ANÁLISIS DEL RETO

Laura Calderón, 202122045, l.calderonm
Manuela Lizcano, 202122826, m.lizcanoc
Mariana Pineda Miranda, 202123330, m.pinedam

Ambientes de pruebas

	Máquina 1	Máquina 2	Máquina 3
Procesadores	3,1 GHz Intel Core i5 de dos núcleo	2,00 GHz Ryzen 7 con gráfico de radeon	1,8 GHz Dual-Core Intel Core i5
Memoria RAM (GB)	8 GB	8 GB	8 GB
Sistema Operativo	MacOs	Windows 10	MacOs

Tabla 1. Especificaciones de las máquinas para ejecutar las pruebas de rendimiento.

Carga de datos

Carga de Catálogo

Factor de Carga	Consumo de Datos [kB]	Tiempo de Ejecución Real @LP [ms]
small	445,66	98,15
5pct	3253,09	479,36
10pct	5956,01	1061,89
20pct	10690,79	1864,73
30pct	15077,55	2890,94
50pct	23216,07	4484,52
80pct	34377,75	6656,32
large	41428,56	7921,40

Requerimiento 1

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Plataforma en la cual se quiere realizar la busqueda, limite inferior fecha de lanzamiento, limite superior fecha de lanzamiento.
Salidas	El numero de total de videojuegos en la plataforma que se encuentren en el rango de fechas. Los tres primeros y los tres ultimos de la lista de elementos que se encuentran en el rango.

Implementado (Sí/No)	Si se implementó.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1. Se busca en el catalogo el mapa en el que se encuentan los elementos ordenados por su plataforma.	O(1)
Paso 2. Crea una lista en la que se van a guardar los elementos que cumplan el rango.	O(1)
Paso 3. Se busca en el mapa el arbol de la plataforma que entra por parametro.	O(n)
Paso 4. Se recorre la lista de elementos que pertenecen a la plataforma, se mira si la fecha se encuentra entre el rango indicado y si es asi se añade al final de la lista creada en el Paso 2.	O(n)
Paso 5. Se ordena la lista por medio del ordenamiento Shell para tener los datos de acuerdo a la fecha de	O(nlogn)
TOTAL	***O(n)***

Pruebas Realizadas – Con máquina 3

Se realizaron las pruebas de tiempo de ejecución para este requerimiento con varios de los archivos variando su tamañp, cargando los datos por medio de un catálogo en el que se creó una lista con las películas y diferentes índices dependiendo de los requerimientos. Con arboles rojo y negro para facilitar su busqueda. Se ejecutaron las pruebas en tres computadores. Las pruebas se realizaron con los diferentes archivos con diferentes tamaños. Además, se usó la librería nativa de python time. Para realizar estas pruebas se utilizaron como parametros la plataforma PC, el limite inferior de fecha 2000-01-01 y limite superior de fecha 2005-12-31, ademas se repitió la prueba tres veces para tener un valor más exacto del resultado.

Tablas de datos

REQUERIMIENTO 1	
Tamaño archivo	**Tiempo**
small	0,85
5pct	4,12
10pct	4,78
50pct	5,04
80pct	6,56
large	7,14

Graficas

Análisis

Realizando el analisis paso a paso del algoritmo implementado se puede observar que este tiene una complejidad O(n) debido a que realiza un solo recorrido a todos los datos para poder encontrar las fechas en el rango ingresado por parametro. De igual manera, se puede ver reflejado en las pruebas de tiempos que el tiempo aumenta conforme a la cantidad de datos por lo que la aproximación a una función de complejidad O(n) es correcta. Los tiempos no pasan de los 8 segundos por lo que son bastante buenos para la cantidad de datos que se estan analizando.

Requerimiento 2

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Nombre del jugador del que se quieren encontrar los 5 registros mas rapidos.
Salidas	El total de elementos en los que el jugador tiene el menor tiempo, los 5 registros con menor tiempo en los que el jugador es el mas rapido.
Implementado (Sí/No)	Sí se implementó

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1. Se busca en el catalogo el indice en donde esta el arbol que contiene los registros ordenados por el jugdor.	O(1)
Paso 2. Se busca el jugador en el arbol del Paso 1.	O(n)
Paso 3. Se encuentra la lista de registros por medio de acceder al valor del arbol encontrado.	O(1)
Paso 4. Se organiza la lista por medio del algoritmo shell teniendo el cuenta el tiempo mas rapido del jugador.	O(nlogn)

Paso 5. Se recorre la lista de los juegos con respecto a los registros para encontrar el nombre del videojuego en el que el jugador tuvo el tiempo mas rapido.	O(n)
Paso 6. Se crea una nueva llave en cada uno de los registros en la cual se incluye el nombre del videojuego coincidido en el Paso 5.	O(1)
TOTAL	***O(1)***

Pruebas Realizadas – Con máquina 3

Se realizaron las pruebas de tiempo de ejecución para este requerimiento con varios de los archivos variando su tamañp, cargando los datos por medio de un catálogo en el que se creó una lista con las películas y diferentes índices dependiendo de los requerimientos. Con arboles rojo y negro para facilitar su busqueda. Se ejecutaron las pruebas en tres computadores. Las pruebas se realizaron con los diferentes archivos con diferentes tamaños. Además, se usó la librería nativa de python time. Para realizar estas pruebas se utilizó como parametro el jugador Flamming en cada una de las pruebas, ademas se repitió la prueba tres veces para tener un valor más exacto del resultado.

Tablas de datos

REQUERIMIENTO 2	
Tamaño archivo	**Tiempo**
small	0,14
5pct	0,43
10pct	1,04
50pct	2,51
80pct	4,18
large	5,36

Graficas

Análisis

El analisis de los pasos lleva a la conclusión que el algoritmo tiene un complejidad O(n) lo cual tambien se ve reflejado en el analisis que se hace en cuanto a los tiempos por cada uno de los archivos. Se puede ver que la grafica se puede aproximar a una grafica lineal en la cual el tiempo de ejecución va en aumento conforme a que aumentan el tamaño de los archivos. Para tener que realizar un recorrido completo de ambos archivos el tiempo es optimo y se puede clasificar como bueno.

Requerimiento 3

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Límite inferior y límite superior del número de intentos.
Salidas	Numero de registros que cumplen con el rango de búsqueda. Los 3 primeros y 3 últimos registros disponibles de dicho rango.
Implementado (Sí/No)	Si se implementó. Hecho por Laura Calderón.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Se accede al índice “gamesbyTime”, que tiene los registros ordenados por el tiempo en segundos para el mejor intento realizado.	O(1)
Paso 2: Se crea una lista vacía.	O(n)
Paso 3: Se hace un recorrido inorder con el fin de ir agregando los elementos del árbol creado con el índice de tiempos dentro de una nueva lista.	O(n)
Paso 4: Se hace un doble recorrido de esta lista para ir obteniendo la cantidad total de intentos que intentan romper el récord.	O(n)
Paso 5: Se evalúa si el número de intentos es mayor al límite inferior y menor al límite superior.	O(n)
Paso 6: Si lo es, se agrega el registro a una lista final con los registros que cumplen con el rango.	O(n)
TOTAL	***O(n)***

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).

Tablas de datos

REQUERIMIENTO 3	
Tamaño archivo	**Tiempo**
small	0.77
5pct	8.45
10pct	16.30
50pct	72.09
80pct	109.01
large	146.33

Graficas

Las gráficas con la representación de las pruebas realizadas.

Análisis

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

La complejidad del requerimiento 3 fue O(n), lo que significa que su orden es lineal. Los tiempos de ejecución se tomaron a partir de los intentos 21 y 75, los cuales fueron los brindados por la guía del reto.

Requerimiento 4

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Los parámetros de entrada son: el límite inferior de la fecha que se desea consultar junto con el tiempo en el formato de YYYY-MM-DDThh:mm:ssZ y el límite superior de la fecha que se desea consultar en el formato de YYYY-MM-DDThh:mm:ssZ.
Salidas	La respuesta de esta función son el número de registros de los juegos que cumplen con el rango de búsqueda dependiendo de la fecha ingresada por el usuario. Además, los primeros 3 registros y los últimos 3 registros que cumplen con dicha condición.
Implementado (Sí/No)	Si fue implementada. Realizada por Manuela Lizcano.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Accedo a índice del catálogo que contiene el árbol RBT, por medio de “gamesbyDate”.	O(1)
Paso 2: Creo una lista vacía	O(1)
Paso 3: Realizo un recorrido inorder del árbol que se obtuvo en el paso 1. Esto con el objetivo de obtener los elementos del árbol y organizarlos en una lista.	O(n)
Paso 4: Realizo un recorrido por la lista del paso 3, para obtener únicamente los valores que cumple con la condición del rango de fechas que fue ingresado como parámetro.	O(n)
Paso 5: Se compara para cada uno de los elementos el valor en “Record_Date_0”, para ver si es mayor al límite inferior de fecha y menor que el límite superior de fecha.	O(1)
Paso 6: En caso de que el elemento cumpla con las dos condiciones del paso 5, se añade a una lista final, la cual fue creada en el paso 2.	O(n)
Paso 7: Ordeno los elementos de la lista del paso 6 dependiendo de los valores en “Time_0” y “Num_Runs”.	O(nlogn)
TOTAL	***O(n)***

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).
Para realizar las pruebas de los tiempos de ejecución se utilizaron los archivos de category_data y game_data con diferentes cantidades de datos los cuales fueron de tamaño small, 5pct, 10pct, 50pct, 80pct y large. La información fue organizada en arbolesRBT los cuales estaban ordenados dependiendo de la fecha en se rompió el récord. En el cálculo de los tiempos se utilizaron tres diferentes computadores, en donde se sacó el promedio de los resultados, esto se pudo realizar con las funciones de la librería de time de python. Todas las pruebas para este requerimiento se realizaron con los mismos parámetros de entrada en donde la fecha de límite inferior fue 2019-03-06T04:03:53Z y la fecha del límite superior fue 2021-10-17T15:48:00Z.

Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

REQUERIMIENTO 4	
Tamaño archivo	**Tiempo**
small	2,25
5pct	13,5
10pct	76,13
50pct	474,26
80pct	885,99
large	1195,57

Graficas

Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el analisis de complejidad.
El requerimiento 4 tiene una complejidad de O(n) debido a que se realizaron recorridos por las listas que fueron creadas a partir de los árboles por medio de la función de la librería inorder, esto con el objetivo de poder obtener únicamente el contenido cuya fecha de récord estuviera

entre las fechas que entraron como parámetro. A partir de la gráfica también se puede ver un crecimiento aproximadamente lineal.

Requerimiento 5

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Límite inferior de tiempos, límite superior de tiempos.
Salidas	El número total de elementos que se encuentran en el rango, los tres primeros y los tres últimos en el rango de tiempos.
Implementado (Sí/No)	Sí se implementó, realizado por Mariana Pineda Miranda.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1. Busco el índice el cual contiene los registros por medio de la fecha en el que se llevaron a cabo.	O(1)
Paso 2. Creo una lista vacía en donde se van a guardar los resultados.	O(n)
Paso 3. Realizo un recorrido inorden en el que resultan los elementos del árbol.	O(n)
Paso 4. Recorro la lista que resulta del recorrido inorden.	O(1)
Paso 5. Verifico si el tiempo del elemento en el que estoy se encuentra en el rango que entra por parámetro.	O(n)
Paso 6. Si el elemento se encuentra en el rango lo añado a la lista creada en el paso 2.	O(n)
Paso 7. Se ordena la lista por medio del algoritmo shell teniendo en cuenta el tiempo.	O(nlogn)
TOTAL	***O(n)***

Pruebas Realizadas – Con máquina 3

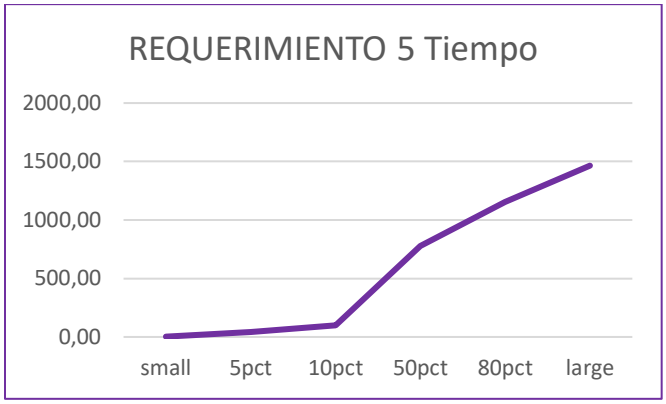
Se realizaron las pruebas de tiempo de ejecución para este requerimiento con varios de los archivos variando su tamaño, cargando los datos por medio de un catálogo en el que se creó una lista con las películas y diferentes índices dependiendo de los requerimientos. Con árboles rojo y negro para facilitar su busqueda. Se ejecutaron las pruebas en tres computadores. Las pruebas se realizaron con los diferentes archivos con diferentes tamaños. Además, se usó la librería nativa de python time. Para realizar estas pruebas se utilizaron como parámetros 542.1 como límite de tiempo inferior y 1887.5 como límite de tiempo superior, además se repitió la prueba tres veces para tener un valor más exacto del resultado.

Tablas de datos

REQUERIMIENTO 5

Tamaño archivo	Tiempo
small	2,70
5pct	40,80
10pct	98,41
50pct	777,94
80pct	1152,56
large	1464,86

Graficas



Análisis

La complejidad del requerimiento 5, fue lineal O(n) por los diferentes recorridos que se deben hacer a los datos, se puede aproximar la gráfica de los tiempos de ejecución a una función lineal, sin embargo, debido a los recorridos que se deben hacer en algunos casos se acerca a una función de segundo grado. Con el análisis de los pasos de la función también se puede decir que el algoritmo implementado tiene una complejidad de O(n)

Requerimiento 6

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	La entrada de esta función es: el límite inferior del año de lanzamiento de los juegos, el límite superior del año de lanzamiento de los juegos, el número de segmentos en los que se divide el histograma, el número de niveles en que se dividen las marcas del histograma. Adicionalmente, incluye el tipo de consulta que se desea realizar debido a que se puede hacer con los tres tiempos (“Time_0”, “Time_2” y “Time_3”) o con el promedio de los tiempos o el número de intentos que han sido registrados.

Salidas	La respuesta final de este requerimiento incluye: el número total de los registros consultados, el número total de los registros que fueron incluidos en el conteo para poder graficar el histograma, el valor mínimo y máximo de todo el rango del histograma y por último debe incluir el histograma, el cual depende del rango y de los niveles que fueron ingresados como parámetro.
Implementado (Sí/No)	Sí se implemento

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Accedo al catálogo con el índice de "gamesbyReleaseYear" que contiene el árbol RBT.	O(1)
Paso 2: Creo una lista vacía	O(1)
Paso 3: Realizo un recorrido inorder con las funciones de la librería. Esto para organizar los elementos del árbol en una lista.	O(n)
Paso 4: Realizo un recorrido en la lista creada en el paso 3, para poder comparar el valor de "Release_Date" de cada uno de los elementos.	O(n)
Paso 5: Los elementos del paso 4 que cumplan con la condición de que su fecha de lanzamiento se encuentre entre el rango de los años que fueron ingresados como parámetros, son agregados a la lista que fue creada en el paso 2.	O(1)
Paso 6: Dependiendo del tipo de consulta que fue ingresado como parámetro se ejecuta un código.	O(1)
Paso 7: Para los tres casos, primero se realiza una lista vacía.	O(1)
Paso 8: Con el objetivo de obtener los tiempos, se accede al catálogo que contiene una lista con todos los elementos de category_data.	O(1)
Paso 9: Se realiza un doble recorrido para comparar los IDs de los elementos de la lista que fue creada en el paso 5 y todos los ID que estan presentes en category_data. Esto con el objetivo de poder obtener los datos necesarios de "Time_0", "Time_1", "Time_2" y "Num_Runs".	O(n)
Paso 10: En la consulta 1, en donde el usuario desea incluir todos los elementos de tiempo, se agregan a la lista que fue creada en el paso 7 todos los elementos por separado. En la consulta 2, en donde le usuaria desea consultar los	O(n)

promedios de los tres tiempos, primero se calcula este promedio y después ese valor se agrega a la lista creada en el paso 7. Y por último, en la consulta 3, se agregan todos los valores del número de intentos.	
Paso 11: Dependiendo la consulta que fue realizada, se ordenan los datos de las tres posibles listas que fueron creadas en el paso 10.	O(nlogn)
Paso 12: Para crear el histograma, se crean dos listas para poder incluir los valores que van a representar los rangos del histograma y la otra la cantidad de datos en cada uno de los rangos.	O(1)
Paso 13: Se hace un doble recorrido por las listas del paso 10 y los segmentos que fueron ingresados por el usuario. Esto con el objetivo de poder contabilizar la cantidad de valores que se encuentre entre un rango, dichos datos se agregan a la segunda lista del paso 12.	O(n)
Paso 14: Se crea una lista vacía	O(1)
Paso 15: Se hace un recorrido por la lista creada en el paso 13 para poder determinar el número de marcas por cada rango dependiendo del valor de niveles que fue ingresado por parámetro.	O(n)
TOTAL	***O(n)***

Pruebas Realizadas – Con máquina 4

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).

Se realizaron las pruebas de los tiempos de ejecución utilizando diferentes archivos de datos los cuales contienen diferentes cantidades de datos, entre estos se encontraba el tamaño small, 5pct, 10pct, 50pct, 80pct y large. Para facilitar el uso de la información, se crear arboles RBT los cuales estaban ordenados por el año de publicación del juego. Para el cálculo del tiempo se utilizaron funciones de la librería de time de python, y estas pruebas de tiempo fueron realizadas en los tres computadores y se sacó un promedio de los tiempos. Las pruebas para este requerimiento se realizaron con los mismos parámetros de entrada en donde el año del límite inferior fue 2017, el año del límite superior 2021, el número de segmentos fue 5, el número de niveles fue 7 y por último la consulta se realizó para el promedio de los tiempos.

Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

REQUERIMIENTO 6	
Tamaño archivo	**Tiempo**
small	5,2
5pct	52,9

10pct	127,4
50pct	1754,63
80pct	2636,51
large	3227,4

Graficas

Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el analisis de complejidad.
El requerimiento 6 tiene una complejidad de O(n), esto debido a que se realizan recorridos entre las listas de los datos. En la gráfica se puede observar un crecimiento aproximadamente lineal, considerando la carga de datos de 50pct, 80pct y large.

Requerimiento 7

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	Plataforma de la cual se quieren encontrar los videojuegos mas rentables y un entero el cual es el top de videojuegos mas rentables.
Salidas	Total de videojuegos presentes en la plataforma, videojuegos con la mayor rentabilidad en la plataforma.
Implementado (Sí/No)	Si se implemento

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**

Paso 1. Buscar en el catalogo el indice en el que se guarda el arbol con los videojuegos dependiendo de la plataforma.	O(1)
Paso 2. Coincidir por medio de la función om.get() la plataforma entrada por parametro con el arbol correspondiente.	O(n)
Paso 3. Calcular la antigüedad teniendo en cuenta el año en el que el videojuego fue lanzado, se recorre la lista correspondiente a los videojuegos de la plataforma y se busca el año de lanzamiento.	O(n)
Paso 4. Teniendo ya el año de lanzamiento se calcula la antigüedad teniendo en cuenta si el videojuego salio luego del 2018, entre 2018 y 1998 o salio luego de 1998.	O(1)
Paso 5. Se calcula la popularidad con el mismo recorrido del Paso 3. Sacando el logaritmo natural por medio de la librería math.	O(1)
Paso 6. Para calcular el promedio de tiempos se recorre la lista que se encuentra en el catalogo con los registros de tiempos de cada videojuego, y se coincide el id en el primer archivo con el que se esta recorriendo.	O(n)
Paso 7. Se crea una lista con los tiempos de cada uno de los videojuegos.	O(1)
Paso 8. Se calcula el tiempo promedio teniendo en cuenta el numero de elementos en la lista.	O(1)
Paso 9. Para encontrar el revenue se multiplica el promedio de tiempos con la popularidad del juego y se divide entre la antigüedad.	O(1)
Paso 10. Para encontrar el market revenue se compara cada videojuego con el total de veces que ese videojuego aparece en la lista de registros.	O(n)
Paso 11. Para sacar la rentabilidad total de cada videojuego se multiplica la rentabilidad obtenida en el Paso 9 con la representación en el mercado en el Paso 10.	O(n)
Paso 12. Se organiza la lista por medio del algoritmo Merge usando la rentabilidad como criterio de comparación.	O(nlogn)
Paso 13. Dependiendo del top de elementos que se hayan ingresado se incluyen a la lista los elementos indicados por el usuario.	O(n)
TOTAL	***O(n)***

Pruebas Realizadas – Con máquina 3

Se realizaron las pruebas de tiempo de ejecución para este requerimiento con varios de los archivos variando su tamaño, cargando los datos por medio de un catálogo en el que se creó una lista con las películas y diferentes índices dependiendo de los requerimientos. Con arboles rojo y negro para facilitar su busqueda. Se ejecutaron las pruebas en tres computadores. Las pruebas se realizaron con los diferentes archivos con diferentes tamaños. Además, se usó la librería nativa de python time. Para realizar estas pruebas se uso la plataforma PC y se quiso encontrar el top 5 de juegos mas rentables, ademas se repitió la prueba tres veces para tener un valor más exacto del resultado.

Tablas de datos

REQUERIMIENTO 7	
Tamaño archivo	**Tiempo**
small	12,14
5pct	280,15
10pct	892,66
50pct	6470,22
80pct	11449,57
large	14250,72

Graficas

Análisis

En este caso, se puede observar que los tiempos de ejecución aumentan de manera acelerada, esto se da debido a que para poder sacar los datos necesarios de este requerimiento se necesita recorrer algunos de los datos varias veces, sin embargo nunca se recorre toda la lista de datos mas de una vez. En el analisis de los pasos tambien se puede evidenciar lo anterior y se ve que al aumentar el tamaño de los archivos se aumenta tambien este tiempo de ejecución. Los tiempos en archivos grandes son algo elevados por lo que se podria buscar un algoritmo mas efectivo con respecto al tiempo de ejecución.

Requerimiento 8 - BONO

Plantilla para el documentar y analizar cada uno de los requerimientos.

Descripción

Breve descripción de como abordaron la implementación del requerimiento

Entrada	El usuario debe ingresar el año de publicación sobre el cual quiere obtener los histogramas, junto con el límite inferior de la duración del mejor tiempo y el límite superior de la duración del mejor tiempo.
Salidas	El número total de registros de speedrun en dicho año y rango. Un mapa interactivo de clústeres que muestra todos los registros de speedrun en dicho año y rango de tiempos.
Implementado (Sí/No)	Si se implementó

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Paso 1: Se accede al índice “gamesbyTime”, que tiene los registros ordenados por el tiempo en segundos para el mejor intento realizado y al índice “”gamesByRelease_Date”.	O(1)
Paso 2: Se crean dos listas vacías, una que tendrá los id de los juegos y otra para los países.	O(1)
Paso 3: Se hace un recorrido inorder con el fin de ir agregando los elementos del árbol creado con el índice de tiempos dentro de una nueva lista.	O(n)
Paso 4: Se hace un recorrido inorder con el fin de ir agregando los elementos del árbol creado con el índice de fechas de publicación dentro de una nueva lista.	O(n)
Paso 5: Se crea un mapa del mundo utilizando la librería folium.	O(1)
Paso 6: Se recorre la lista de los juegos, en donde se saca el año de publicación del juego, se revisa si este es igual al año ingresado, y si si lo es, se agrega a la lista creada anteriormente los id de los juegos que corresponden al año solicitado.	O(n)
Paso 7: Se recorre la lista de los id de los juegos, la lista de las categorías creada al principio, y los elementos de la lista de dichas categorías.	O(n)

Paso 8: Se compara que el game id del registro a evaluar coincida con uno de los game id de la lista que se tenía, también que el tiempo sea mayor al límite inferior del tiempo y que el tiempo sea menor al límite superior del tiempo.	O(1)
Paso 9: Si lo anterior se cumple, se agrega a la lista vacía de países creada al inicio, el país del registro que cumple con dichos criterios.	O(1)
Paso 10: Se crea una variable a la cual se le asigna Nominatim (función que permite sacar coordenadas).	O(n)
Paso 11: Se recorre la lista final de países, en donde se utiliza la librería geopy para obtener la longitud y latitud dado un país específico.	O(n)
Paso 12: Se utiliza folium marker para ir marcando en el mapa las ubicaciones de los países de la lista, con las latitudes y longitudes brindadas por geopy.	O(n)
Paso 13: Se guarda el mapa en un archivo html.	O(n)
TOTAL	***O(n)***

Pruebas Realizadas – Con máquina 3

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).

Tablas de datos

REQUERIMIENTO 8 - BONO	
Tamaño archivo	**Tiempo**
small	1091.52
5pct	14865.47
10pct	30579.03
50pct	165314.744
80pct	297271.42
large	443144.54

Graficas

Análisis

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el análisis de complejidad.
La complejidad del requerimiento 8 es lineal O(n) debido a que no se recorrieron la totalidad de los datos brindados por los archivos de game y category. Las pruebas de ejecución se realizaron con el año 2017, y los tiempos 1000 y 5000. Con esto se obtuvo el mapa con las respectivas ubicaciones que permiten que el usuario conozca los países a los que pertenecen los juegos del año y del rango ingresado.