Análisis del reto 1

Alumno: Pablo Pedreros Díaz – 202112491 - p.pedreros@uniandes.edu.co

Carga de datos.

Ya con las pruebas realizadas en el laboratorio 4 pudimos ver que la estructura de datos más óptima en términos de tiempo para hacer los requerimientos es el **arreglo**, así que siempre vamos a usar arreglos para todas las listas en el Reto.

Requerimiento 1. ($N = Número\ de\ artistas$)

Nuestro requerimiento 1 empieza ordenando los datos por medio de merge sort, lo que sería NlogN de complejidad para un caso promedio como este, siendo N el número de artistas cargados. Con los elementos ordenados, sacamos por medio de una búsqueda binaria el índice del elemento mayor y menor que definen el rango de artistas que estará dentro de nuestra sublista que cumple el rango de fechas. Estas dos búsquedas binarias serán de logN cada una. Ya con la sublista podemos imprimir los valores sin iterar, por lo que el orden de complejidad de este requerimiento en notación Big O sería de $(O)NlogN$.

El promedio de tiempo luego de 3 intentos para el requerimiento por cada tamaño de archivo está dado por la siguiente tabla, usando como ejemplo un rango entre los años 1920 y 1985:

Porcentaje de datos (%)	Número de artistas	Tiempo (ms)
5	4996	125.0
10	6656	156.25
20	8724	218.75
30	10063	265.625
50	12137	328.125
80	14143	375.0
100	15223	406.25

Requerimiento 2. ($N = Número\ de\ obras$)

El requerimiento 2 funciona de la misma forma que el 1, solo que con obras. Empieza ordenando los datos por medio de merge sort y, ya con los elementos ordenados, saca por medio de búsqueda binaria el índice del elemento mayor y menor que definen el rango de obras que estará dentro de nuestra sublista que cumple el rango de fechas. Estas dos búsquedas binarias serán nuevamente de logN cada una. Ya con la sublista podemos imprimir los valores sin iterar, por lo que el orden de complejidad de este requerimiento en notación Big O sería de $(O)NlogN$.

El promedio de tiempo luego de 3 intentos para el requerimiento por cada tamaño de archivo está dado por la siguiente tabla, usando como ejemplo un rango entre las fechas de 6 de junio de 1944 y 9 de noviembre de 1989:

Porcentaje de datos (%)	Número de obras	Tiempo (ms)
5	7572	187.5
10	15008	359.375
20	29489	812.5
30	43704	1312.5
50	71432	2156.25
80	111781	3468.75
100	138150	4609.375

Requerimiento 3. ($N = Número\ de\ obras; M = Número\ de\ artistas; Q = Número\ de\ obras\ del\ artista$)

El requerimiento 3 empezará buscando el ID del artista que queremos por medio de búsqueda secuencial, pues esto tendrá complejidad M, mientras que organizar las obras por ID y luego buscar nuestro artista sería de MlogM + logM con Merge Sort. Ya con el ID, hará una lista con todas las obras del artista. Para este caso tampoco nos vale la pena organizar las obras, pues esto nos tomaría NlogN iteraciones del merge sort promedio, más las dos búsquedas binarias para encontrar el rango de obras que son de un artista, además que las obras con más de un artista nos causarían problemas y añadirían complejidad. En vez de esto, haremos una búsqueda secuencial buscando por cada obra si alguno de sus artistas es el que estamos buscando, y agregándola a una nueva lista si es el caso. Esto nos

daría un orden de complejidad de N, siendo N el número de obras cargadas. Lo siguiente que hará será mover todas las obras a una estructura de datos que las separa por medio, esto implicará comparaciones por cada elemento de la nueva lista de obras del autor (Q) que no es ni 1% de las lista de obras. Esto sería de orden de complejidad Q. Ya con estas iteraciones podríamos encontrar los valores que pide el requerimiento, por lo que el orden de complejidad en notación Big O sería de $(O)M + N + Q$.

El promedio de tiempo luego de 3 intentos para el requerimiento por cada tamaño de archivo está dado por la siguiente tabla, usando como ejemplo la artista Louise Bourgeois:

Porcentaje de datos (%)	Número de artistas	Número de obras	Número de obras del artista	Tiempo (ms)
5	4996	7572	159	15.625
10	6656	15008	304	15.625
20	8724	29489	650	46.875
30	10063	43704	982	78.125
50	12137	71432	1633	110.125
80	14143	111781	2654	171.875
100	15223	138150	3337	218.75

Requerimiento 5. ($N = Número\ de\ obras;\ M = Número\ de\ obras\ del\ departamento$)

Parecido al requerimiento 3, el requerimiento 5 empezará haciendo una lista con todas las obras de un departamento específico por medio de una búsqueda secuencial que sería de orden de complejidad N, pues tampoco nos es conveniente organizar los datos con NlogN solo para armar esta lista. Esto nos generará una lista de M elementos, y a cada uno tendremos que asignarle su valor de transporte, que serían M iteraciones. Ya con nuestra lista, para imprimir las obras más costosas y luego las obras más antiguas, tendremos que organizar por merge sort dos veces con diferentes parámetros nuestra lista de obras del departamento, lo que sería de complejidad 2MlogM (solo MlogM en notación Big O). Ya con la lista ordenada imprimiremos los valores que pide el requerimiento, por lo que el orden de complejidad en notación Big O sería de $(O)N + MlogM$.

El promedio de tiempo luego de 3 intentos para el requerimiento por cada tamaño de archivo está dado por la siguiente tabla, usando como ejemplo el departamento “Drawings & Prints:

Porcentaje de datos (%)	Número de obras	Número de obras del departamento	Tiempo (ms)
5	7572	4109	1515.625
10	15008	8133	2296.875
20	29489	15983	3718.75
30	43704	23709	5140.625
50	71432	38888	7390.625
80	111781	61397	10921.875
100	138150	76117	13281.25