Nicolas Camargo (202020782)

Juan Manuel Pérez Sanchez (202021827)

Análisis de Complejidad Reto 3.

Requerimiento 1.

Complejidad	O(m log m)

Para este Requerimiento 1. la complejidad está en el Mergesort que se hace en la lista de los avistamientos que se encontraron para una ciudad en específico. Esta lista es una lista mucho más corta que la lista original y por eso recibe un m. Para la búsqueda de la ciudad especifica la complejidad de este algoritmo se da en O(1) al ser un .get() en un Ordered Map y un .get() en un Map.

Requerimiento 2. (Nicolas Camargo)

Complejidad	**O(x log x)**

Para el requerimiento 2, la complejidad es dada por el ciclo realizado para la búsqueda de valores en el árbol de duración, por tanto, primero se extraen los valores del árbol según la duración dada por el usuario, los cuales quedan en un hash table, de ahí se procede a recorrer dicha tabla hash para conseguir cada valor y guardarlo en un arreglo. En la línea 261 se obtiene un O(n) teniendo en cuenta que solo extrae los valores n que corresponden a la cantidad de datos entre las duraciones insertadas por el usuario. Posteriormente, el ciclo (x log x) donde x corresponde a cada set de valores de avistamientos.

Requerimiento 3. (Juan Manuel Pérez)

Complejidad	O(p log p)

Para este Requerimiento 3. al inicio hay una complejidad de búsqueda en el árbol para encontrar los rangos que tiene una complejidad de O(log n) al ser un RBT y estar balanceado. Luego se crea una lista con los avistamientos en el rango que tiene una complejidad de O(p) porque toca sacar avistamiento por avistamiento, al final toca organizar la lista basándose en la fecha y hora por lo que se hace un Mergesort con complejidad O(p log p). La complejidad mayor esta en el Mergesort.

Requerimiento 4.

Complejidad	O(v log v)

Para el Requerimiento 4. de igual forma al haber una búsqueda por rangos se obtiene una complejidad de O(log n) ya que es un RBT y estar balanceado. Después de crea una lista con los avistamientos que tiene una complejidad O(v) la recorrer avistamiento por avistamiento, la fecha ya esta ordenada

solo falta ordenar las horas por lo que se hace un Mergesort con una complejidad O(v log v). La mayor complejidad se da en en el Mergesort con complejidad O(v log v).

Requerimiento 5.

Complejidad	O(h log h)

Para el Requerimiento 5. También hay una búsqueda por rangos y como es en un RBT la complejidad de este algoritmo al estar balanceado es de O(log n). Se crea una lista y se recorre los avistamientos por avistamiento que tiene una complejidad de O(h). Esa Lista se tiene que ordenar por latitud entonces se hace un mergesort de O(h log h). Se sacan las posiciones de las latitudes con unos ciclos de complejidad O(h) y se hace un .sublist() con las posiciones por lo que se recorre un O(f). El algoritmo con mayor complejidad esta dado en el mergesort con una complejidad de O(h log h)