ANÁLISIS DEL RETO

Antonia Streubel, 202123766, *a.streubel@uniandes.edu.co*

Sofía Vásquez, 202123910, *s.vasquezt@uniandes.edu.co*

Isabella Caputi, 202122075, i.caputi@uniandes.edu.co

Pruebas de la Máquina con Diferentes Tamaños de Datos

	Máquina
Procesadores	Intel(R) Core(TM) i3-7100U CPU @ 2.40GHz 2.40 GHz
Memoria RAM (GB)	8,00 GB
Sistema Operativo	Sistema operativo de 64 bits, procesador basado en x64

Maquina 1

Resultados

Porcentaje de la muestra [pct]	**Req. 1**	**Req. 2**	**Req. 3**	**Req. 4**	**Req. 5**	**Req. 6**	**Req. 7**
0.50%	2.235	0.900	0.013	0.119	0.017	0.013	2.098
5.00%	45.952	15.416	0.017	1.775	0.028	0.017	10.678
10.00%	174.218	99.467	0.014	6.737	0.085	0.032	78.987
20.00%	867.173	180.705	0.012	39.090	0.329	0.081	200.77
30.00%	2158.884	392.682	0.119	78.278	0.431	0.140	454.89
50.00%	7608.461	1651.363	0.154	309.97	0.906	0.295	734.78
80.00%	20134.31	3452.479	0.244	727.04	2.257	0.310	1023.56
100.00%	33382.09	6031.558	0.306	1108.9	2.845	1.665	214.67

Entradas:

1. 1920-1999
2. 2018-01-07; 2020-12-31
3. Sissy Spacek
4. Fantasy
5. Colombia
6. John Hughes
7. 3

Requerimiento << 0>>

Descripción

Para abordar este requerimiento, decidimos previamente crear un catálogo vacío que contiende 5 listas, una para cada Streaming y una total con todos los datos, de manera de carga de datos. Luego, recorrimos los archivos según el tamaño del archivo seleccionado por el usuario y añadimos los datos respectivos a cada una de las listas del catálogo.

Entrada	Funcion1: Tipo_lista(ARRAY o SINGLE_LINKED): str para crear un catálogo con las listas de datos (Hulu, Prime…). Funcion2: catalogo (con las listas vacías): dict, tamanio_datos (tamaño de los archivos de los datos a cargar en las listas): str (-small, -5pct…).
Salidas	Catalogo (dict) con todos los datos cargados en listas, lista organizada por año de lanzamiento
Implementado (Sí/No)	Sí, grupal (entre los tres estudiantes).

Requerimiento << 1>>

Adicionalmente, lo que decidimos hacer fue crear una función que comparará los años entrados por parámetro con los años de lanzamiento de cada elemento de la lista total. Después de comparar los años y encontrar cuales películas o series hacían parte del rango entrado por parámetro, metimos cada uno de los datos como parte de este rango en otra lista. Durante este proceso, también creamos un contador que fuera capaz de hacer cuenta de la cantidad de películas que hacían parte de este rango de años. Luego, imprimimos los primeros 3 y últimos tres datos de esta nueva lista creada.

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_T) - Año inicial del periodo (formato: AAAA) - Año final del periodo (formato: AAAA)
Salidas	- Número de películas en ese periodo - La información de las primeras 3 películas de la lista - La información de las primeras 3 películas de la lista
Implementado (Sí/No)	Sí, grupal (entre los tres estudiantes).

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Para el requerimiento 0 de carga de datos...

Pasos	**Complejidad**
Paso 1: crear las listas vacías de Hulu, Amazon, Prime y Disney en un catálogo, adicional a una lista Total conjunta con todos los datos.	O(1)

Paso 2: Preguntarle al usuario el tamaño de la muestra de los datos que desea	O(1)
Paso 3: Cargar los datos de los cuatro Streaming en las listas creadas en el catálogo, y todos los datos en una lista total.	O(n)
Paso 4: Cada lista (la total (Sortedlist_T) y las 4 listas por Streaming) se organiza por año de lanzamiento, título y duración con la función de CmpFunctionByRealeaseYear y el algoritmo shell sort.	O(log(n))
Paso 5: imprimir los primeros 3 y últimos 3 datos de cada lista por Streaming de los datos cargados en la lista organizada anterior.	O(n)
Paso 6: se crea una lista Sortedlist_Series organizada por fecha en la que fue agregado, título y duración con la función de cmpSeriesByDateAdded y el algoritmo shell sort para uso del requerimiento 1.	O(n^2)
Paso 7: se crea una lista Sortedlist_Actor organizada por título, año de lanzamiento y duración con la función de CmpByTitles y el algoritmo shell sort. para uso del requerimiento 2.	O(n^2)
Paso 8: se crea una lista Sortedlist_Director organizada por año de lanzamiento, título, y director con la función de cmpByReleaseYearAndDirector y el algoritmo shell sort para uso del requerimiento 3 y 4.	O(n^2)
TOTAL	***O(***n^2+n+log(n)+1***)***

Para el requerimiento 1...

Pasos	**Complejidad**
Paso 1: Se le piden al usuario los datos de los años: inicial y final para el rango.	O(1)
Paso 2: Se crea una nueva lista para guardar las películas que hacen parte de este rango.	O(1)
Paso 3: Se recorre la lista total que contiene todos los datos de los cuatro Streamings, encontrando las que sean películas (es decir, que su “type” = “Movie”) y que, a la vez, sus años de lanzamiento estén dentro de los años dados por parámetro. Estos títulos que cumplan estas condiciones se van agregando a una lista nueva. (while+if)	O(n*log(n))
Paso 4: Se devuelve la lista con los títulos que hacen parte del rango de los años que entran por parámetro. En la vista, se imprime la longitud de la	O(1)

lista como la cantidad de títulos que hacen parte este rango.	
Paso 5: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 6: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	O(1)
TOTAL	***O(n*log(n)+1)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
1920; 1999	1.800
2003; 2022	51.949
1970;2003	3.384

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Se pueden ver grandes diferencias en los tiempos de ejecución por más de ser hechas sobre una misma base de datos. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargo en el repositorio de GIT-hub con el fin de poder ejecutarlas.

Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.

Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Tras ver la toma de tiempos al tomar la medida de los diferentes tamaños de datos podemos evidenciar como a medida que crece la muestra de datos, crece el tiempo de ejecución. La gráfica nos muestra un tiempo un crecimiento linearítmico como también nos indica la tabla del análisis de complejidad.

Requerimiento <<2>>

Para abordar este requerimiento, lo que decidimos hacer fue parecido a lo que realizamos en el requerimiento 1. Para esto, creamos una función que comparará las fechas entradas por parámetro con las fechas de cada elemento de la lista total. Después de comparar las fechas y encontrar cuales películas o series hacían parte del rango entrado por parámetro, metimos cada uno de los datos como parte de este rango en otra lista. Luego, imprimimos los primeros 3 y últimos tres datos de esta nueva lista creada. En este caso las fechas eran entregadas en un formato distinto, por lo que tuvimos que separar la entrada de fecha por año, mes y día, para después comprara la fecha de lanzamiento de las películas por, en caso necesario, los tres valores, confirmando que se encontrara en el rango.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_Series) - Fecha inicial periodo (formato: año-mes-día)

	- Fecha final periodo (formato: año-mes-día)
Salidas	- Número de series presentes en periodo - La información de las primeras 3 series de la lista - La información de las ultimas 3 series de la lista
Implementado (Sí/No)	Sí, se hizo de forma grupal.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Se le piden al usuario los datos de las fechas: inicial y final para el rango.	O(1)
Paso 2: Se crea una nueva lista para guardar las películas que hacen parte de este rango.	O(1)
Paso 3: Se recorre la lista total que contiene todos los datos de los cuatro Streamings, y se separa la fecha de cada elemento en una lista utilizando la función Split("-"). (while)	O(n)
Paso 4: Dentro de este mismo recorrido se encuentra si el dato que está siendo recorrido no tiene una fecha vacía, su tipo es de serie, y que su año, mes y día estén dentro del rango correspondiente dado por parámetro. Estos datos que hagan parte del rango se añaden a una lista nueva que se va a devolver más tarde. (while+if+if+if)	O(n*log(n))
Paso 5: Se devuelve la lista con los títulos que hacen parte del rango de las fechas que entran por parámetro.	O(1)
Paso 6: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 7: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	O(1)
TOTAL	***O(***n*l***og(n)+n+1)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
2018-01-07; 2020-12-31	0.900

2020-01-07; 2022-12-31	0.382
2018-02-17; 2020-11-31	0.654

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Se pueden ver grandes diferencias en los tiempos de ejecución por más de ser hechas sobre una misma base de datos. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.

Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Tras ver los tiempos de medida de este requerimiento evidenciamos un comportamiento muy similar al del requerimiento 1, sin embargo, a diferencia del otro, esta muestra un crecimiento logarítmico y cuadrático en el gráfico.

Requerimiento <<3>>

Para abordar este requerimiento, optamos por crear una lista, en donde, al recorrer la lista ya posteriormente creada (en donde se encuentra toda la información de películas y series antes recolecta), guarda los elementos que contengan al actor pasado por parámetro. Al comparar los actores de a película y el del parámetro, y concordar, se agrega la información de esta película/ serie a la lista. Se comprarán los elementos hasta llegar al final de la lista. Después se recorre esta lista de los elementos en donde se encuentra el actor pasado por parámetro y se contabiliza cuantos elementos son películas y cuantas series. Después de esto se toma la lista de los elementos por artista (películas + series) y se organizan alfabéticamente por título y luego por año de lanzamiento y duración, para posteriormente mostrar los primeros 3 elementos y los últimos 3 (en cade de que se encuentren 6 o más elementos por artista), si este solo ha participado en menos de 6, se imprime la lista completa.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_Actor) - Nombre del actor al cual se quiere encontrar el contenido
Salidas	- El número de películas en las que participa el actor - El número de series en las que participa el actor - La información de los primeros 3 elementos de la lista - La información de los últimos 3 elementos de la lista
Implementado (Sí/No)	Sí, lo implementó el estudiante Antonia Streubel.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Pedirle el usuario el nombre de actor para encontrar el contenido de las películas o series a las que pertenece el actor.	O(1)
Paso 2: Se crea una nueva lista para guardar las películas y series a las que pertenece el actor dado por parámetro.	O(1)
Paso 3: Se recorre la lista organizada (sorted_listT) y se encuentra si el actor entrado por parámetro hace parte del “cast” del elemento que está siendo recorrido. Si el título que esta siendo recorrido tiene como parte de su “cast” al actor, se añade el dato a la lista creada en el paso 2 y si no, no se agrega. (while+if)	O(n*log(n))
Paso 3: Se crean dos listas adicionales, una para separar las series y otra para separar las películas de la lista creada y llenada en el paso	O(n*log(n))

3. Se recorre la lista de las películas y series del actor, y se compara el tipo de cada título, para saber si es un “Movie” o un “TV Show”. Si es una película, lo guarda en la lista de las películas, y si es una serie, la guarda en la lista de series. (while+if)	
Paso 4: Después, se devuelve la lista de películas, la lista de series en las que participa el actor.	O(1).
Paso 5: Imprime la cantidad de películas y series por separado en las que participan el actor (longitud de cada una de las listas devueltas en el paso anterior).	O(1)
Paso 6: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 7: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	O(1)
TOTAL	***O(n*log(n)+1)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
Sissy Spacek	0.013
Pedro Infante	0.014
Edward G. Robinson	0.015

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

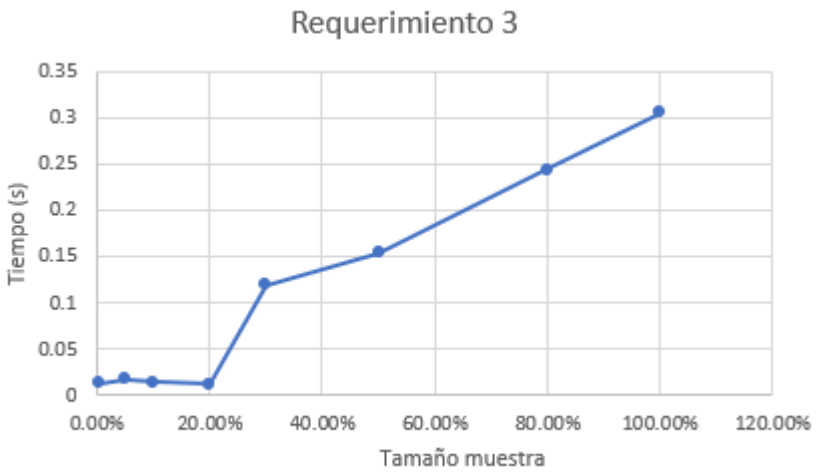
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Tras ver los tiempos de medida de este requerimiento evidenciamos un comportamiento crecimiento linearitmico en la gráfica. Aunque no es completamente recto, pues tiene unas irregularidades. Según las pruebas no se encuentra una gran diferencia de tiempo debido al cambio de tamaño de los datos.

Requerimiento <<4>>

Descripción

Para abordar este requerimiento, optamos por crear una lista, en donde, al recorrer la lista ya posteriormente creada (ordenada por el criterio compuesto por título (title), fecha de lanzamiento (release_year) y director (director), guarda los elementos que contengan el género ingresado por parámetro. Además, generamos dos contadores, uno para las películas y uno para las series. Luego, retorna la lista que contiene el género ingresado por el parámetro, el número de películas y el número de series. Finalmente, imprimimos los primeros 3 y últimos 3 datos de esta nueva lista creada.

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_Director) - Género
Salidas	- Cantidad de películas por género - Cantidad de series por género - La información de los primeros 3 elementos de la lista - La información de los últimos 3 elementos de la lista
Implementado (Sí/No)	Sí, lo implementó el estudiante Sofia Vasquez.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Pedirle el usuario el nombre del género para encontrar el contenido de las películas o series a las que pertenece este.	O(1)
Paso 2: Se crea una nueva lista para guardar las películas y series a las que pertenecen al género dado por parámetro.	O(1)
Paso 3: Se recorre la lista organizada (sorted_listDirector) y se encuentra si el género entrado por parámetro hace parte del "listed_in" del elemento que está siendo recorrido. Si el título que está siendo recorrido tiene como parte de su "listed_in" al género, se añade el dato a la lista creada en el paso 2 y si no, no se agrega. (while+if+if+elif) Ademas, revisa si el "type" del elemento que está siendo recorrido (Movie o TV Show) y lo añade a dos contadores para luego devolverlos.	O(n*log(n))
Paso 4: Retorna la lista creada, el número de películas y el número de series que contienen el género ingresado por parámetro	O(1)
Paso 5: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 6: se encuentra en la nueva lista creada organizada los ultimos 3 datos a través de un while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	O(1)
Paso 7: Le informa al usuario el total de contenidos que pertenece al género ingresado por parámetro y cuantos de ellos son películas y cuantas son series.	O(1)
TOTAL	O(n*log(n)+1)

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
Fantasy	0.199
Comedy	2.362
Drama	18.851

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

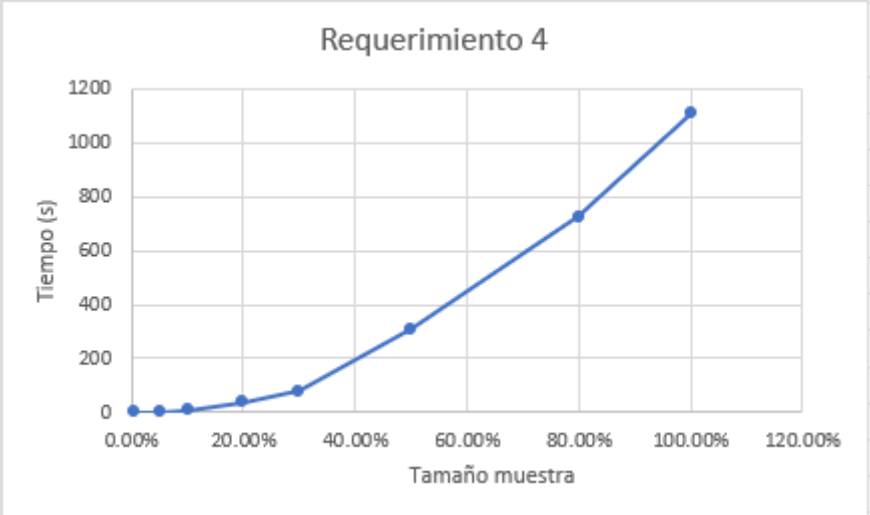
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Este gráfico nos muestra igualmente un comportamiento linearitmico. Los resultados de la toma de datos en este ejercicio y su análisis de complejidad también nos llevan a la misma solución.

Requerimiento <<5>>

Para este requerimiento se siguió un procedimiento similar al requerimiento pasado. Se recorre la lista previamente creada con todos los datos, para comparar el país de producción del elemento recorrido con el nombre del país pasado por parámetro. En caso de que estos elementos concuerden, se agrega el elemento en la lista. Además de agregar este elemento a la lista, se revisa si este es una película o una serie y se agrega a un contador correspondiente.

Al llegar al final de la lista de datos, se organiza la lista de elementos comparados según el orden alfabético de títulos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_Director) - País de producción
Salidas	- Número de películas producidas en el país - Número de programas producidas en el país - La información de los primeros 3 elementos de la lista - La información de los últimos 3 elementos de la lista
Implementado (Sí/No)	Sí, lo implementó el estudiante Isabella Caputi

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Pedirle el usuario el nombre del país para encontrar el contenido de las películas o series a las que pertenece este.	O(1)
Paso 2: Se crea una nueva lista para guardar las películas y series a las que pertenece el país dado por parámetro.	O(1)
Paso 3: Se recorre la lista organizada (sorted_listDirector) y se encuentra si el país entrado por parámetro es igual al “country” del elemento que está siendo recorrido. Si al analizar el título que está siendo recorrido, su “country” es igual al país entrado por parámetro, se añade el dato a la lista creada en el paso 2 y si no, no se agrega. (while+if+if+elif). Además, revisa si el “type” del elemento que está siendo recorrido (Movie o TV Show) y lo añade a dos contadores para luego devolverlos.	O(n*log(n))
Paso 4: Retorna la lista creada, el número de películas y el número de series (contadores) que contienen el país ingresado por parámetro	O(1)
Paso 5: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 6: se encuentra en la nueva lista creada organizada los últimos 3 datos a través de un	O(1)

while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	
Paso 7: Le informa al usuario el total de contenidos que pertenece al país ingresado por parámetro y cuantos de ellos son películas y cuantas son series.	O(1)
TOTAL	O(n*log(n)+1)

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
Colombia	0.013
Argentina	0.011
United States	4.593

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

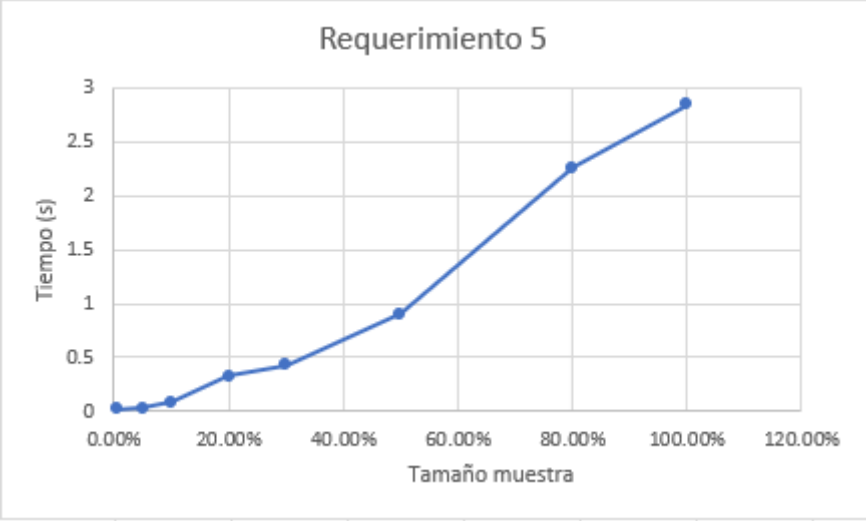
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Trans analizar los resultados de los tiempos evidenciamos como este requerimiento tiene un ordenamiento temporal Linearítmico. Esto lo deducimos por la forma de la curva de la gráfica, dada por lso datos obtenidos por la medida de tiempo y el análisis de complejidad.

Requerimiento <<6>>

Descripción

Para abordar este requerimiento, decidimos crear una nueva lista que guardara todos los datos de las películas y series de un director dado por parámetro, comparándola con el valor de la llave de “director” en cada elemento de la lista organizada por año de lanzamiento (Sortedlist_T). Adicionalmente, se cuenta el número de películas y de series con dos contadores y se imprimen estos resultados. Además, se cuentan cuantas series y cuantas películas hay por streaming service. Luego se imprimen los tres primeros datos y los últimos tres datos de la lista creada de los títulos en el que el director participo.

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_T) - Director del cual se quiere encontrar el contenido
Salidas	- Numero de series y películas dirigidas por director - Numero de películas y series distribuidas por género - La información de los primeros 3 elementos de la lista - La información de los últimos 3 elementos de la lista - Número de película series distribuidas por plataforma
Implementado (Sí/No)	Sí, se hizo de forma grupal

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Se crea una lista para guardar los datos en los que el director es el que aparece en el valor de la llave de “director” de los elementos de la lista SortedList_T.	O(1)
Paso 2: Se recorre la lista SortedList_T y se van añadiendo a la lista de los datos del director, y se van contando las películas y las series de estos datos (al revisar con el valor de la llave de “type” de cada elemento). (while-if-if)	O(n*log(n))
Paso 3: Se recorre la lista creada del director, y se encuentran las películas y series por servicio de Streaming y se guardan los datos de cada servicio en un diccionario.	O(n*log(n))
Paso 4: Se crea una lista donde se guardan los géneros que existen en la llave “listed_in” de los datos de la lista creada para el director dado por parámetro. Estos tags de los géneros que se encuentran se guardan en un nuevo diccionario con su respectiva cantidad por tag (es decir, cuantas veces se encuentra el tag en la lista de datos por el director).	O(n^2)
Paso 5: Se devuelve la lista con los datos por director, el diccionario por Streaming, y los diccionarios con los tags de género y su respectiva cantidad.	O(1)
Paso 6: Se imprimen el número total de películas y series dirigidas por el director, además de la cantidad de películas y la cantidad de series por separado.	O(1)
Paso 7: Se imprime la cantidad de peliculas por servicio de Streaming y la cantidad en cada servicio de Streaming utilizando el diccionario que guarda estos valores.	O(n)
Paso 8: Se imprimen los tags de los géneros encontrados en los datos de la lista del director con su respectiva cantidad de apariciones.	O(n)
Paso 9: se encuentra en la nueva lista creada organizada los primeros 3 datos a través de un while que vaya de la posición 1 a la posición 3 de la lista. Se imprimen estos datos.	O(1)
Paso 10: se encuentra en la nueva lista creada organizada los últimos 3 datos a través de un while que vaya de la posición ante penúltima hasta la posición final de la lista. Se imprimen estos datos.	O(1)

TOTAL	***O(***n^2+n*log(n)+1***)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
John Hughes	0.014
Steven Spielberg	0.012
Woody Allen	0.011

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño “-small”. Para realizar estas tomas se tomó la carpeta de “Streaming” y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

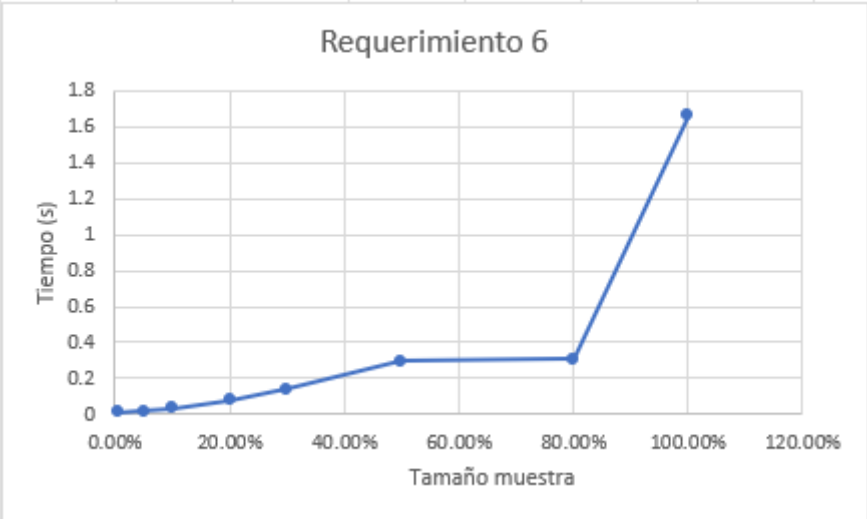
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Tras analizar el análisis de complejidad de este requerimiento, evidenciamos que se trata de una función de tipo Linearítmico y cuadrático. Por el análisis de su complejidad podemos deducir lo mismo, ya que el orden que predomina en a la función es de tipo (n^2*(log(n))).

Requerimiento <<7>>

Descripción

Para abordar este requerimiento, decidimos crear un diccionario que guardara las películas y series a las que pertenece cada género. Luego creamos varias listas para separar los géneros y la cantidad de veces que aparecía cada uno en la lista. Luego organizamos los datos de la cantidad de veces que aparece cada genero con un mergesort por mayor a menor. Despues, comparamos que este valor se encontrara en la lista con los tags y los valores para sacar el numero N del TOP que se desea ver. Finalmente, se imprime N géneros organizados por el número de películas y programas y de cada género: el número total de películas y programas por cada plataforma y el número total de películas y programas que contienen el género.

Entrada	- Lista total organizada por año de lanzamiento (carga de datos: Sortedlist_T)) - Cantidad de datos que se quieren identificar en el TOP de los géneros
Salidas	- El grupo de N géneros organizados por el número de películas y programas - De cada género: el número total de películas y programas por cada plataforma y el número total de películas y programas que contienen el género.
Implementado (Sí/No)	Sí, se implementó de forma grupal.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Pedirle el usuario el número de géneros que desea ordenar en el TOP.	O(1)
Paso 2: Se crea un diccionario para guardar las películas y series a las que pertenece cada género (donde las llaves son los tags y los valores son la cantidad de veces que aparece el género en la lista inicial).	O(1)
Paso 3: Se recorre la lista organizada (sorted_listT) y se separan los géneros que contiene cada elemento en una lista utilizando la función Split (“-”). Luego, revisa cada genero del elemento y si este no se encuentra como una llave del diccionario creado anteriormente, es añadido y se le da un valor de 1. Si se encuentra	O(n^2)

que esta llave ya está en el diccionario, se le suma 1 a el valor (este proceso se realiza la cantidad de veces igual a la longitud de los géneros totales). (while+while+if+else)	
Paso 4: Se crea una lista (listanombres) para guardar los nombres de todos los géneros (tags).	O(1)
Paso 5: Recorre cada llave del diccionario creado anteriormente y añade estos tags a la lista (listanombres).	O(n)
Paso 6: Se crea una lista (listanumeros) para guardar la cantidad de veces que aparecen los géneros (tags) encontrados.	O(1)
Paso 7: Recorre cada valor del diccionario creado anteriormente y añade estos valores a la lista (listanumeros).	O(n)
Paso 8: Realizamos un mergesort para ordenar la listanumeros de mayor a menor	O(n log (n))
Paso 9: Se crea una lista (listaposiciones) para guardar el nombre en la que se encuentra el valor de los tags.	O(1)
Paso 10: Recorre cada valor de la listanumeros (la cual está organizada de mayor a menor) y recorre cada tag de listanombres. Con esto, compara si de la listageneros (el diccionario creado al comienzo) con la llave de cada tag, es igual al número de listanumeros (organizado de mayor a menor) y si el size de la listaposiones es menor a la longitud del diccionario. Si esto es verdadero, añade el nombre a la listaposiciones. (for+for+if)	O(n^2)
Paso 11: Se crea una lista (listafinaltags) para guardar el nombre de los tags que se encuentra en el TOP.	O(1)
Paso 12: Se crea una lista (listafinalnumeros) para guardar la cantidad de veces que aparecen los tags que se encuentran en el TOP.	O(1)
Paso 13: Se recorre un while desde que un contador sea menor o igual a el número de géneros que desea ordenar en el TOP. En este, se sacan 2 elementos, uno de listaposiciones (corresponde los tags) y uno de listanumeros (corresponde a la cantidad de veces que aparece el tag). Luego, añade cada tag a la lista listafinaltags y cada cantidad a listafinalnumeros. (while)	O(n)
Paso 14: Se crean 6 listas en las que se guardan, la cantidad de peliculas, la cantidad de series y la cantidad de veces que aparece el género en cada streaming service (4).	O(1)

Paso 15: Se recorre un while desde que un contador sea menor o igual a el número de géneros que desea ordenar en el TOP. Se saca cada elemento de la listafinaltags (elemento2) y se inicializan 6 contadores igualados a 0 (1 que cuenta la cantidad de peliculas, 1 que cuenta la cantidad de series, 4 que cuenta la cantidad total de películas y programas por streaming service). Se recorre un while desde que un contador sea menor o igual a el size de la lista total. Se saca cada elemento de la lista total (elemento1) y se comprara si elemento2 está en "listed_in" del elemento1. Luego mira si este es una pelicula o una serie y se suma a los contadores correspondientes. Despues, revisa cada Streaming service y le suma a su contador correspondiente. Finalmente, añade a cada lista creada en el paso anterior, su contador correspondiente. (while+while+if+if+if)	O(n log(n))
Paso 16: Retorna la lista creada de los listafinaltags, la lista creada de los listafinalnumeros, la lista de las peliculas, la lista de las series y cada la lista de cada streaming service.	O(1)
Paso 17: Imprime el grupo de N géneros organizados por el número de películas y programas (while)	O(n)
Paso 18: Imprime por cada género el número total de películas y programas por streaming service, el número de películas y el número de programas. (while)	O(n)
TOTAL	***O(***n^2 **+ *n log (n)+n+1)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Entrada	**Tiempo (s)**
3	2.098
5	4.345
9	5.673

Bajo una memoria de 8GB en una maquina con las descripciones descritas en la parte superior del archivo, encontramos estos tiempos de ejecución bajo estas entradas. Estas tomas fueron medidas bajo los datos de tamaño "-small". Para realizar estas tomas se tomó la carpeta de "Streaming" y se cargó en el repositorio de GIT-hub con el fin de poder ejecutarlas.

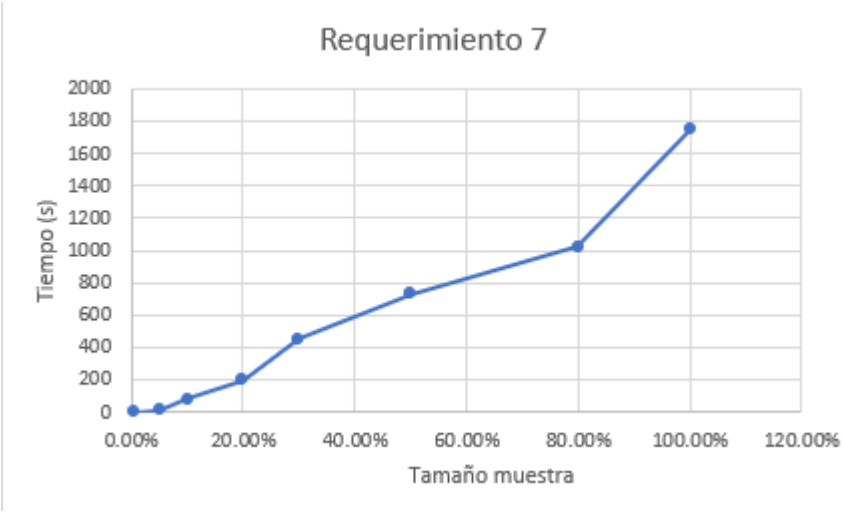
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

- La tabla se encuentra adjunta en la parte superior del documento.

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.

Después de revisar el comportamiento del gráfico sacado de la toma de tiempos del ejercicio, evidenciamos que se trata de un orden de crecimiento linearitmico. Esto también lo podemos comprobar con el análisis de complejidad.