Obligatorio 2023

Una casa de apuestas en Inglaterra está constantemente ajustando sus dividendos para apuestas, y quiere incorporar los resultados de carreras de Formula 1 a su oferta de apuestas deportivas. Para ello, la casa se basó en un dataset público con más de medio millón de tweets desde julio de 2021 a agosto de 2022 sobre Formula 1 para obtener una idea preliminar de la opinión pública de los fanáticos sobre los pilotos.

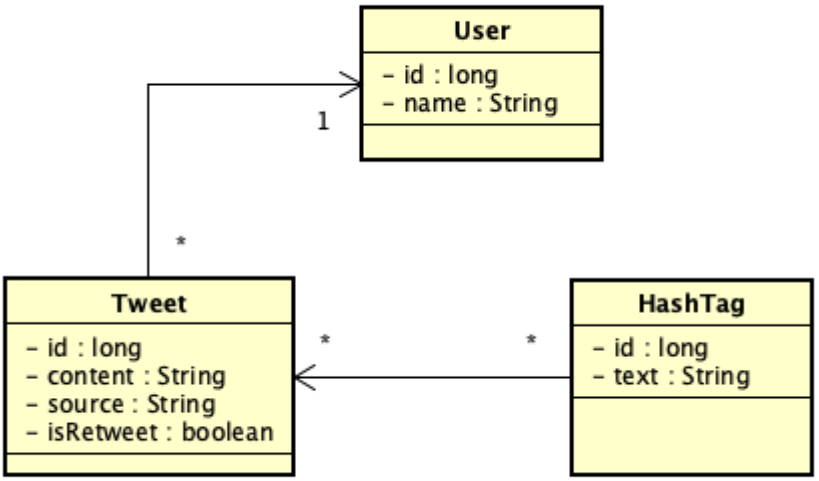
Para poder realizar un análisis inicial sobre cuáles serán los dividendos en las próximas carreras, se desea generar los siguientes reportes a partir del dataset:

- Listar los 10 pilotos **activos** en la temporada 2023 más mencionados en los tweets en un mes (este mes será ingresado como 2 parámetros separados, mes y año, por consola). Este listado debe incluir el nombre de los pilotos y la cantidad de ocurrencias para cada uno de manera ordenada. Se espera que esta operación sea de orden n en notación Big O.
- Top 15 usuarios con más tweets. Este listado debe incluir el nombre de usuario, la cantidad de tweets, y si el usuario está verificado o no, ordenado por cantidad de tweets en orden descendente. Se espera que esta operación sea de orden n en notación *Big O*.
- Cantidad de hashtags distintos para un día dado. El día será ingresado en el formato YYYY-MM-DD.
- Hashtag más usado para un día dado, sin tener en cuenta #f1. El día será ingresado en el formato YYYY-MM-DD.
- Top 7 cuentas con más favoritos. Para este listado se deberá retornar el nombre del usuario, junto con la cantidad de favoritos.
- Cantidad de tweets con una palabra o frase específicos (que será ingresado como parámetro).

Se debe realizar una aplicación Java SE que a través de la entrada estándar permita:

1. Realizar la carga masiva de datos a través de un archivo CSV. Puede asumir que el mismo se encuentra en la raíz del proyecto.
2. Realizar un menú a través de la entrada estándar, que permita la ejecución de los 5 reportes.

Para representar la realidad del problema, se propone el siguiente diagrama UML **base**:



A su vez, se debe realizar un documento, que contenga:

1. Diagrama UML de clases de la solución.
2. Breve descripción de los procesos de carga y realización de reportes.
3. Medición de eficiencia de la aplicación. Para ello se deberá indicar, tanto para la carga como para la ejecución de los reportes:
    a. Cantidad de memoria RAM consumida
    b. Tiempo de ejecución promedio.

Este trabajo consta de dos entregas:

1. La primera entrega consiste en generar un repositorio remoto GIT dentro del proveedor GitHub (https://github.com/), y se deberán subir los TAD que luego van a ser usados para resolver el obligatorio. Cada equipo deberá ponerse de acuerdo en cual implementación usar en base a las realizadas durante el curso.

    Como mínimo se deberá tener los siguientes TADs disponibles en dicho repositorio (incluyendo interfaces, excepciones que estos utilicen, y tests unitarios que demuestren el correcto funcionamiento de los mismos):

    - Lista Enlazada Simple
    - Stack
    - Queue
    - Árbol Binario de Búsqueda
    - Heap
    - Hash Cerrado

La fecha límite para esta entrega es el martes 6/6/2023 hasta las 23:59. Se debe entregar vía moodle. Esta entrega deberá contener un enlace con una invitación para ver el repositorio, incluyendo los nombres de ambos integrantes del equipo en una única entrega.

2. La segunda entrega consta de un zip con los contenidos del repositorio, incluyendo un archivo README.txt describiendo los procesos de carga, las decisiones tomadas y el consumo de memoria en cada reporte. El método de entrega es el mismo que para la primera parte del obligatorio una tarea en moodle. La fecha límite para entregar esta parte es el 23/6/2022.

Consideraciones generales:

- El proyecto que consolida los TADs se tiene que llamar “grupoX-p2-tads”, donde X debe sustituirse con el número de grupo asignado.
- Dentro del proyecto se debe disponer de 2 folders: “src”, conteniendo el código fuente de sus TADs y “test” para el código fuente de sus test unitarios. Un ejemplo de esta división puede ser vista en el proyecto de TADs que se utilizó en el primer parcial.
- Los TADs deben estar bajo el paquete uy.edu.um.prog2.adt.
- Se debe utilizar los TADs realizados durante el curso y NO es posible en esta instancia utilizar TADs de terceros (brindados por la máquina virtual de Java u otros).
- Se debe crear otro repositorio GIT para la segunda parte del obligatorio. Se va a verificar que todos los participantes hayan realizado commits sobre los proyectos. Se va a penalizar en caso de que esto no suceda.
- Se deberá elegir la estructura más adecuada para resolver cada problema.
- No se admiten estructuras intermedias que almacenen los resultados de las consultas. Se deben realizar los cálculos en el momento en que se solicita correr una consulta.

A continuación se deja el link para descargar el set de datos que va a utilizarse, junto con un archivo .txt conteniendo los nombres de los pilotos activos esta temporada:

- https://www.dropbox.com/s/hhe191srrvi66sr/obligatorio2023.zip?dl=0

El mismo contiene 3 archivos:

- f1_dataset.csv, archivo en formato CSV con la fuente de datos completa.
- f1_dataset_test.csv**,** archivo en formato CSV con la fuente de datos reducida para test.
- drivers.txt, archivo en formato TXT que **debe ser leído desde el programa Java** para cargar los nombres de los pilotos activos. **No se considerará válido tener una lista en memoria con los nombres de los pilotos.**

Links de interés:

- Pro GIT Book - Capitulo 1 y 2 como punto de partida.
    - https://git-scm.com/book/en/v2
- Laboratorio práctico de Github para práctica los comandos básicos con un simulador de líneas de comando (15 minutos).
    - https://try.github.io/levels/1/challenges/1