

## Laboratorio Nro.2

### Notación O Grande

**Duvan Andres Ramírez Saavedra**  
Universidad Eafit  
Medellín, Colombia  
daramirezs@eafit.edu.co

**Santiago Santacruz Ramirez**  
Universidad Eafit  
Medellín, Colombia  
ssantacrur@eafit.edu.co

### 3) Simulacro de preguntas de sustentación de Proyectos

**3.1** De acuerdo a lo realizado en el numeral 1, construya una tabla donde muestre, para cada algoritmo, cuánto se demora para cada uno de los 20 valores del tamaño del problema.

Valores de entrada	Tiempo Insert Sort	Tiempo Merge Sort
519449	1	203
576369	1	229
584844	1	247
626248	1	248
626630	1	278
719341	1	359
1087852	2	454
1126242	2	463
1208764	2	506
1231999	2	506
1305746	2	532
1311201	2	543
1386920	2	576
1395518	2	613
1430194	3	632
1477134	3	642
1558975	3	655
1562193	3	657
1564430	11	665
1582730	15	760

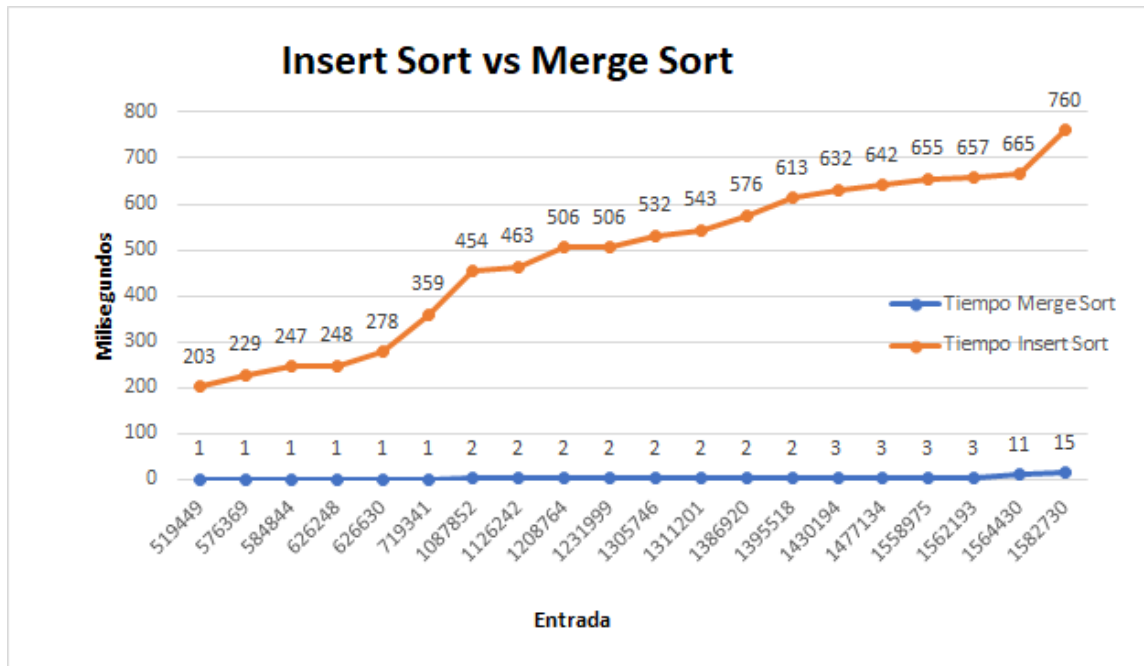
**PhD. Mauricio Toro Bermúdez**

Docente | Escuela de Ingeniería | Informática y Sistemas  
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627  
Tel: (+57) (4) 261 95 00 Ext. 9473

## ESTRUCTURA DE DATOS 1

### Código ST0245

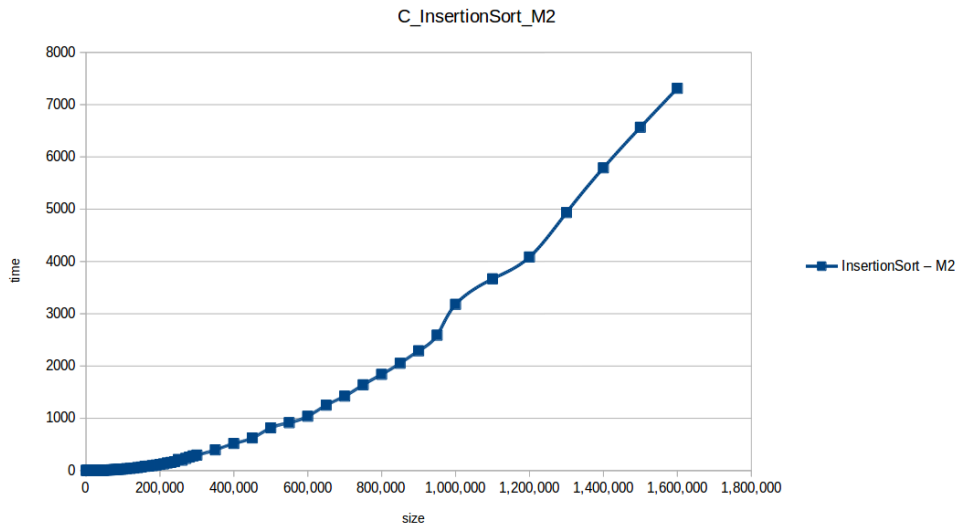
**3.2** Grafiquen los tiempos que tomó en cada algoritmo para los diferentes tamaños del problema. Grafiquen el Tamaño de la Entrada Vs. Tiempo de Ejecución.



**3.3** Teniendo en cuenta lo anterior, ¿Qué tan eficiente es merge sort con respecto a insertion sort para arreglos grandes?

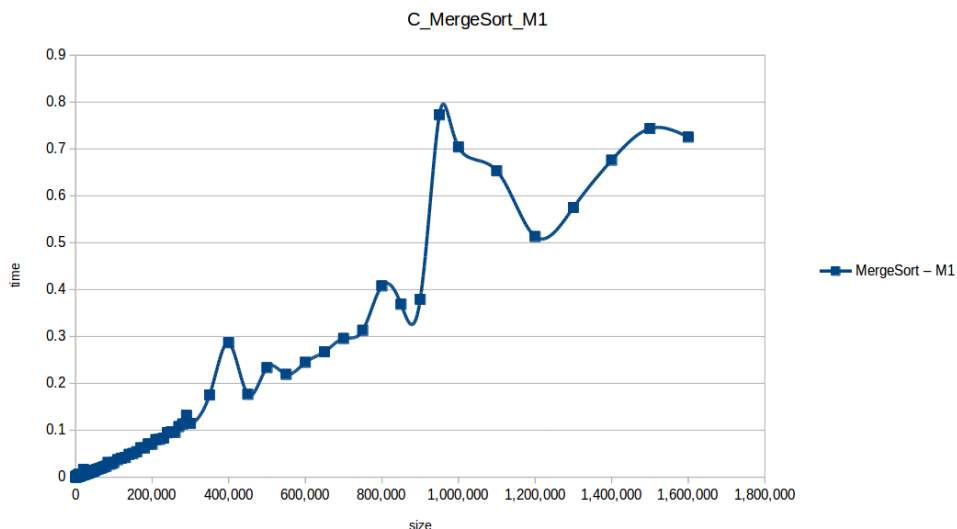
La complejidad de insert sort es de  $O(n^2)$  (Figura 1), mientras que para merge sort es de  $O(n \log n)$ , por tanto, Merge sort es mucho más eficaz que insert sort para arreglos de gran tamaño.

## ESTRUCTURA DE DATOS 1 Código ST0245



Xergioalex (2017) analysisOfSortAlgorithms: <https://bit.ly/2HnrvB2>

Figura 1.



Xergioalex (2017) analysisOfSortAlgorithms: <https://bit.ly/2HnrvB2>

Figura 2.

### 3.4 ¿Es apropiado usar insertion sort para un videojuego con millones de elementos?

No sería lo más apropiado, pues el rendimiento del videojuego se vería afectado entre más crezca el número de elementos que desea analizar, debido a la complejidad del algoritmo.

### 3.5 Para arreglos grandes, ¿en qué caso insertion sort es más rápido que merge sort? ¿cómo deben ser los datos para que insertion sort sea más rápido que merge sort?

## ESTRUCTURA DE DATOS 1

### Código ST0245

Insertion sort en el mejor de los casos tendría una complejidad de  $O(n)$  mientras que el merge sort su complejidad no varía de  $O(n \log n)$ , pero solo se podría llevar a cabo si los datos están ordenados de menor a mayor, sin necesidad de tener un arreglo en el ordenamiento, de resto la complejidad sería en promedio  $O(n^2)$ .

**3.6** Expliquen con sus propias palabras cómo funciona el ejercicio de Array 3 de Coding Bat llamado maxSpan y ¿por qué?

El código de maxSpan de codingbat, sirve para contar cuál es el mayor número de datos en un arreglo si este es igual al primer dato él cuenta hasta encontrar otro, o entre los datos más repetidos el mayor número de span que hay entre ellos, no tiene que ser el primer dato con el que se cuenta ej: [1,4,1,5,4], aquí max span sería de 4 empezando desde el cuatro de la derecha incluido hasta el 4 de la izquierda.

#### 4) Simulacro de Parcial

**4.1** a

**4.2** b

**4.3** length-1

**4.4** B

**4.5** D

**4.6** D

**4.7**  $T(n) = T(n-1) + C \rightarrow O(n)$

1.9 C

1.11 C

**PhD. Mauricio Toro Bermúdez**

Docente | Escuela de Ingeniería | Informática y Sistemas  
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627  
Tel: (+57) (4) 261 95 00 Ext. 9473