Reto 1: Documento de análisis

Integrantes del grupo

Grupo 11: **Federico Melo Barrero.**
202021525.
f.melo@uniandes.edu.co
Implementa el requerimiento individual 3.

Juan Camilo Prieto Avella.
201814815.
jc.prietoa@uniandes.edu.co
Implementa el requerimiento individual 4.

Análisis de complejidad de los requerimientos

Carga de los datos:

Al cargar los datos en el catálogo, se crean 5 listas ordenadas:

- artists_BeginDate: Tiene a los artistas ordenados por su fecha de nacimiento, BeginDate.
- artworks_DateAcquired: Tiene a las obras ordenadas por su fecha de adquisición, DateAcquired.
- artworks_Date: Tiene a las obras ordenadas por antigüedad, Date.
- artworks_Artist: Crea un vínculo entre un artista y sus obras.
- nationality: Crea un vínculo entre la nacionalidad de un artista y las obras.

Como las listas se ordenan al cargar los datos, sólo se ordenan una vez y luego se pueden usar los requisitos libremente.

Para ordenar las listas con las funciones de ordenamiento se eligió el algoritmo **mergesort**, que tiene como complejidad temporal O(n log(n)) en su peor caso, mejor caso y caso promedio, lo que lo hace el algoritmo más veloz de los vistos en clase.

Como estructura de datos se eligió el **arreglo** (ARRAY_LIST) para cada una de las listas, porque es la más rápida para el propósito de buscar en ella (nótese que getElement es O(1) en arreglo pero O(n) en lista encadenada) y ordenarla, tanto teóricamente como prácticamente.

Requerimiento 1

Para el requerimiento 1, se usa la lista de artistas artists_BeginDate que está ordenada cronológicamente por la fecha de nacimiento.

Cuando se ejecuta el requerimiento, se toma una sublista con artistas cuyas fechas de nacimiento están en el intervalo de años dados. Esto se hace con la función rangoArtists que tiene una

complejidad temporal de O(n), porque el primer ciclo acaba en la posición start de la lista y allí inicia el segundo. En el peor de los casos, que la posición inicial no está en la lista, corre el primer ciclo pero no el segundo.

En el view, se usa la función printReq1() para mostrar los resultados del requerimiento uno en una tabla, hecha con la librería prettytable. Esta función no interviene en el funcionamiento del requerimiento.

Requerimiento 2

Para el requerimiento 1, se usa la lista de obras artworks_DateAcquired que está ordenada cronológicamente por la fecha de adquisición.

Se implementó de forma muy similar al requerimiento 1. Cuando se ejecuta el requerimiento, se toma una sublista con obras cuyas fechas de adquisición están en el intervalo de fechas dadas. Esto se hace con la función rangoArtworks que tiene una complejidad temporal de O(n), porque el primer ciclo acaba en la posición start de la lista y allí inicia el segundo. En el peor de los casos, que la posición inicial no está en la lista, corre el primer ciclo pero no el segundo.

En el view, se usa la función printReq2() para mostrar los resultados del requerimiento dos en una tabla, hecha con la librería prettytable. Esta función no interviene en el funcionamiento del requerimiento.

Requerimiento 3:

Implementado por Federico Melo Barrero.

Se busca el ID del artista con base en el nombre dado, para lo que se usa la función id_artist() que tiene complejidad temporal de ~N, pues tiene que recorrer cada elemento de la lista de artistas buscando.
Luego, de la lista en artworks_Artist el catálogo que tiene el ID de cada artista todas sus obras, se halla el ID correspondiente (lo cual es O(n) o en notación tilda ~N).
Tras eso, se recorren las obras de dicho artista para meterlo a una lista de mediums, que es O(n) o en notación tilda ~N.
Por último, se busca el medium más grande y su posición, que es otro recorrido y también es O(n) o en notación tilda ~N.

Todos los ciclos son **independientes**, por eso cada uno es ~N y jamás hay ene cuadrado. En general, la complejidad es de ~4N que en función big O es simplemente O(n).

En el view, se usa la función printReq3(). Esta función no interviene en el funcionamiento del requerimiento.

Requerimiento 4:

Implementado por Juan Camilo Prieto Avella.

Para poder realizar este requerimiento creamos en la carga de las estructuras de datos un array list que tuviera una llave con la nación y otra llave con todas las obras que estuvieran dentro para esto tuvimos que recorrer la lista de artworks para luego buscar el id en la lista de artista para sacar la nacionalidad y poner esta información en el array list mencionado anteriormente, lo que costo O(2n) y luego usamos el algoritmo de merge sort (O(nlog(n)) para organizar por tamaño las naciones. Sin embargo, este proceso al usarlo al cargar los datos solo lo haremos una vez y cuando imprimamos la implementación de este requisito solo costara O(1) porque solo tendrá que imprimir valores ya ordenados y hechos en el requisito 0.

En el view, se usa la función printReq4(). Esta función no interviene en el funcionamiento del requerimiento.

Requerimiento 5:

Esta función tiene complejidad de ~2N, recorre las obras para obtener las de un departamento y guardarlas en una lista, y luego en otro ciclo independiente recorre las obras del departamento aplicándoles las condiciones para el precio. ~2N es equivalente a O(n).

En el view, se usa la función printReq5(). Esta función no interviene en el funcionamiento del requerimiento.

Requerimiento 6:

En el view, se usa la función printReq6(). Esta función no interviene en el funcionamiento del requerimiento.

Pruebas de tiempos de ejecución de los requerimientos

Máquina utilizada:

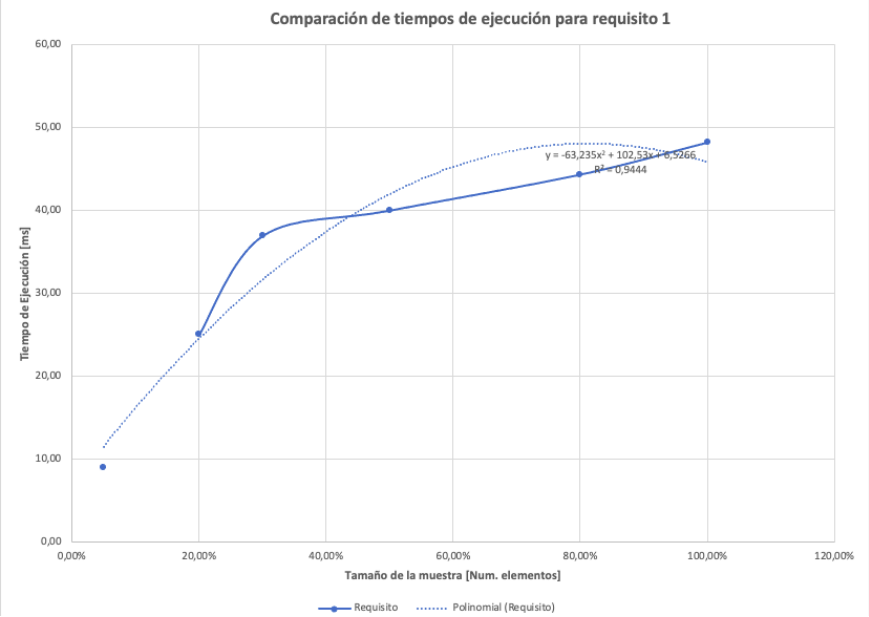
	Máquina
Procesadores	1,6 GHz Intel Core i5 de dos núcleos,
Memoria RAM (GB)	8.00 GB.
Sistema Operativo	MacOS Big Sur 64-bits

Requerimiento 1:
Tabla:

Porcentaje de la muestra [pct]	**Tiempo (ms)**
small	8,907
20%	25,056

30%	36,948
50%	40,008
80%	44,347
100%	48,201

Grafica:

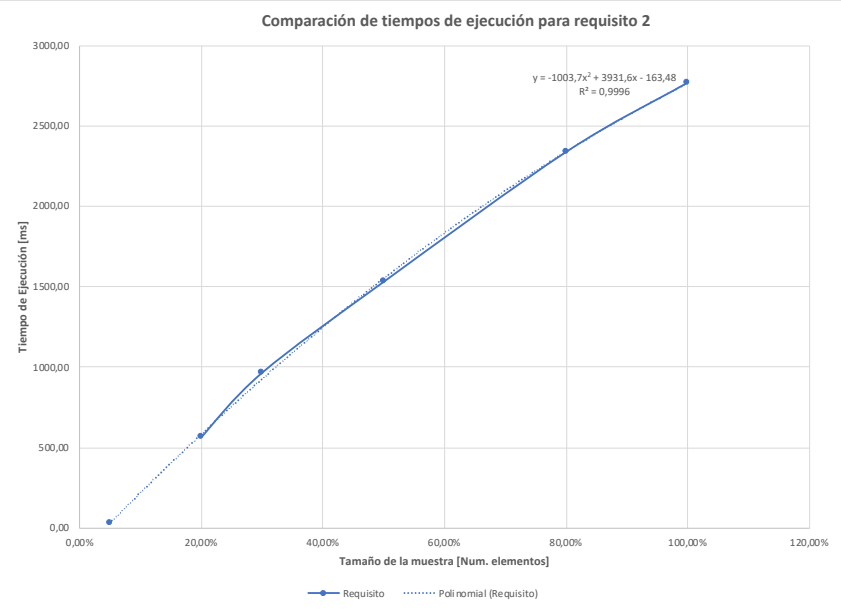


Requerimiento 2:

Tabla:

Porcentaje de la muestra [pct]	Tiempo (ms)
small	26,932
20%	565,22
30%	965,828
50%	1530,945
80%	2337,218
100%	2768,036

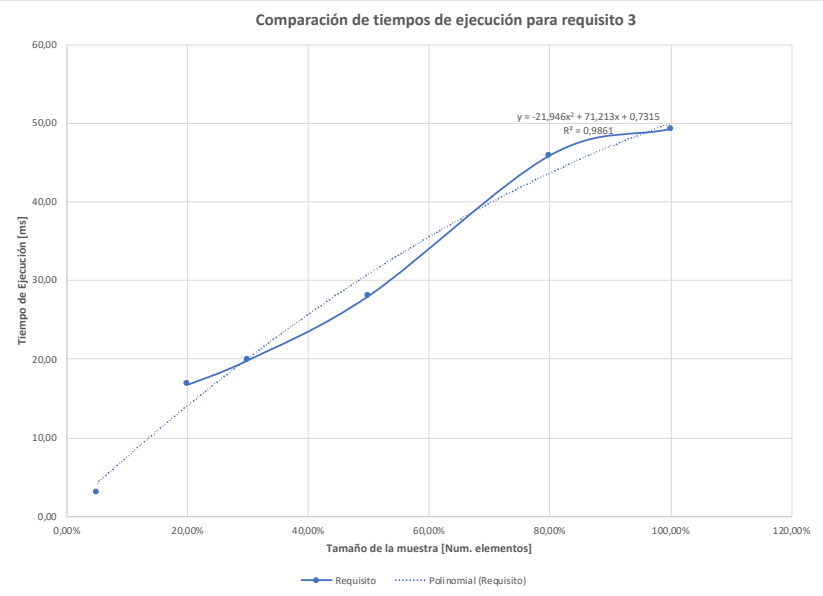
Grafica:



Requerimiento 3:
Tabla:

Porcentaje de la muestra [pct]	Tiempo (ms)
small	3,079
20%	16,822
30%	19,924
50%	28,089
80%	45,849
100%	49,198

Grafica:



Requerimiento 4:

Tabla:

Porcentaje de la muestra [pct]	Tiempo (ms)
small	8,537
20%	8,568
30%	7,359
50%	7,665
80%	9,480
100%	7,263

Grafica:

Requerimiento 5:
Tabla:

Porcentaje de la muestra [pct]	Tiempo (ms)
small	22,690
20%	18783,318
30%	44077,052
50%	136853,936
80%	328783,518
100%	509095,672

Grafica:


Comparación de tiempos de ejecución para requisito 5
600000,00
500000,00
400000,00
300000,00
200000,00
100000,00
0,00
-100000,00
Tiempo de Ejecución [ms]
y = 480173x² + 34467x - 5294
R² = 0,9998
0,00%
20,00%
40,00%
60,00%
80,00%
100,00%
120,00%
Tamaño de la muestra [Num. elementos]
Requisito
Polinomial (Requisito)