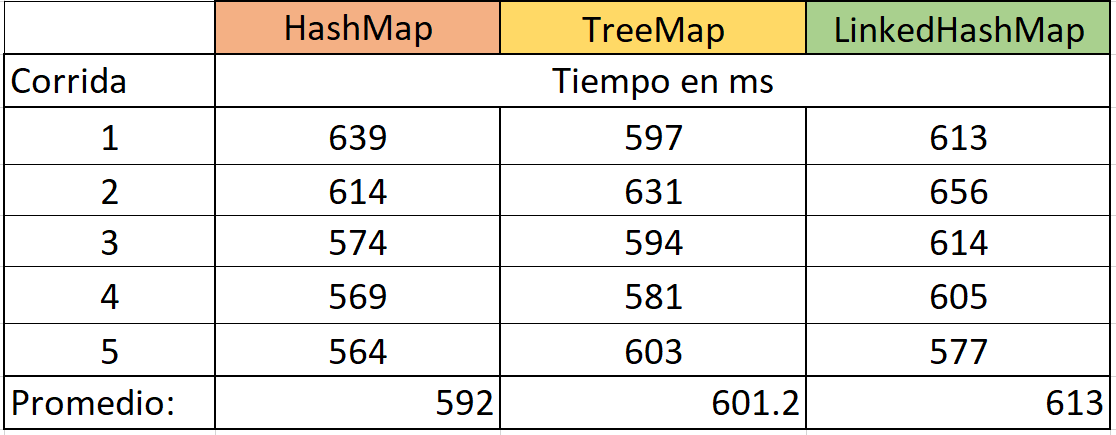
**Hoja de Trabajo #6**

* **Repositorio del proyecto en GitHub:**

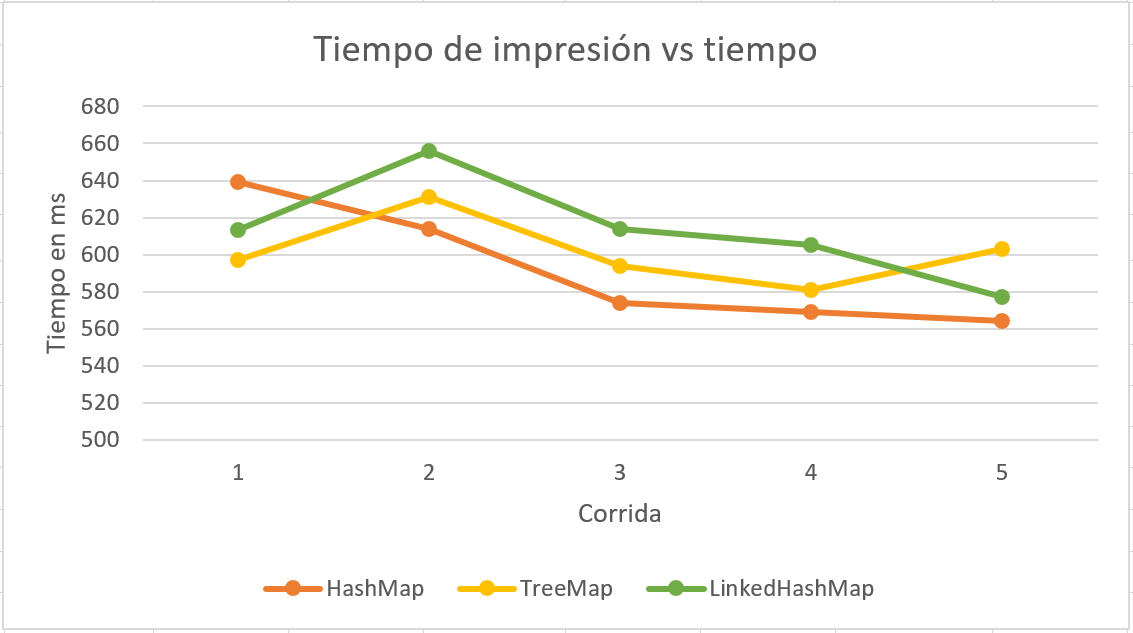
<https://github.com/OJP98/HojaDeTrabajo6>

* **Tablas y gráficas:**

Tabla #1. Tiempos en las diferentes corridas de cada implementación



Gráfica #1. Comparación del rendimiento de cada implementación.

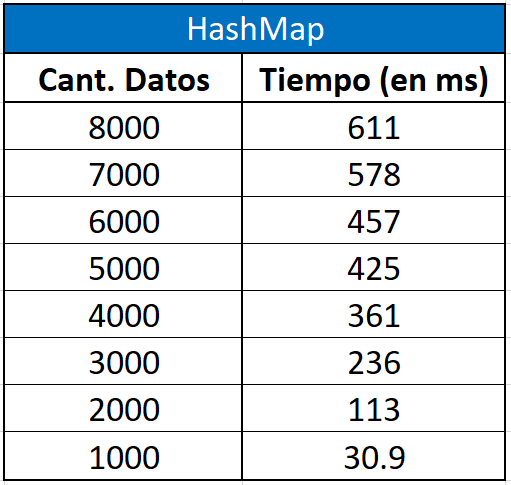


* **Implementación más rápida:**

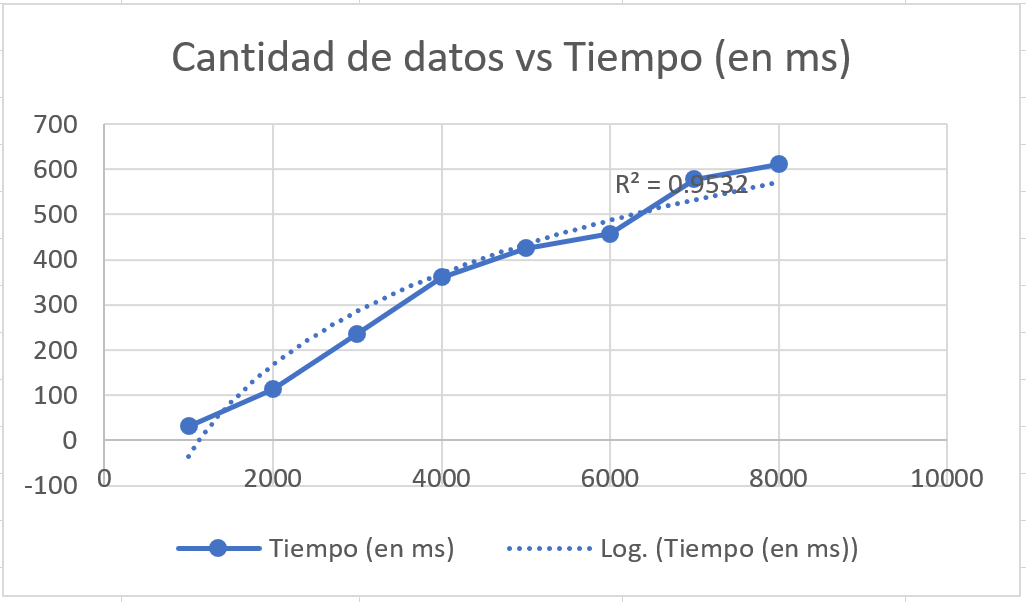
Como se puede apreciar tanto en la tabla #1, se midió 5 veces la velocidad con la que las diferentes implementaciones mostraban todas las cartas existentes. Para cada implementación, se sacó un promedio y el tiempo más pequeño fue dado por la implementación **HashMap, la cual mostró un tiempo promedio de 592 ms.** Estos datos, sin embargo, pueden variar dependiendo de la computadora utilizada o bien de las aplicaciones que corran en el fondo, ya que esto altera los resultados de los tiempos obtenidos. Adicionalmente, la gráfica #1 nos muestra que el tiempo obtenido no es constante, es variable, aleatorio e impredecible, por lo que puede ser posible que el resto de implementaciones tengan un mejor rendimiento que HashMap. A pesar de ser muy cercano a los otros valores obtenidos por el resto de implementaciones, **se puede concluir que HashMap es el más rápido.**

* **Complejidad de HashMap:**

Tabla #2. Tiempo en el que se muestra diferentes cantidades de datos.



Gráfica #2. Rendimiento en cuanto a los datos utilizados.



La complejidad de tiempo para la implementación de HashMap al utilizar el profiler y medir los tiempos para mostrar datos, resultó ser de **O log (n)**. Esto se puede respaldar dado que la regresión lineal de la gráfica #2 es de 0.9532, lo cual nos indica que es muy cercana a una función logarítmica. Ciertos factores como el rendimiento de la computadora y las aplicaciones que se tienen abiertas en el momento, hacen que el coeficiente de correlación no sea exactamente 1, pero sí acercan a dicho valor.

* **Rubrica a calificar:**

