Preguntas del Laboratorio #1

1. ¿Cuál es el ciclo regular para actualizar código en un repositorio GIT?

Lo primero es poner el repositorio en el IDE; luego se crea una rama en la cual se aplican las respectivas ediciones del código; posterior a eso, se hace un merge con la rama master y se verifica que no haya ningún tipo de error de compatibilidad, y finalmente con una pull request se actualiza el repositorio GIT

2. ¿Qué ventajas y limitantes tiene el uso de Ramas/Branches?

Las branches pueden permitir el trabajo simultaneo de varios en un mismo programa, facilitando el trabajo de código en un ambiente remoto, sin embargo, para trabajar con las nuevas versiones del programa y las branches que pasaron a ser parte de la master, se requiere descargar otra vez el archivo, lo que puede llegar a ser molesto en algunos casos

3. ¿Cuáles serían las buenas prácticas para solucionar conflictos?

Lo ideal serie revisar ambas partes del proyecto (el código previo al merge y posterior al merge) para de esa forma poder analizar que parte es la que está generando un conflicto con el código master, además, la documentación de los cambios puede ser de ayuda a la hora de revisar algún conflicto con el código.

4. ¿Qué orden de complejidad tendría las funciones (consulta y lectura de archivo)?

La función que se encarga de cargar el archivo para su uso posterior tendría una complejidad aproximada de O(n) ya que obligatoriamente deberá recorrer el archivo hasta su limite de n elementos. Mientras que la función de búsqueda de película por director puede llegar a tener una complejidad O(2n) en su peor caso, pues si el elemento de búsqueda estuviera localizado en la posición n, el algoritmo tendría de recorrer toda la selección de archivo hasta 2 veces (una para la búsqueda de director y otra para la búsqueda de las películas)

5. ¿Cómo podría reducir o aumentar la complejidad de la consulta?

Talvez la mejor forma de reducir la complejidad podría estar en las posiciones de los datos, pues estos están ubicados en el mismo lugar en ambos archivos. Una posible solución sería guardar la posición de las películas en una lista y luego usar esos valores para ir directamente en el segundo archivo, evitando así la revisión de la misma información 2 veces

6. ¿Cómo afecta un TAD en la complejidad?, ¿Qué alternativas existen?

Un TAD permite generalizar a algún tipo de datos con amplia información (en nuestro caso, las películas, sus atributos y características) y transformarlo en algo mucho mas fácil de manejar (una agenda con el id de cada película y sus datos respectivos). Esto permitió que la elaboración de las funciones fuese mas sencilla, pues solo se debía buscar el id de una película y de esa forma hallar su director, su calificación, genero, etc.

Una alternativa podría ser el crear una clase llamada “película” y otra llamada “biblioteca”. La primera tendría la función de convertir cada línea del csv en una película, y la segunda las organizaría como una biblioteca en donde se puede tener acceso fácil solo buscando alguna categoría, ya sea el género, el director, los actores o el nombre. Todo esto creado por funciones para simplificar el código final.