Sección 6, Grupo 3:

Blanca Valencia Aguirre, 202013797 [b.valenciaa@uniandes.edu.co](mailto:b.valenciaa@uniandes.edu.co)

Sergio Arango Arango, 201921814, [s.arangoa@uniandes.edu.co](mailto:s.arangoa@uniandes.edu.co)

**Complejidad de los requerimientos:**

**Req #1:**

En orden de como se van ejecutando las funciones internas la complejidad es de:

N + M + K + JLogJ + J + RLogR . Donde N>M>K>J>R

Los valores disminuyentes son porque las funciones van produciendo sublistas que son de tamaño menor a la original por lo tanto la cantidad de datos a entrar en la siguiente función es variablemente menor. La creación de sublistas es de complejidad lineal porque consiste en un recorrido total de los datos con cantidad de operaciones constantes por dato (procesarlo y añadirlo a la sublista con la info deseada). Crear una sublista de los N datos originales tiene complejidad N, y retorna una cantidad de datos M<N. Hacer otra vez una sublista por categoría sobre los M datos tiene complejidad nuevamente M y resulta en K<M datos. Sigue hacer una sublista sobre los K datos por país (complejidad K) y retorna J<K datos. En la función ObtenerVideos distintos se hace un ordenamiento alfabético por video\_id (asumiremos que se hace con MergeSort) y tiene complejidad JLogJ y esta función vuelve a hacer una sublista colapsando los videos por video\_id lo cual tiene complejidad J y retorna R<J datos. Finalmente, como respuesta se vuelven a ordenar por views, pero esta vez los R datos finales, se vuelve a asumir que el algoritmo es MergeSort por lo que tiene complejidad RLogR. Dado que los criterios por los que se crean las sublistas pueden hacer variar bastante los tamaños relativos de M,K,J,R, solo se puede concluir que la complejidad es menor o igual a N.

**Req #2:**

N + MLogM + M + K . Donde N>M>K

Primero se crea una sublista por país de los N datos (complejidad N) para obtener M datos. Luego se hace la función de obtener distintos que hace un ordenamiento (MLogM) y luego retorna sublista (complejidad M) colapsando los videos por video\_id resultando en K datos. Finalmente hace un recorrido completo sobre los K datos para ver cuál tiene el mayor atributo ‘repeticiones’, notar que en este caso como solo busca al mayor y no a los n mayores, sería ineficiente primero ordenar para luego retornar una sublista de tamaño 1, un recorrido lineal basta para encontrar a el máximo.

**Req #4:**

En orden de como se van ejecutando las funciones internas la complejidad es de:

N + M + K + JLogJ + J + RLogR . Donde N>M>K>J>R

La complejidad de este algoritmo es igual a la del requerimiento #1.

Los valores disminuyentes son porque las funciones van produciendo sublistas que son de tamaño menor a la original por lo tanto la cantidad de datos a entrar en la siguiente función es variablemente menor. La creación de sublistas es de complejidad lineal porque consiste en un recorrido total de los datos con cantidad de operaciones constantes por dato (procesarlo y añadirlo a la sublista con la info deseada). Crear una sublista de los N datos originales tiene complejidad N, y retorna una cantidad de datos M<N. Hacer otra vez una sublista por país sobre los M datos tiene complejidad nuevamente M y resulta en K<M datos. Sigue hacer una sublista sobre los K datos por tag (complejidad K) y retorna J<K datos. En la función ObtenerVideos distintos se hace un ordenamiento alfabético por video\_id (asumiremos que se hace con MergeSort) y tiene complejidad JLogJ y esta función vuelve a hacer una sublista colapsando los videos por video\_id lo cual tiene complejidad J y retorna R<J datos. Finalmente, como respuesta se vuelven a ordenar por likes pero esta vez los R datos finales, se vuelve a asumir que el algoritmo es MergeSort por lo que tiene complejidad RLogR. Dado que los criterios por los que se crean las sublistas pueden hacer variar bastante los tamaños relativos de M,K,J,R, solo se puede concluir que la complejidad es menor o igual a N.