude ver como ciertos detalles evitan que el usuario pueda usar la GUI correctamente, por lo que creo que es óptimo asegurar que el sistema responda adecuadamente ante entradas inválidas o incompletas. En general, esta práctica me permitió no solo reforzar la teoría, sino también comprobar con resultados cómo la elección del algoritmo adecuado influye en el rendimiento y en los resultados, además de conocer algunas funcionalidades extra de ciertas librerías usadas en el programa y comprender mejor cómo aprovecharlas para optimizar el código y su estructura.