**Análisis de complejidad temporal de las funciones**

Análisis de complejidad de las funciones para cargar los datos:

* **Cargar Artistas:** La función recorre con un “for” el archivo de entrada para cada línea de este archivo y crea un artista el cual añade a una lista nueva, estas funciones internas toman un tiempo “t” por lo cual la complejidad no se ve afectada por estas funciones, solo por el “f”. Por lo tanto, la complejidad es O(n).
* **Cargar Obras:** La función recorre con un “for” el archivo de entrada para cada línea de este archivo y crea una obra la cual añade a una lista nueva, estas funciones internas toman un tiempo “t” por lo cual la complejidad no se ve afectada por estas funciones, solo por el “for”. Por lo tanto, la complejidad es de una función polinomial O(n).

Análisis de complejidad de las funciones del requerimiento 1:

* **getArtistsbyYear:** La función recorre la lista completa una vez con un “for” y lo añade a otra lista más pequeña con los elementos requeridos. Después, con otro “for” se recorre la lista que se obtuvo anteriormente y en este se ordena con QuickSort el cual tiene una complejidad de O (log (n)). Por lo tanto, la complejidad es de una función polinomial O(n)^2\* O (log (n)).

Análisis de complejidad de las funciones del requerimiento 2:

* **getArtworksbyDate:** Esta función recorre la lista con 4 “for” y las líneas de código en medio de estos “for” son funciones de tiempo “t” por lo cual son despreciables. Por lo tanto, la complejidad es de una función polinomial O(n)^4.
* **PurchaseArtworks:** Esta función únicamente recorre la lista dada en la función anterior con un “for”. Por lo tanto, la complejidad es de una función polinomial O(n).
* **SortArtworks:** en esta se ordena con QuickSort el cual tiene una complejidad de O (log (n)).

La complejidad final es de O(n)^5\* O (log (n)).

Análisis de complejidad de las funciones del requerimiento 3:

* **ArtworksByArtist:** Primero recorre totalmente la función para encontrar el ID del artista. Luego, recorre nuevamente la función para encontrar las obras hechas por ese ID. La complejidad es de una función polinomial O(n)^2.

* **MediumInArtwork:** recorre una vez la lista buscando los medios de cada obra. La complejidad es de una función polinomial O(n).
* **FreqMediums:** Recorre una vez la lista sacando los medios y luego la vuelve a recorrer buscando la frecuencia de cada medio. La complejidad es de una función polinomial O(n)^2.
* **MostUsedMedium:** Recorre una vez la lista de frecuencias buscando el medio más usado. La complejidad es de una función polinomial O(n).
* **MUMList:** Recorre una vez la lista La complejidad es de una función polinomial O(n).

La complejidad final es de O(n)^7.

Análisis de complejidad de las funciones del requerimiento 4:

* **TopNacionalidades:** La lista grande es recorrida completamente dos veces. Además, se tiene un “for” dentro de otro “for” pero el “for” que está dentro recorre una lista que varía entre 1 y 21 elementos por lo cual no es comparable con el n que se está trabajando. La complejidad es de una función polinomial O(n)^2.
* **SortNacionalidades:** en esta se ordena con QuickSort el cual tiene una complejidad de O (log (n)).
* **MayorNacionalidad:** Esta función tiene un “for” dentro de otro “for” por lo que la complejidad si eleva a una función exponencial O (2) ^n.

La complejidad final es de O(n)^2 \* O (log (n)) \* O (2) ^n.

Análisis de complejidad de las funciones del requerimiento 5:

* **ArtworksByDepto:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **tamanoObras:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **pesoObras:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **precioObras:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **sumaTotal:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **obrasPorFecha:** Recorre una vez la lista La complejidad es de una función polinomial O(n).

* **Zipper:** Recorre una vez la lista La complejidad es de una función polinomial O(n).

* **Zipper2:** Recorre una vez la lista La complejidad es de una función polinomial O(n).
* **Obrasporcosto:** en esta se ordena con QuickSort el cual tiene una complejidad de O (log (n)).

La complejidad final es de O(n)^8 \* O (log (n)).

**Pruebas de Tiempos de Ejecución**

**Equipo 1:**

Procesador: AMD A10 9600P

Memoria RAM (GB): 12.0 GB

Sistema Operativo: Windows 10 Pro 64-bits

**(-large)**

1- Cargar información en el catálogo: 3281.25

2- Listar cronológicamente los artistas: 627078.125

3- Listar cronológicamente las adquisiciones: Más de 3 horas.

4- Clasificar las obras de un artista por técnica: 355187.5

5- Clasificar las obras por la nacionalidad de sus creadores: Más de 5 horas.

6- Transportar obras de un departamento: Más de 5 horas.

**(-small)**

1- Cargar información en el catálogo: 46.875

2- Listar cronológicamente los artistas: 812.5

3- Listar cronológicamente las adquisiciones: 468.75

4- Clasificar las obras de un artista por técnica: 15.625

5- Clasificar las obras por la nacionalidad de sus creadores: 5140.625

6- Transportar obras de un departamento: 1015.625

**Equipo 2:**

Procesador: Intel i7 8750 H 4.1GHz

Memoria RAM (GB): 16.0 GB

Sistema Operativo: Windows 10 Pro 64-bits

**(-large)**

1- Cargar información en el catálogo: 1468.75

2- Listar cronológicamente los artistas:

3- Listar cronológicamente las adquisiciones: Más de 3 horas.

4- Clasificar las obras de un artista por técnica:

5- Clasificar las obras por la nacionalidad de sus creadores: Más de 5 horas.

6- Transportar obras de un departamento: Más de 5 horas.

**(-small)**

1- Cargar información en el catálogo:

2- Listar cronológicamente los artistas:

3- Listar cronológicamente las adquisiciones:

4- Clasificar las obras de un artista por técnica:

5- Clasificar las obras por la nacionalidad de sus creadores:

6- Transportar obras de un departamento: