Reporte Tarea 01

Juan José Navarro Coto - B85590

En este trabajo se buscará analizar el uso del tiempo de los ordenamientos de selección, inserción,mezcla, montículos, rápido y por residuos.

I. INTRODUCCIÓN

En este trabajo se busca exponer, comparar y analizar los resultados de los distintos tipos de ordenamientos: selección, inserción, mezcla, montículos, rápido y por residuos. Los ordenamientos, previamente programados, serán puestos a prueba ante distintos tipos de arreglos, estos con números totalmente aleatorios y de dimensiones distintas, esto para obtener resultados lo más acertados posibles.

Con los resultados se buscará analizar el comportamiento de los ordenamientos y su eficiencia enfocada en el tiempo de ejecución de cada uno, respecto al tamaño del arreglo que deben ordenar. Posteriormente se buscará comparar los resultados obtenidos para concluir el ordenamiento más eficiente, utilizando el tiempo como punto de referencia.

II. METODOLOGÍA

Para lograr los objetivos de esta tarea se programaron diversos métodos para lograr calcular el tiempo de ejecución de los distintos tipos de ordenamientos planteados en el enunciado. En este caso los ordenamientos a programar son: ordenamiento de **selección**, ordenamiento de **inserción** y ordenamiento por **mezcla**, ordenamiento por **montículos**, ordenamiento **rápido**, ordenamiento por **residuos**.

Los códigos se basan mayormente en el pseudocódigo del libro de Cormen y los apuntes del profesor, material que se encuentra en mediación virtual.

Para lograr obtener los valores del tiempo se utilizó la función *clock()*, la cual retorna el tiempo de CPU medido en *clock_t*. Con estos valores, restando el tiempo justo antes de iniciar el proceso y el tiempo al finalizar el proceso, se logra obtener el tiempo total que duró el método.

Posteriormente se realizaron una serie de corridas de cada uno de los métodos para así obtener distintos resultados los cuales serán utilizados para estudiar el resultado de cada uno de los distintos ordenamientos. Para obtener diferentes rangos para medir la velocidad de los ordenamientos estos se encargaran de ordenar arreglos de distintos tamaños, siendo estos: **50000**, **100000**, **150000** y **200000** el más grande.

Se adjuntan tablas con los resultados obtenidos tras la corrida de los métodos.

Figura 1

		CORRIDAS					PROM
ALGORITMO	TAMAÑO	1	2	3	4	5	
Selección	**50000**	3.524	3.469	3.484	3.474	3.547	3.499
	100000	14.026	13.850	14.005	14.223	14.289	14.079
	150000	31.265	31.410	31.148	31.199	32.164	31.437
	200000	55.722	55.501	55.947	55.661	56.175	55.801
Inserción	**50000**	2.317	2.351	2.220	2.335	2.317	2.308
	100000	8.499	8.270	8.349	9.784	9.142	8.809
	150000	20.845	20.936	20.838	20.993	22.382	21.199
	200000	36.927	39.401	44.129	41.769	43.059	41.057
Mezcla	**50000**	0.000288	0.000427	0.000383	0.000250	0.000262	0.000322
	100000	0.000670	0.000524	0.000487	0.000529	0.000516	0.000545
	150000	0.001043	0.000729	0.000769	0.000766	0.000800	0.000821
	200000	0.001043	0.001043	0.001043	0.001043	0.000976	0.001043

Figura 2

Montículos	**50000**	0.01598	0.012316	0.017308	0.012767	0.017317	0.0151376
	100000	0.026627	0.025635	0.030446	0.026258	0.029144	0.027622
	150000	0.0403	0.041239	0.04299	0.043474	0.041744	0.0419494
	200000	0.057896	0.054563	0.055883	0.055338	0.055761	0.0558882
Rápido	**50000**	0.011606	0.010189	0.010795	0.0125	0.014241	0.0118662
	100000	0.021726	0.02204	0.022138	0.01857	0.020555	0.0210058
	150000	0.032202	0.030312	0.032365	0.034086	0.03245	0.032283
	200000	0.04119	0.040807	0.041705	0.042227	0.042434	0.0416726
Residuos	**50000**	0.012498	0.016954	0.015896	0.017192	0.012974	0.0151028
	100000	0.028917	0.026865	0.028103	0.029628	0.026719	0.0280464
	150000	0.040691	0.037356	0.042914	0.040133	0.039127	0.0400442
	200000	0.052762	0.054237	0.050673	0.052443	0.049	0.051823

En la tabla se representan los resultados del tiempo obtenido según cada uno de los métodos, esto usando un arreglo aleatorio en cada corrida y haciendo cinco corridas para conseguir un promedio. En este caso la medida para el tiempo son segundos, y el valor de tamaño es la cantidad de elementos del arreglo.

III. RESULTADOS

Observando los tiempos que se muestran en la *figura 1 y figura 2*, se puede analizar como cada uno de los ordenamientos varía según el tamaño, dando así un tiempo de ejecución distinto. Iniciando con el ordenamiento de **selección**, en este caso podemos observar como según el tamaño del arreglo el tiempo de ejecución aumenta de manera considerable, llegando incluso a un promedio de 55.801 segundos en el arreglo de mayor tamaño. Este, de entre todos los ordenamientos, fue el que mantuvo un promedio mayor en todos los tiempos de ejecución, tal como podemos ver en la *figura 10*, la comparación de los arreglos.

Como segundo ordenamiento se tiene el de **inserción**, en este caso podemos observar que su tiempo promedio en segundos en el arreglo de mayor tamaño es de 41.057, teniendo una diferencia notable a la hora de compararlo con el ordenamiento de selección, el cual destaca por tomar mayor tiempo a la hora de su ejecución.

Tenemos el ordenamiento por **mezcla**, los resultados obtenidos nos muestran que es el ordenamiento que menos recurso de timepo utiliza para realizar el proceso en la máquina, dando como promedio en segundos, un promedio de 0.001043, lo cual está muy por debajo de los ordenamientos anteriores, esto nos muestra como el uso de sub arreglos, con la filosofía divide y vencerás muestran un avance a la hora de buscar eficiencia enfocada en el tiempo de ejecución, de igual forma se debe utilizar más memoria como consecuencia a la creación de los subarreglos, cuando por otro lado, los ordenamientos anteriores no generaban más arreglo del ya dado.

En caso del ordenamiento por **montículos** cuenta con la ventaja de ser un ordenamiento relativamente rápido y eficiente, teniendo un promedio de 0.0558882 en el arreglo más extenso, este no es el más veloz, ya que queda superado por

el ordenamiento de mezcla, pero sí denota una clara ventaja sobre los ordenamientos de selección e inserción.

El ordenamiento **rápido** nos muestra una forma curiosa con la que se puede trabajar utilizando pivotes y dividiendo el arreglo en pequeños subarreglos, esto nos muestra su eficiencia, ya que en el arreglo más extenso tiene un promedio de 0.0416726, donde supera al ordenamiento por montículos pero de igual forma queda por debajo del ordenamiento por mezcla.

Por último el ordenamiento por **residuos** a pesar de que su eficiencia no supera al ordenamiento por mezcla ni al ordenamiento rápido, es el siguiente en la lista de eficiencia como se representa en la gráfica *figura 10*, donde se observan las comparaciones, teniendo un promedio de 0.051823 frente al arreglo de mayor tamaño.

Se adjuntan gráficas de los ordenamientos.

Figura 3

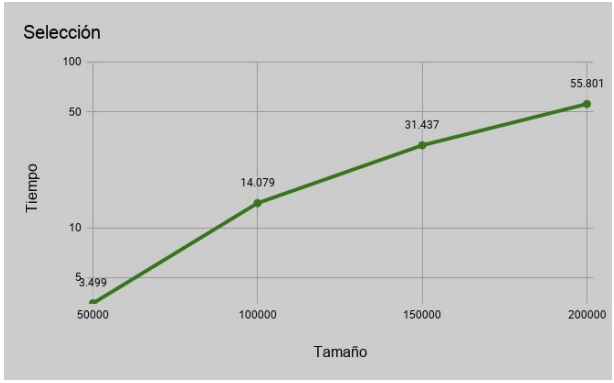


Figura 4

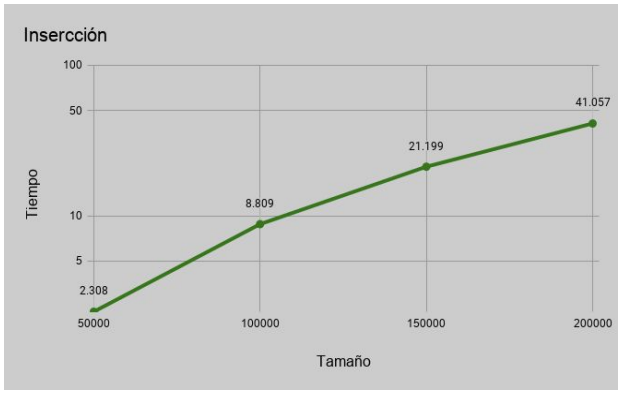


Figura 5

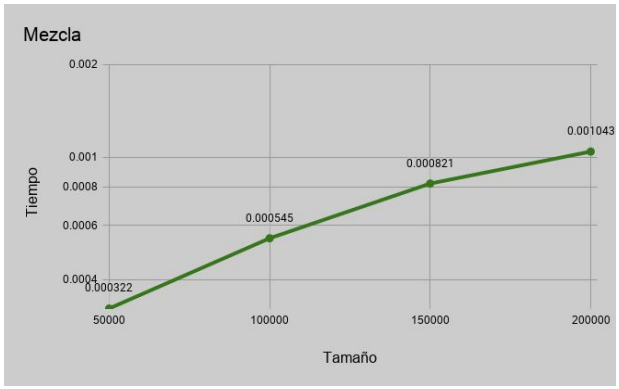


Figura 6

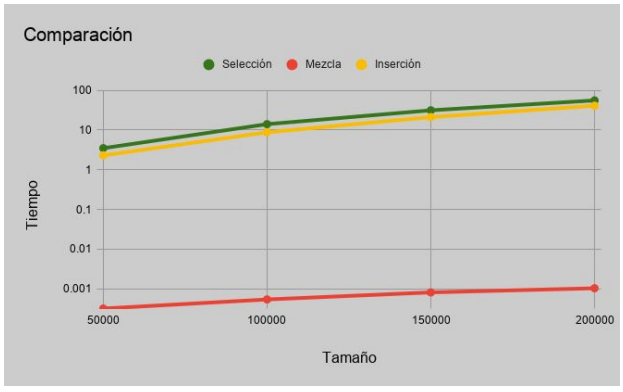


Figura 7

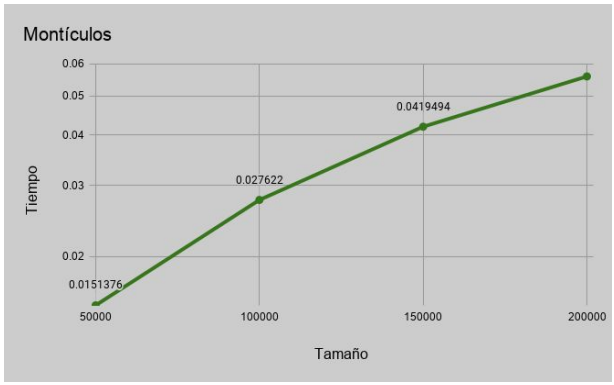


Figura 8

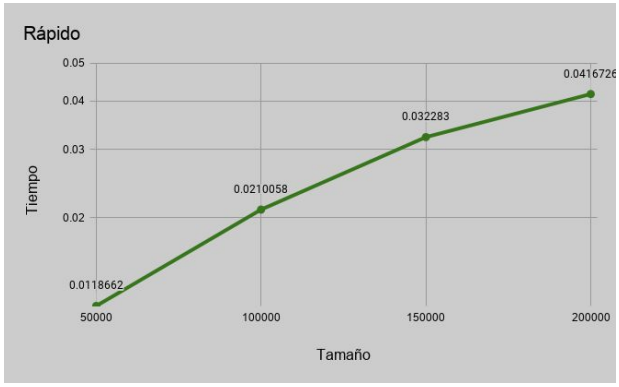


Figura 9

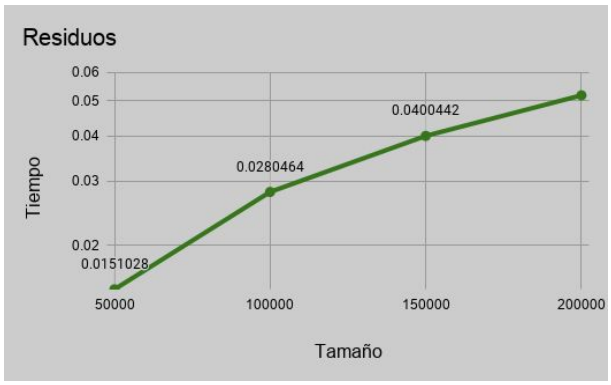
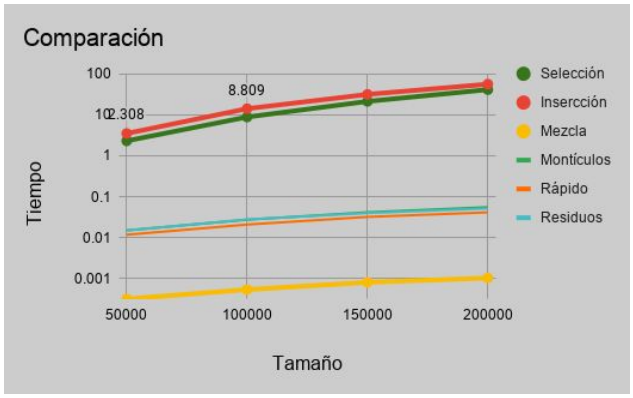


Figura 10



IV. CONCLUSIONES

Partiendo de los resultados previamente comentados se puede concluir que el ordenamiento más eficiente utilizando como referencia el tiempo de ejecución, sin duda alguna es el ordenamiento por mezcla, el cual queda muy por debajo comparado ante el ordenamiento de selección, inserción, por montículos, rápido y por residuos. Se entiende que llega a sacrificar un poco más la parte de memoria al generar nuevos arreglos, pero en el área en la cual se enfoca el problema, la cual es el tiempo, supera con creces a los otros ordenamientos.

El ordenamiento de inserción, supera al ordenamiento de selección, pero no a los demás, supera a este ordenamiento no con una diferencia tan abismal como el del mezcla pero si con una diferencia notoria. Por otro lado los ordenamientos de montículos y residuos llegan a dar una eficiencia de tiempo prácticamente iguales, por este motivo es casi imposible diferenciarlos en la gráfica de comparación.

El ordenamiento rápido aunque sobresale de los ordenamientos por residuo y por montículos, no llega al nivel de eficiencia que tiene el ordenamiento por mezcla. El cual supera de manera notoria a todos los demás ordenamientos, quedando muy por encima en temas de eficiencia por tiempo.

REFERENCIAS

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms*. MIT press.

Juan José Navarro Coto
B85590