**Análisis Reto 3**

Juan David Aparicio Gutiérrez  
202116532  
[j.apariciog@uniandes.edu.co](mailto:j.apariciog@uniandes.edu.co)  
Requerimiento 3

Paula Cecilia Daza Díaz  
202111276  
[p.dazad@uniandes.edu.co](mailto:p.dazad@uniandes.edu.co)  
Requerimiento 2

**Análisis de complejidades**

**Requerimiento 1. Contar los avistamientos en una ciudad.**

La complejidad de este requerimiento es de **O(m log(m))+O(k)**, siendo **m** el número de avistamientos en la ciudad ingresada por parámetro, y **k** el número de ciudades diferentes que hay. Tiene dicha complejidad porque para resolverlo usamos un get (O(1)) para obtener la lista con los avistamientos de la ciudad consultada, y un merge sort (O(m log(m)) para ordenarlos según su fecha y hora. De este modo, queda la complejidad mayor. Y a esto se suma la operación para buscar la ciudad con mayor número de avistamientos, la cual tiene una complejidad O(k).

**Requerimiento 2. Contar los avistamientos por duración.**

La complejidad de este es **O(m\*k)+O(p log(p))**. Donde **m** es el número de listas que entran dentro del rango de segundos, **k** es el número de avistamientos dentro de cada lista, y **p** el número total de avistamientos dentro del rango de segundos. Tiene esta complejidad porque se obtiene una lista con las listas de avistamientos dentro del rango, y se recorre esta lista (O(m)). Dentro de cada lista se recorren todos sus elementos para agregarlos a una lista filtrada (O(k)), y finalmente se hace merge sort a la lista con los elementos filtrados (O(p log(p)).

**Requerimiento 3. Contar avistamientos por hora/minuto del día.**

Su complejidad es **O(m\*k log(k))**. En la que **m** es el número de listas de avistamientos que entran dentro del rango, y **k** el número de avistamientos que hay en cada lista. Debido a que después de hacer una lista con las listas que entran dentro del rango con la función values, se recorre cada una (O(m)) y en cada lista se realiza un merge sort (k log(k)).

**Requerimiento 4. Contar los avistamientos en un rango de fechas.**

Al igual que el requerimiento 2, la complejidad del requerimiento 4 es **O(m\*k)+O(p log(p))**. Siendo **m** el número de listas que entran en el rango de fechas, **k** el número de avistamientos en cada fecha, y **p** el número de avistamientos que están dentro del rango. En este también se recorre una lista de listas (O(m)) y en cada una se recorren todos sus elementos (O(k)) para añadirlos a una lista filtrada, y después se realiza un merge sort (O(p log(p)).

**Requerimiento 5. Contar los avistamientos de una zona geográfica.**

Su complejidad es de **O(m\*k\*t)**, en donde **m** es el número de mapas cuya longitud entra en el rango, **k** es el número de listas por latitud (que ya están filtradas dentro del rango de longitud) que entran dentro del rango de latitud, y **t** el número de avistamientos que hay en cada lista por latitud filtrada. En este requerimiento primero se toma una lista de mapas que entran dentro de las longitudes ingresadas, e ingresa a cada mapa dentro de esa lista (O(m)). Después, en cada mapa toma una lista de listas que ingresan dentro del rango de latitudes (O(k)) y en cada lista recorre todos los avistamientos para agregarlos a una lista con los elementos filtrados (O(t)).

**Requerimiento 6.**

Su complejidad es la misma que la del requerimiento 5, **O(m\*k\*t)**. Esto debido a que la lista filtrada por zona geográfica se obtiene con la misma función que en el requerimiento 5, y a partir de esa lista se crea el mapa. El proceso de creación del mapa es O(t), y al sumarlo con la complejidad de filtrar los avistamientos, se deja sólo el mayor (O(m\*k\*t)).