UNIVERSIDAD SIMÓN BOLÍVAR
LABORATORIO DE ALGORITMOS II
PROF: GUILLERMO PALMA
ALUMNOS: **SABÁS GONZÁLEZ** **CARNET No: 15-10625**
MIGUEL PÉREZ **CARNET No: 15-11126**

INFORME PROYECTO 1

Introducción

El objetivo del proyecto es el de hacer un estudio experimental de algoritmos de ordenamientos, simples y de alto rendimiento, sobre diferentes tipos de secuencias. La idea es comparar el rendimiento de los algoritmos de ordenamiento vistos en los libros de texto, junto con otras versiones que son el estado del arte y que son ampliamente usados en la práctica.

Los Algoritmos a estudiar

- Mergesort
- Quicksort Iterativo
- Quicksort simple
- Median-of-3 Quicksort
- Introsort
- Quicksort with 3-way partitioning
- Dual pivot Quicksort
- Timsort

Las Pruebas a realizar

- Arreglo Punto Flotante
- Arreglo de números enteros ordenados
- Arreglo de números enteros ordenados descendentemente
- Arreglos de ceros y unos
- Arreglo cuyos elementos van de 1 a N/2 y luego de N/2 a 1
- Arreglo casi ordenado donde se intercambian entre 16 pares de elementos al azar
- Arreglo casi ordenado donde se intercambian entre si N/4 pares de elementos al azar

La Máquina donde se correrán las mismas

- SO = **Ubuntu 19.10**
- CPU = **Intel i3**
- RAM = **8GB**

Resultados

A continuación, se presentan los resultados obtenidos. Cada prueba fue aplicada un número de 3 veces sobre los 8 algoritmos, luego se calculó el promedio de tiempo que tardaron en cada corrida y se creó una gráfica y tabla con estos datos.

Figura 1.1 – Tabla 1

PRUEBA 1: **PUNTO FLOTANTE**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	0,03s	0,02s	0,02s	0,40s	0,02s	0,01s	0,02s
8192	0,03s	0,06s	0,04s	0,04s	1,26s	0,03s	0,03s	0,04s
16384	0,05s	0,12s	0,08s	0,07s	9,06s	0,07s	0,06s	0,09s
32768	0,10s	0,26s	0,16s	0,16s	68,50s	0,15s	0,13s	0,18s
65536	0,21s	0,55s	0,39s	0,37s	84,87s	0,34s	0,29s	0,39s

Figura 1.2 – Grafico Cuadráticas

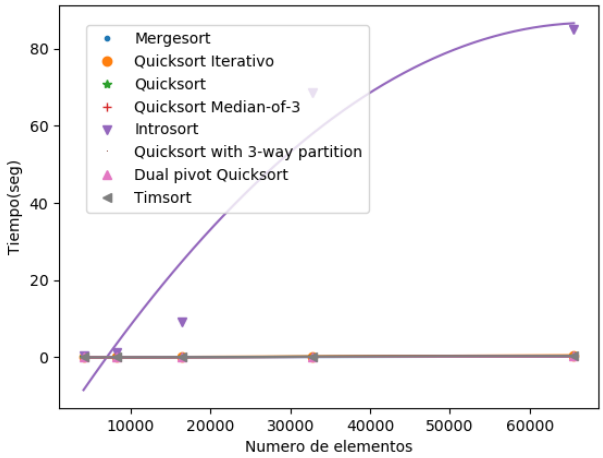


Figura 1.3 – Grafico Lineales

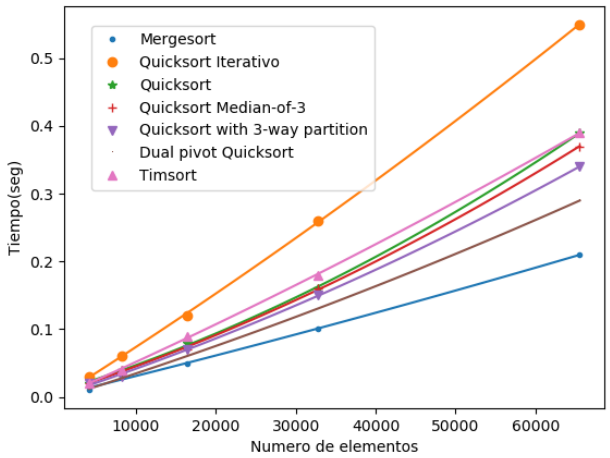


Figura 2.1 – Tabla 2

PRUEBA 2: **ORDENADO**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	0,02s	2,45s	0,00s	0,01s	0,20s	0,96s	0,01s
8192	0,01s	0,05s	9,95s	0,01s	0,01s	0,56s	3,97s	0,02s
16384	0,03s	0,11s	49,89s	0,03s	0,03s	3,19s	20,77s	0,07s
32768	0,09s	0,33s	194,35s	0,07s	0,06s	8,42s	79,41s	0,15s
65536	0,15s	0,51s	736,19s	0,15s	0,18s	26,56s	279,01s	0,34s

Figura 2.2 – Grafico Cuadráticas

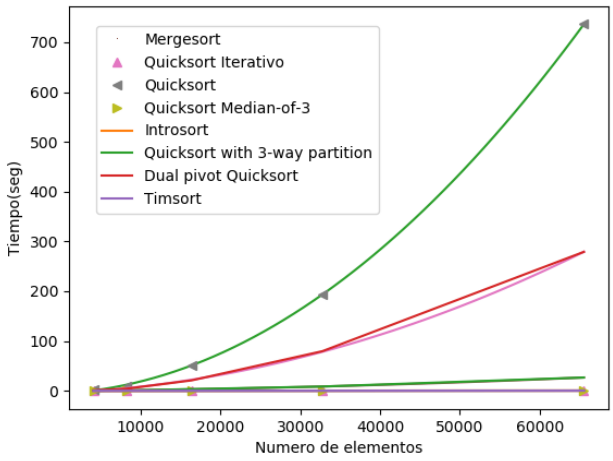


Figura 2.3 – Grafico Lineales

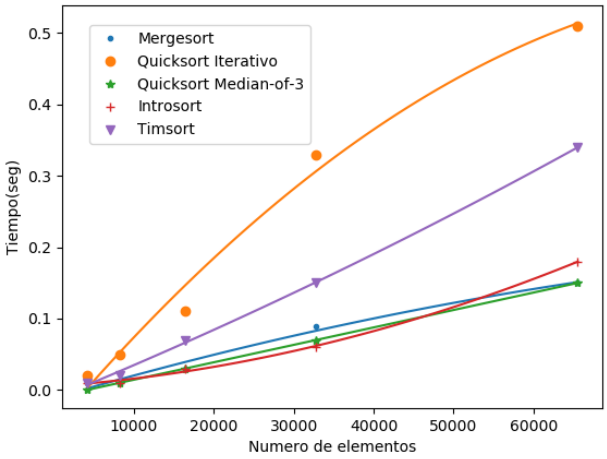


Figura 3.1 – Tabla 3

PRUEBA 3: **ORDEN INVERSO**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,02s	0,02s	1,04s	0,00s	0,04s	0,54s	0,01s	0,02s
8192	0,05s	0,04s	4,73s	0,01s	0,09s	2,43s	3,77s	0,06s
16384	0,09s	0,11s	18,88s	0,03s	0,20s	10,81s	13,03s	0,12s
32768	0,17s	0,21s	77,38s	0,05s	0,37s	38,01s	50,04s	0,2s
65536	0,31s	0,41s	277,91s	0,10s	0,77s	144,83s	187,37s	0,43s

Figura 3.2 – Grafico Cuadráticas

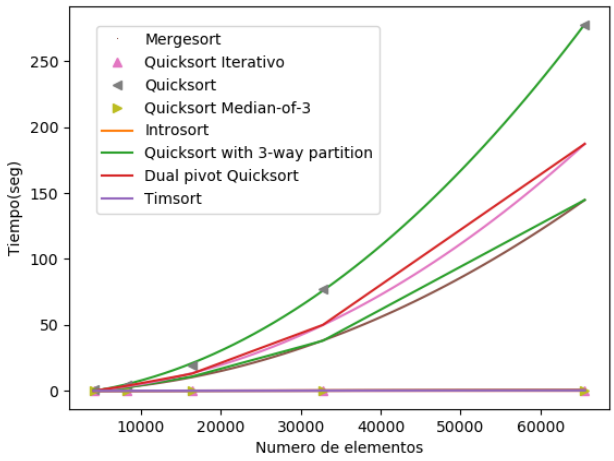


Figura 3.3 – Grafico Lineales

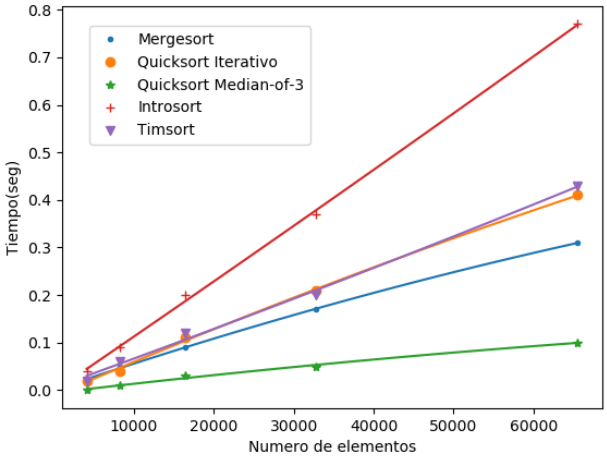


Figura 4.1 – Tabla 4

PRUEBA 4: **CERO-UNO**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	0,00s	1,10s	0,01s	0,01s	0,00s	0,01s	0,01s
8192	0,02s	0,00s	4,33s	0,02s	0,02s	0,01s	0,02s	0,03s
16384	0,04s	0,01s	17,62s	0,04s	0,04s	0,01s	0,05s	0,06s
32768	0,08s	0,02s	72,78s	0,08s	0,11s	0,03s	0,11s	0,14s
65536	0,15s	0,03s	295,98s	0,18s	0,22s	0,06s	0,23s	0,28s

Figura 4.2 – Grafico Cuadráticas

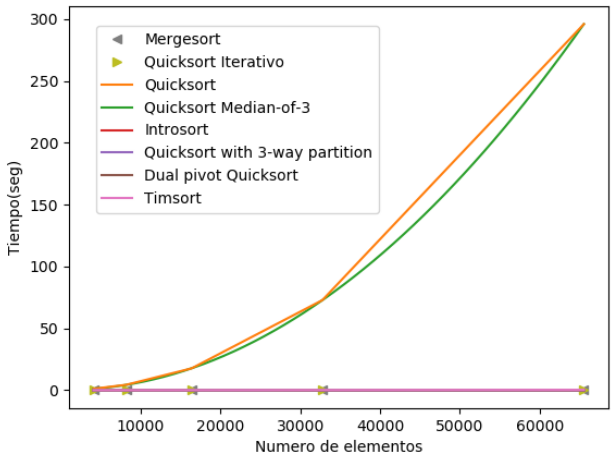


Figura 4.3 – Grafico Lineales

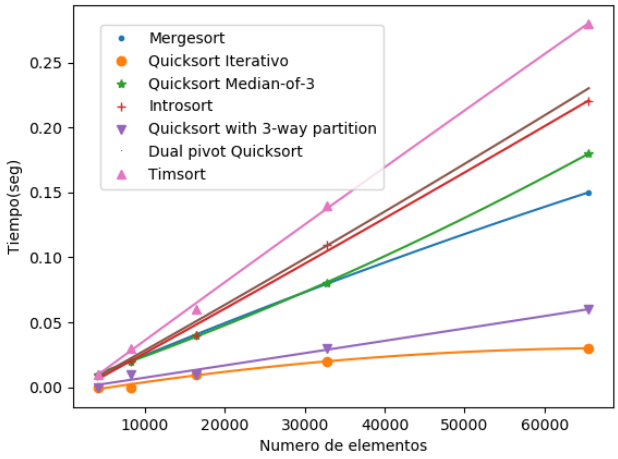


Figura 5.1 – Tabla 5

PRUEBA 5: **MITAD**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	1,46s	0,38s	0,02s	0,06s	0,02s	0,03s	0,02s
8192	0,02s	5,70s	1,51s	0,06s	0,12s	0,05s	0,06s	0,03s
16384	0,05s	24,89s	6,24s	0,14s	0,28s	0,13s	0,16s	0,07s
32768	0,11s	100,62s	24,81s	0,31s	0,58s	0,27s	0,34s	0,16s
65536	0,22s	288,97s	98,90s	0,68s	1,22s	0,59s	0,69s	0,34s

Figura 5.2 – Grafico Cuadráticas

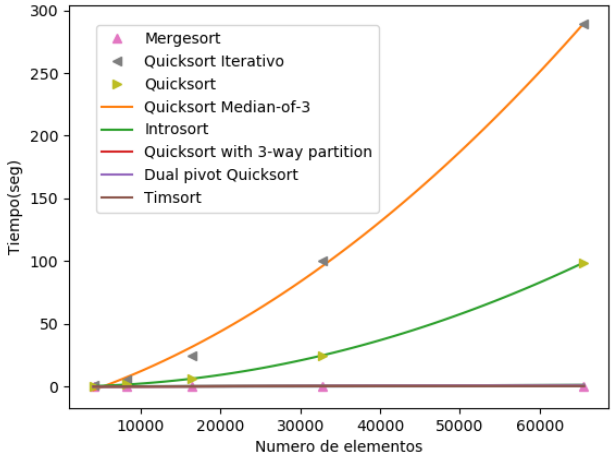


Figura 5.3 – Grafico Lineales

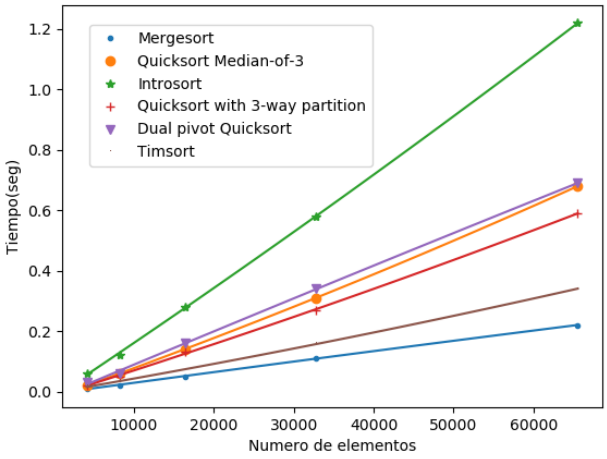


Figura 6.1 – Tabla 6

PRUEBA 6: **CASI ORDENADO 1**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	0,02s	2,14s	0,00s	0,00s	0,49s	0,61s	0,01s
8192	0,01s	0,04s	8,83s	0,01s	0,01s	2,01s	2,53s	0,02s
16384	0,03s	0,09s	37,16s	0,02s	0,02s	8,33s	10,43s	0,04s
32768	0,05s	0,19s	156,46s	0,05s	0,05s	34,65s	41,97s	0,10s
65536	0,13s	0,44s	646,63s	0,10s	0,10s	148,77s	185,83s	0,23s

Figura 6.2 – Grafico Cuadráticas

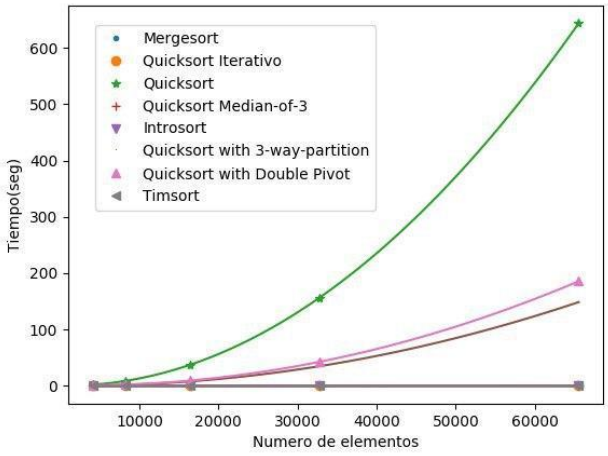


Figura 6.3 – Grafico Lineales

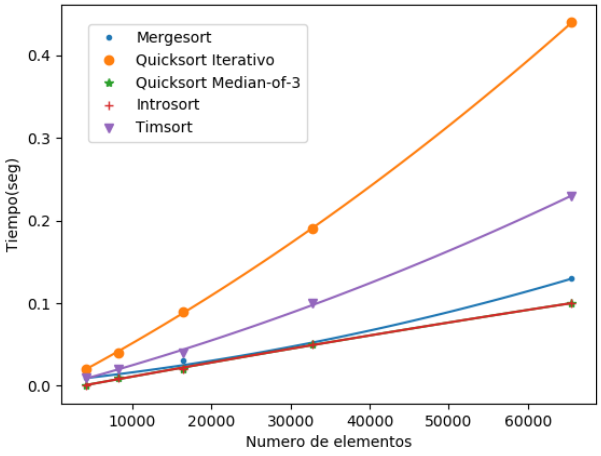


Figura 7.1 – Tabla 7

PRUEBA 7: **CASI ORDENADO 2**								
N	Mergesort	QS Iter	Qs Simple	Med-of-3	Introsort	3-Way	DP	Timsort
4096	0,01s	0,02s	1,18s	0,01s	0,01s	0,32s	0,36s	0,01s
8192	0,02s	0,04s	4,78s	0,01s	0,01s	1,30s	1,46s	0,02s
16384	0,03s	0,09s	19,88s	0,03s	0,03s	5,79s	6,18s	0,05s
32768	0,07s	0,21s	82,25s	0,05s	0,05s	21,66s	24,81s	0,11s
65536	0,15s	0,45s	358,22s	0,12s	0,12s	96,57s	117,37s	0,26s

Figura 7.2 – Grafico Cuadráticas

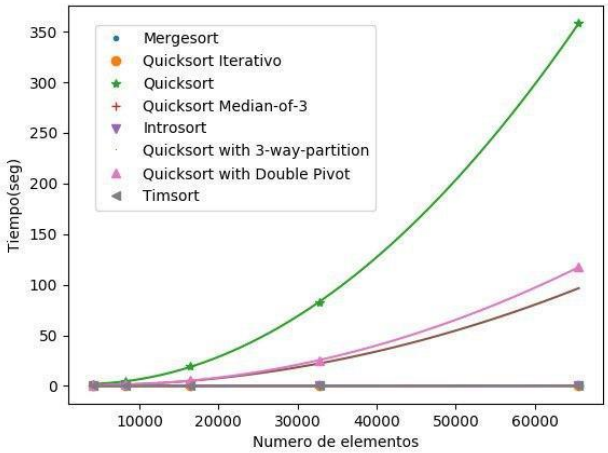
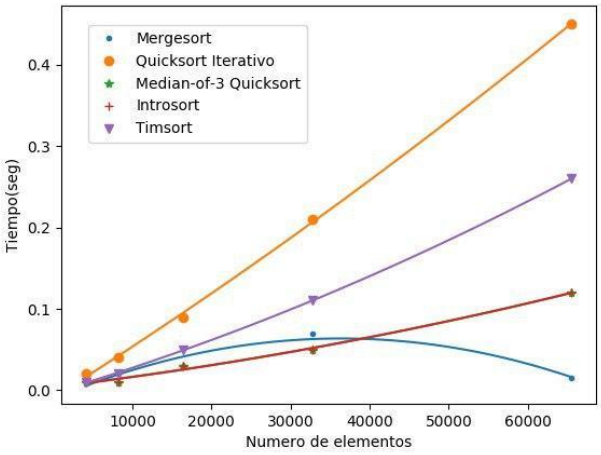


Figura 7.3 – Grafico Lineales



Análisis de resultados

- **Prueba 1***:* En esta prueba el algoritmo con el mejor desempeño fue Mergesort y el de peor desempeño fue Introsort. El rendimiento obtenido se corresponde con el esperado para estos algoritmos excepto para el caso del Introsort el cual a pesar de que su peor caso debería ser de tiempo nlog(n), adquiere una curva de crecimiento mucho mayor.

- **Prueba 2***:* En esta prueba el algoritmo con el mejor desempeño fue Mergesort y el de peor desempeño fue Quicksort Simple. El rendimiento obtenido es el esperado para estos algoritmos y tal como su peor caso, el Quicksort simple es el que tarda mucho más debido a que debe atravesar cada elemento del arreglo hasta el final.

- **Prueba 3***:* En esta prueba el algoritmo con el mejor desempeño fue Quicksort Median-of-3 y el de peor desempeño fue Quicksort Simple. Como en el anterior se esperaba que Quicksort Simple sufriera una gran ralentización en su rendimiento y además esta prueba es perfecta para las otras implementaciones de QS las cuales realizan cambios a la forma de pivotear o comparar elementos para evitar los atascos que sufre la versión simple, permitiendo que en este caso Quicksort Median-of-3 sea el más veloz de todos.

- **Prueba 4***:* En esta prueba el algoritmo con el mejor desempeño fue Quicksort Iterativo y el de peor desempeño fue Quicksort Simple. De igual forma que en la prueba anterior, el hecho de que solo haya dos elementos posibles en estos arreglos es un obstáculo inmensio en la vía del Quicksort Simple pero que es sorteado por sus otras implementaciones más avanzadas.

- **Prueba 5***:* En esta prueba el algoritmo con el mejor desempeño fue Mergesort y el de peor desempeño fue Quicksort Iterativo. Tanto Quicksort Simple como iterativo demstraron sus casos de peor tiempo en este tipo de prueba.

- **Prueba 6***:* En esta prueba los algoritmos con mejor desempeño fueron Quicksort Median-of-3 e Introsort y el de peor desempeño fue Quicksort Simple. Salvo por el Quicksort Iterativo y Median-of-3, todas sus implementaciones fallan en esta prueba que a pesar de ser manejada de forma eficiente de manera ordenada en la prueba 2, logra plantar los peores casos de las implementaciones de QS al realizar los intercambios de pares que realiza.

- **Prueba 7***:* En esta prueba los algoritmos con mejor desempeño fueron Quicksort Median-of-3 e Introsort y el de peor desempeño fue Quicksort Simple. Mismo planteamiento y razonamiento que la prueba anterior.