Tarea 2

MyAnimeList es uno de los portales con mayor información de Anime y Manga que existe en el mundo y a usted, como un gran fanático del Anime, le gusta ver las series que posean la mayor calificación posible. Es por esto que utilizando la gran base de datos de esta página, se necesita ordenar las series pertenecientes a un género dado en base a su puntuación en un árbol ABB, AVL y 2-3

Cada uno de los árboles debe poseer métodos para insertar, buscar y eliminar un anime, un método que permita imprimir la lista de anime en base a su puntuación, conocer el animé con mejor puntuación y el animé con peor puntuación.

Se debe, además de la implementación de cada una de las estructuras, crear un script que permita escoger al usuario qué tipo de animé desea consultar y además permita ejecutar las distintas opciones que el árbol debe poseer.

Junto al código, deberá realizar un pequeño análisis comparativo sobre el tiempo de ejecución de las distintas estructuras al insertar, buscar y eliminar un animé. Para esto puede hacer uso de la librería `Time` (https://docs.python.org/3.6/library/time.html). Puede incluir en su análisis los ordenes de complejidad teóricos de cada una de las funciones que tienen las estructuras para así demostrar sus resultados.

Requisitos

Para poder realizar esta tarea, debe tener instalado en su sistema:

- Python versión > 3.4
- Jikanpy (https://github.com/AWConant/jikanpy)

Documentación API

La documentación de la API la pueden encontrar en https://jikan.docs.apiary.io/ .

Ejemplo

Por ejemplo, si queremos obtener los anime correspondiente al genero "acción" (genre_id=1)

```
from jikanpy import Jikan
jikan = Jikan()

# Obtiene los anime correspondientes a genre_id, 1=action
action = jikan.genre(type='anime', genre_id=1)
# Imprime cada anime con su respectiva puntuación
for anime in action["anime"]:
    print(anime["title"], anime["score"])
```

Entrega

Fecha límite: Viernes 7 de Diciembre a las 23:59 hrs.

Para la entrega de la tarea deberá crear un repositorio en el sitio GitHub (github.com) y enviar el link del repositorio al mail `john.bidwell@mail.udp.cl` con el asunto `Tarea EDD xxx` (donde `xxx` corresponde al nombre del alumno). Dicho repositorio deberá contener cada uno de los archivos de código en formato `.py` y el informe de análisis en formato `.pdf`.

Tendrá puntos extra si hace un buen uso de git, es decir, hace commits para identificar los cambios que va haciendo a su código.

Recomendaciones

Haga la tarea con tiempo.

Utilice las pautas vistas en la primera ayudantía para escribir bien su código.

Utilice un archivo `.py` para cada estructura de manera que se mantenga el código organizado.

Git

Puede utilizar GitHub a través de:

- GitHub Desktop
- Terminal
- Página web

Guía sencilla para aprender Git http://rogerdudler.github.io/git-guide/index.es.html

Se recomienda realizar un commit para identificar cada uno de los cambios que hace en su código.

Este sería un ejemplo del flujo de trabajo que deben hacer con Git:

```
git init # Iniciliza un repositorio en la carpeta que nos encontremos
git add . # Añade todos los archivos, pueden utilizar git add xxx para
añadir un archivo en específico
git commit -m "Mi primer commit"
git push origin master
```

Luego al hacer un cambio:

```
git add "AVL.py"
git commit -m "Implementado AVL"
```

```
git add "hash.py"
git commit -m "Implementado hash"
```

```

```

```
git add "AVL.py"
git commit -m "Fix en función insertar para AVL"
```

...