Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen

Observaciones Laboratorio 4

Tabla 1

	Maquina 1 (Camilo Molina)	Maquina 2 (Samuel Rozen)	Maquina 3 (Sebastian Martinez)
Procesadores	12th Gen Intel(R) Core(TM) i5-1240P 1.70 GHz	Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz 2.50 GHz	Intel ® Core (™) i7-6700 CPU @ .340GHz
Memoria RAM (GB)	16 GB	8 GB	8GB
Sistema Operativo	Windows 11 Home Single Language 64-bit operating system, x64-based processor	Windows 11 Home Single Language 64-bit operating system, x64-based processor	Windows 10 Versión 22H2 (SO 19045.3324)

Toma de datos computador 1:

Tabla 2

Vacíos = 10 minutos (En este momento se para la prueba debido a exceso de tiempo)

Porcentaje de la	Tamaño de la muestra	Insertion	Selection	Shell Sort

Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen

muestra [pct]	(ARRAY_LIST)	Sort [ms]	Sort [ms]	[ms]
5.00%	3463	24050,74	44323,62	183,13
20.00%	12364	312993,62	564424,12	4767,52
30.00%	17486			6178,59
50.00%	22381			10940,88
100.00%	44762			22193,50

Tabla 3

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]
5.00%	3463	613,04		4454,74
20.00%	12364			270755,49
30.00%	17486			
50.00%	22381			
100.00%	44762			

Toma de datos computador 2

Tabla 2

Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAY_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]
5.00%	3463	46079,14	71401.94	783.28
20.00%	12364	547493.05		3681.74
30.00%	17486			5440.98
50.00%	22381			8950.43
100.00%	44762			20518.39

Tabla 3

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]

Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen

5.00%	3463	No corre	No corre	23875.76
20.00%	12364			No corre
30.00%	17486			
50.00%	22381			
100.00%	44762			

Toma de datos computador 3

Tabla 2

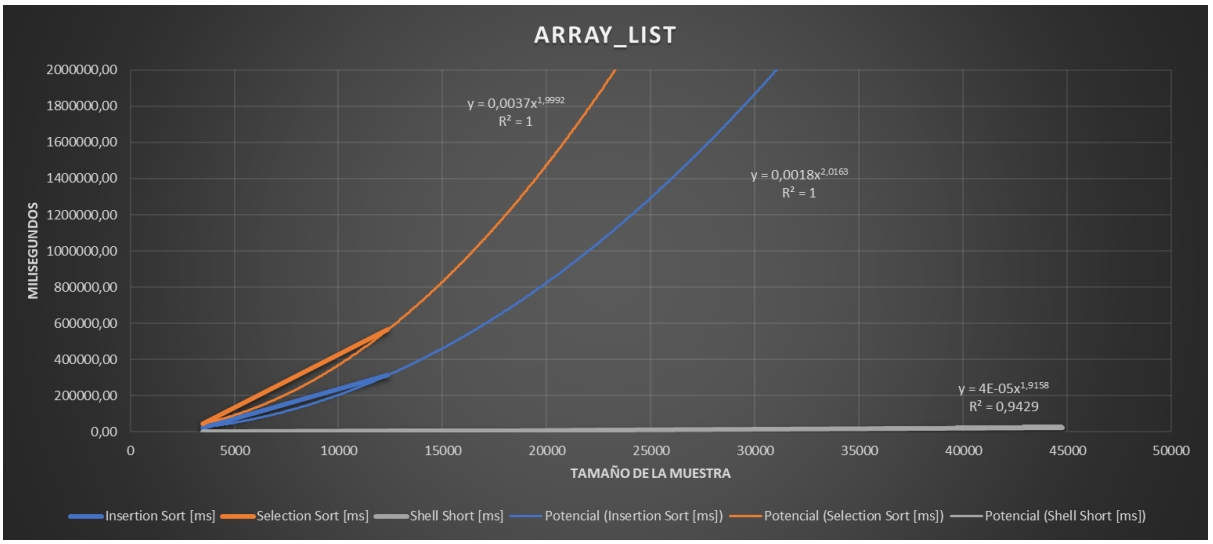
Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAY_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]
5.00%	3463	148665.77	298768.41	2153.4
20.00%	12364	Pasa de 5m	Pasa de 5m	14666.1
30.00%	17486	"	"	21038.13
50.00%	22381	"	"	34301.14
100.00%	44762	"	"	69996.23

Tabla 3

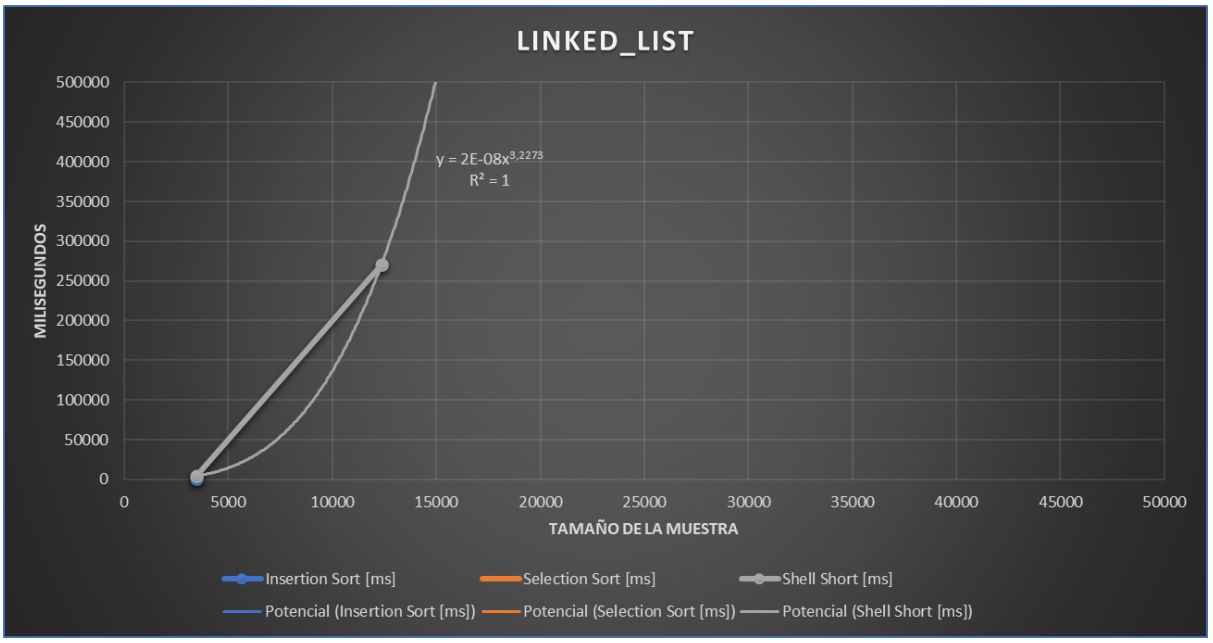
Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]
5.00%	3463			116042.06
20.00%	12364			
30.00%	17486			
50.00%	22381			
100.00%	44762			

Gráfica ARRAY_LIST (1)

Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen

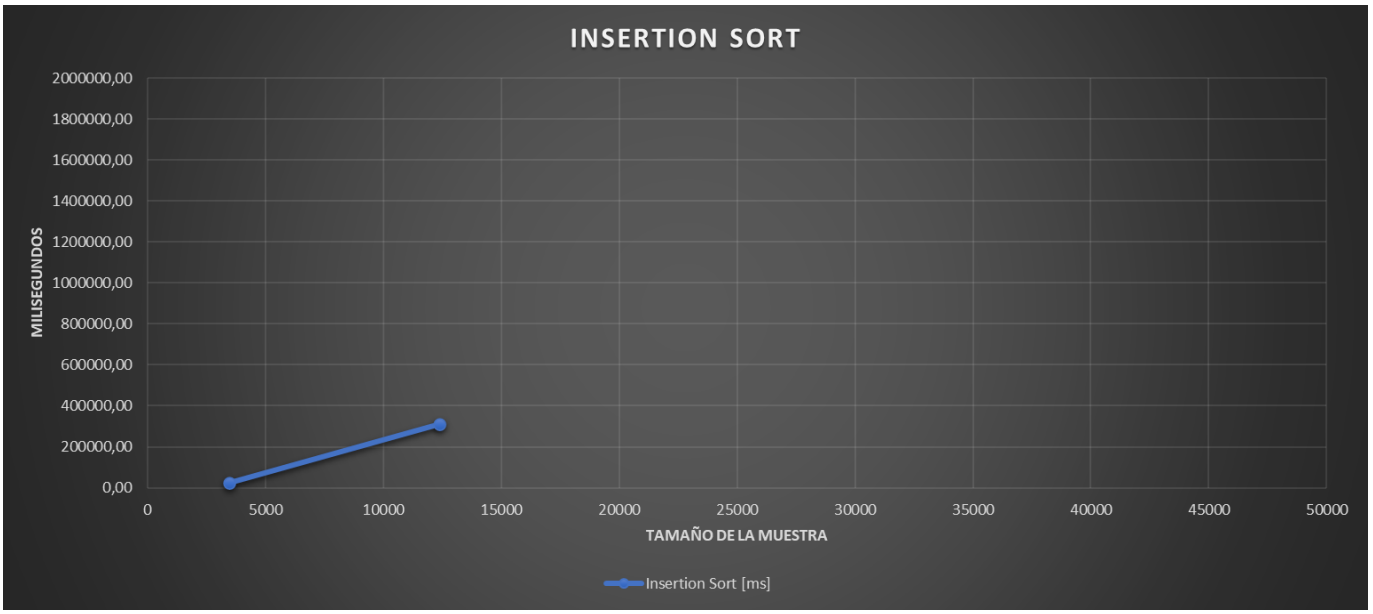


Gráfica LINKED_LIST (1)

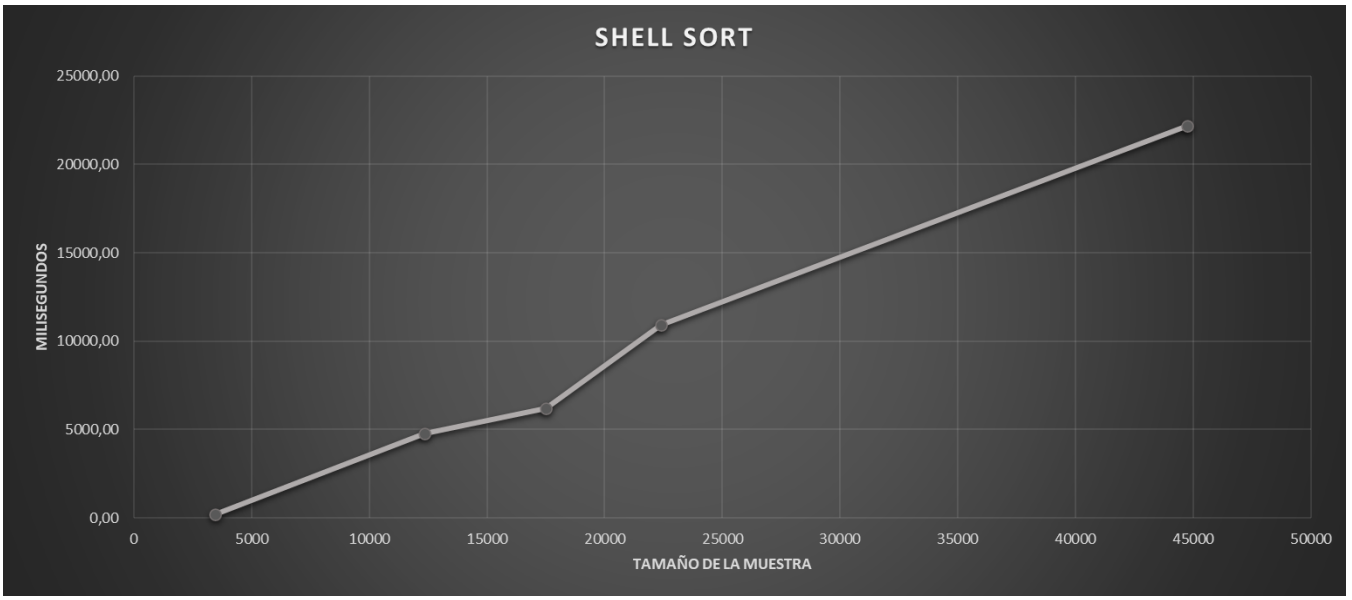


Gráficas individuales de tipo de ordenamiento, teniendo en cuenta los datos de ARRAY_LIST (1)

Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen



Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen



Preguntas:

a. ¿El comportamiento de los algoritmos es acorde a lo enunciado teóricamente?

Insertion Sort: Teóricamente, el algoritmo de ordenamiento por inserción tiene una complejidad temporal de $O(N^2)$ en el peor caso. Observando los datos, vemos que el tiempo de ejecución aumenta cuadráticamente con el tamaño de la muestra, especialmente en el caso de ARRAY_LIST. Esto es consistente con la teoría.

Selection Sort: Al igual que Insertion Sort, Selection Sort tiene una complejidad temporal de $O(N^2)$ en el peor caso. Los datos muestran un comportamiento similar al de Insertion Sort, con tiempos de ejecución que aumentan cuadráticamente con el tamaño de la muestra en ARRAY_LIST.

Shell Sort: Shell Sort es un poco más complicado de analizar porque su tiempo de ejecución depende de la secuencia de intervalos utilizada. Sin embargo, en general, es más rápido que $O(N^2)$. Los datos muestran que Shell Sort es significativamente más rápido que los otros dos algoritmos, lo cual es consistente con la teoría.

b. ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?

Las máquinas con mejores RAM y CPU en general tuvieron mejor rendimiento (menos tiempo) al momento de ejecutar las pruebas.

- Procesador (CPU):
    - Velocidad y Eficiencia: Un CPU más rápido y moderno puede procesar instrucciones a una velocidad mayor, lo que significa que puede completar tareas y cálculos en menos tiempo.

Camilo Molina Plata - 202221119
Sebastian Martinez Arias - 202312210
Samuel Rozen

- Memoria RAM:
    - Acceso Rápido a Datos: La RAM permite un acceso rápido a los datos. Un computador con más RAM puede manejar conjuntos de datos más grandes en memoria sin tener que recurrir al almacenamiento secundario (como un disco duro), que es significativamente más lento.

c. De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?

Hardware: Diferentes máquinas pueden tener diferentes capacidades de hardware, como velocidad del procesador, cantidad de RAM, etc., lo que puede afectar el rendimiento de los algoritmos.

Software: Las diferencias en el sistema operativo, el compilador o el intérprete utilizado, y otros programas en ejecución pueden influir en el rendimiento.

Optimizaciones: Algunas máquinas pueden tener optimizaciones específicas que aceleren ciertas operaciones, mientras que otras no.

d. ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?

Tabla 4

Algoritmo	ARRAY_LIST [ms]	LINKED_LIST [ms]
Insertion Sort	**427408,87**	480122,61
Selection Sort	**4817,49**	600000,00
Shell Sort	**8852,72**	415042,05

(Para los valores vacíos se tuvieron en cuenta un valor de 600,000 ms al calcular el promedio de tiempo, ya que este es equivalente a un tiempo máximo de 10 minutos antes de que se tenga que parar la prueba)

Basándonos en los tiempos promedio de ejecución de los algoritmos, ARRAY_LIST es la estructura de datos más eficiente para los tres algoritmos en comparación con LINKED_LIST. Por lo tanto, si solo se tiene en cuenta los tiempos de ejecución, es mejor utilizar ARRAY_LIST.