1. ¿Cómo implementaría la función? ¿Qué orden de complejidad tendría la función implementada?

Implementaríamos la función a través de un algoritmo con 4 parámetros de entrada, los cuales son: Criteria (nombre del director) ingresado por el usuario, la columna con los nombres de los directores y las dos listas con los datos. En primer lugar verificaríamos que ninguna lista estuviera vacía, acto seguido inicializaríamos las variables de “contador” y de “suma”. Posteriormente, buscaríamos en el archivo “Casting” los ids del director solicitado por el usuario mediante un ciclo y con estos ubicaríamos en el archivo “Movies Details” las películas buenas que dirigió según el criterio establecido, esto con un ciclo interno desde el ya mencionado. Al encontrar una película de interés, se le adicionaría 1 al contador y a su vez la calificación a la variable suma. Finalmente, asignaríamos una variable llamada “prom” para la división de suma entre contador. El algoritmo retornaría una cadena que contiene el número de películas buenas junto con la calificación media del director.

El algoritmo propuesto por el grupo presenta un orden de complejidad cuadrática en la notación BigO: O(n²).

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

El ciclo regular para actualizar código en un repositorio GIT es guardar el trabajo en el repositorio local con <git commit> y después utilizar el comando <git push origin> para subir los cambios al repositorio remoto.

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

El uso de GitHub para los proyectos grupales resulta de gran aprovechamiento debido a que este permite la sincronización del código de los múltiples miembros del equipo en un solo lugar. Lo cual facilita el proceso de desarrollo mediante el sistema de ramas que da lugar a la sincronización de las versiones individuales del código y su publicación conjunta para el análisis del grupo. Sin embargo, la actualización del código debe ser consultada con los otros miembros del equipo, ya que si dos o más personas se encuentran trabajando en una rama, el comando <git pull> podría borrar el progreso del otro integrante en la misma rama. Además, VS Code no ofrece una herramienta gráfica para ver las diferentes ramas y organización del repositorio.

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

Para solucionar los conflictos se debería consultar con los otros miembros del equipo para actualizar los cambios del código, y decidir entre todos cual es la versión óptima para subir a la rama master. Además, se aconsejaría graficar manualmente la organización del repositorio para evitar confusiones.

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

La complejidad de las funciones de consulta exceptuando la número 4 es de n.

La lectura de los archivos tendría una complejidad de 2n al ser realizada con dos archivos diferentes desde la misma función.

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

La complejidad podría aumentar a través de tareas innecesarias, como por ejemplo eliminar los directores diferentes al solicitado y las películas que no cumplan con el criterio de búsqueda. También se podría presentar falta de entendimiento de los conceptos del lenguaje de programación, agregando un nuevo ciclo al ciclo interno para calcular el promedio de las películas buenas del director.

La complejidad podría disminuir si una lista fuera la que entrara por parámetro en lugar de dos, ya que los ids solamente se referenciarían una vez y se podría buscar el nombre del director y la valoración de las películas, en un mismo archivo.

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

Un TAD disminuiría la complejidad ya que nos permite variar la estructura de datos con la que vamos a trabajar según nuestra necesidad, como por ejemplo para crear el tipo abstracto, ingresar información y obtener información.