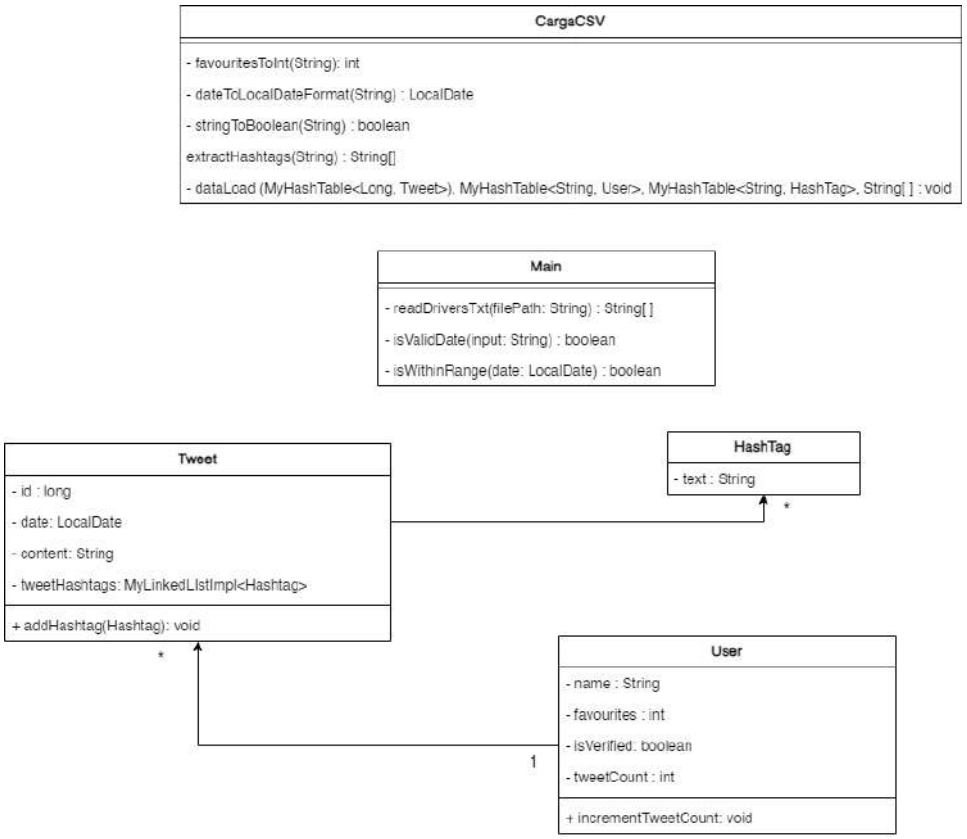
Grupo 20 - Obligatorio Programación 2

Justino Rodriguez y Ramiro Martinez

UML Final



Carga de Datos

Utilizamos una librería llamada Apache Commons CSV que permite leer el archivo csv con la función FileReader y poder recorrer los records almacenados en el mismo. Implementamos funciones que "limpian" la información:

- favouritesToInt convierte los datos de favoritos a double y luego a integer dado que hay valores con punto decimal.
- dateToLocalDateFormat convierte el string que contiene el detalle de fecha y hora a formato LocalDate para guardar solamente la fecha.
- stringToBoolean convierte el string "True" o "False" a un valor booleano.
- extractHashtags limpia el "array" (String) que contiene los hashtags de un tweet y crea un String[] array en minúscula para luego guardarlo en cada tweet.

dataLoad es la función que hace la carga, para cada fila leída correctamente del csv aplicamos para las columnas que necesitamos las funciones correspondientes y guardamos los valores procesados. Una vez realizado esto, instanciamos los usuarios, tweets y hashtags que almacenamos en HashMaps. También llenamos un array con los nombres de usuario para poder iterar sobre el hash de *users* para las funciones 2 y 5. El hash *users* tiene como key el nombre de usuario y el objeto User como value. El hash *tweets* tiene key el id del tweet y value el Tweet y el de hashtags tiene como clave el String del texto del hashtag y valor el objeto Hashtag.

El principal motivo de guardar los usuarios y hashtags en Hashmaps es porque es muy fácil saber si estamos enfrentando a un nombre de usuario o texto de hashtag ya existente. Los tweets fueron también guardados en un Hashmap aunque tengan una key autoincremental para hacer la carga eficiente (con MyLinkedList no terminaba más).

Asunciones:

- nombre del usuario único. A pesar de que es posible que dos cuentas distintas puedan tener el mismo nombre en la red social Twitter, dado que el dato proporcionado es un campo abierto a llenar por el usuario, decidimos esto dado que al buscar en Twitter la cantidad de usuarios con mismo Nombre (**no @usuario**) las otras cuentas estaban prácticamente inactivas.
- favoritos (likes) aumentan con el tiempo. En la carga de datos, cuando había coincidencia por nombre del usuario, comparamos el número de favoritos y, si este era mayor, actualizamos dicho campo. En una red social lo normal es que los likes de una cuenta aumentan a lo largo del tiempo (a medida que "interactúa")
- case-insensitive para hashtags. Al fijarnos en el buscador de Twitter para hashtags, identificamos que Twitter adopta una postura case-insensitive para los mismos y decidimos seguirlo para nuestros reportes. Sí fuimos sensibles al uso de tildes y diéresis (es posible que esto haya afectado las apariciones de (Sergio Pérez Y Nico Hülkenberg).

Implementación de Reportes

1. *10 pilotos activos en la temporada 2023 más mencionados en los tweets en un mes*

Leemos el archivo txt que contiene los nombres de los pilotos y lo almacenamos en un array como Strings. Recorremos el array y luego recorremos el **hash** de tweets chequeando que el mes y año coincidan con el ingresado y que el contenido menciona al piloto (siendo case insensitive pero sensible a tildes/diéresis). Para cada piloto llevamos un contador que aumenta según si aparece el piloto. Al final de cada recorrido de tweets por piloto guardamos el contador y nombre del piloto en un **heap** que permite ordenar por una key (contador) y almacenar un valor (nombre del piloto). La naturaleza del **heap máximo** ordena estas claves cada vez que se ingresa un "Pair" <contador, nombrePiloto>. Para sacar los pilotos más mencionados hacemos un removeMax 10 veces (para cada remove el heap contiene la mayor cuenta como raíz).

2. *Top 15 usuarios con más tweets*

En la carga de datos guardamos un array conteniendo los nombres de usuario (sin repeticiones). Esto nos resultó muy útil para recorrer el hash recorriendo este array y llamando a la funcion get de hash. Guardamos en un **heap máximo** la cantidad de tweets y el objeto User. Al guardar en un hash con key el nombre de usuario podemos acceder rápidamente al objeto User y su respectivo tweetCount. Removemos 15 veces el Pair <tweetCount, User> que guardamos en el heap (se ordena por tweetCount). Era necesario guardar el objeto User para poder tener la información de si está verificado o no.

3. *Cantidad de hashtags distintos para un día dado*

Hacemos un loop sobre el hash de tweets verificando si cumple con la fecha. De ser apropiada la fecha accedemos a la lista de hashtags (si no es nula) que almacena como **MyLinkedList** y vamos guardando en un hash *usedHashtagsHash* el hashtag si no había aparecido antes. Decidimos usar un Hash por la propiedad del método contains qué eficientemente nos dice si ese hashtag ya apareció. Si el hashtag no había aparecido lo guardamos en el hash y llevamos un contador por cada vez que hacemos un put en el hash. Lo único que nos importa es la key de hash, en el value almacenamos el contador temporal pero era para rellenar. Una vez iterado sobre todos los tweets imprimimos el contador *hashtagsDistintos*.

4. *Hashtag más usado para un día dado, sin tener en cuenta #f1*.

Utilizamos dos hashes: uno para filtrar los tweets por día y almacenarlos con un nuevo autoincrement *fakeID;* el otro para poder nuevamente construir una lista con unicidad de hashtags que además ya están filtrados por el día al acceder desde el hash de tweets filtrados. En el hash de hashtags guardamos como key el String hashtag y como value una clase Pair que nos permite ya preparar una estructura para migrar a un heap máximo y poder ordenarla. Pair guarda <Integer aparicionesDelHashtag, String hashtagText >. Al finalizar la iteración tenemos un hash que contiene las apariciones del hashtag en la key de Pair. Usando la función *.keys()* de Hash podemos tener una LinkedList para recorrer y acceder al value Pair y almacenarlo en un heap. Para sacar el hashtag más usado para el día llamamos a la función *.removeMax()* de Heap que nos devuelve el Pair <apariciones,hashtag> y poder acceder a los dos valores para imprimirlo.

5. *Top 7 cuentas con más favoritos*

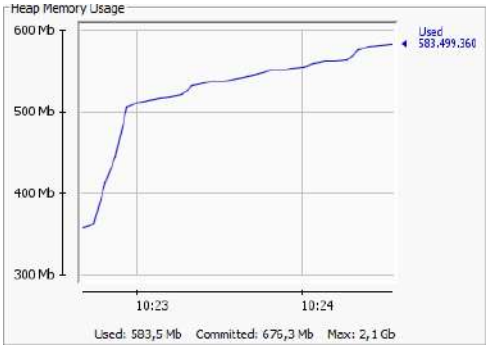
Volvimos a ayudarnos del Heap Máximo como herramienta de ordenamiento. Recorremos el array de usuarios y usando la funcion get(key) de Hash podemos obtener la cantidad de favoritos por usuario y los guardamos en un heap, en este caso con el Pair <userFavourites, userName> dado que con tener el username nos basta. Guardamos los Pair en un heap con la key de cantidad de favoritos. Al ir removiendo del heap máximo obtenemos la informacion de la cantidad de favoritos y usuario sabiendo que esa cantidad es máxima dentro del heap. Recordar que favoritos es el valor máximo de favoritos para ese usuario en todo el dataset.

6. *Cantidad de tweets con una palabra o frase específicos*

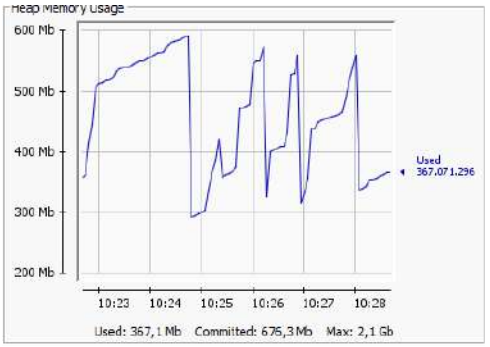
Recorremos el hash de tweets (fácil porque la key es autoincrement) y verificamos que el contenido incluye la palabra o frase ingresada (case insensitive). Esta función puede no ser del todo fiable porque si buscamos (“car” puede venir de la misma palabra “car” como de “racecar”, “cargador”, “cartoon”, “carbonero” que puede traer inconsistencias).

Medición de eficiencia - Cantidad de memoria RAM consumida

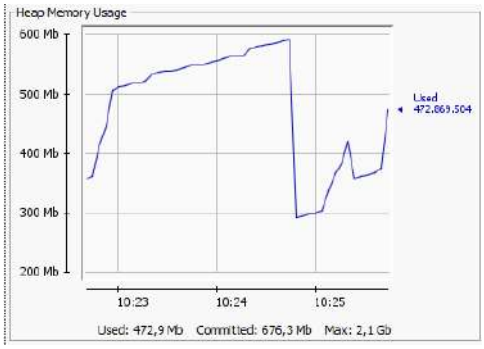
Carga de datos:



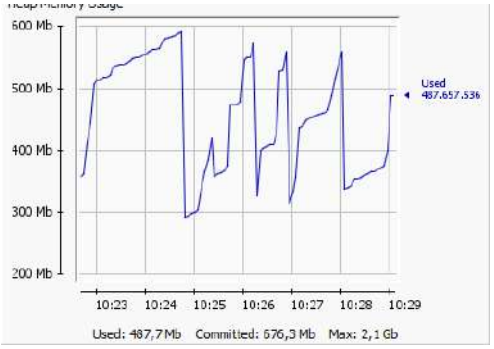
Hashtag más usado en un dia (2021-12-12):



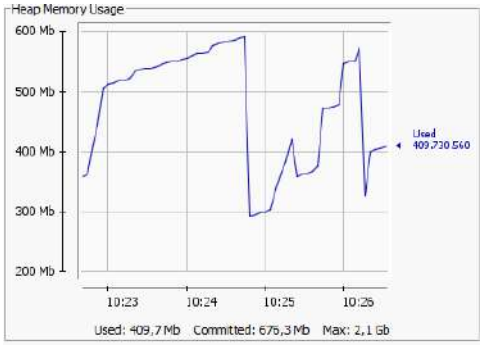
Listar 10 pilotos (12-2021):



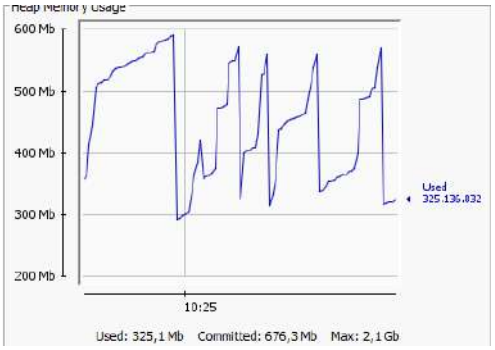
Top 7 cuentas con mas favoritos:



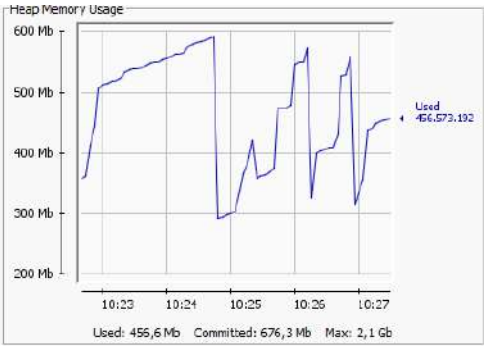
Top 15 más tweets:



Cantidad de tweets con una frase específica (“race”):



Cantidad de Hashtags distintos en un dia (2021-12-12):



ii. Tiempos de ejecución

Para la función de pilotos más mencionados tomamos el mes de diciembre de 2021. Para las funciones que dependen de un día particular elegimos el 12-12-2021 dado que era la **final del calendario 2021** y es posible que comparado con otros días haya sido de mayor relevancia en términos de datos e interacciones.

Función	**Tiempo de ejecución (milisegundos)**
Carga de datos	*7085*
10 pilotos más mencionados	*2443*
Top 15 usuarios con más tweets	*25*
Cantidad de hashtags distintos para un día	*60*
Hashtag más usado para un día	*464*
Top 7 cuentas con más favoritos	*14*
Cantidad de tweets con palabra/frase	*337*