## **Documento de Análisis**

Estructuras de datos y algoritmos

Reto 3

Integrantes:

* Raul Santiago Rincon Corba – [r.rincon@uniandes.edu.co](mailto:r.rincon@uniandes.edu.co) - 202120414
* Benjamin Raisbeck Garcia - [b.raisbeck@uniandes.edu.co](mailto:b.raisbeck@uniandes.edu.co) – 202120398

Análisis de complejidad:

# **Requerimiento 1:**

|  |  |
| --- | --- |
| ARCHIVO | TIEMPO DE REQUERIMIENTO (ms) |
| Small | 0 |
| 5pct (%) | 46.8 |
| 10pct (%) | 62.5 |
| 20pct (%) | 78.1 |
| 30pct (%) | 62.5 |
| 50pct (%) | 78.1 |
| 80pct (%) | 83.7 |
| Large (todos los datos) | 78.1 |

La complejidad es de O(n+m), donde n es el recorrido de la lista de llaves (clubes) hasta encontrar la llave ingresada por el usuario. Al encontrar esa llave en la lista, se saca el valor de la llave en el mapa, el valor siendo la lista de jugadores en ese club y su información. m es el recorrido de la lista de la información de jugadores, donde para cada elemento (la info de jugador i), se coge el primer elemento (la id de jugador i), y se añade a la lista de ids que se retorna.

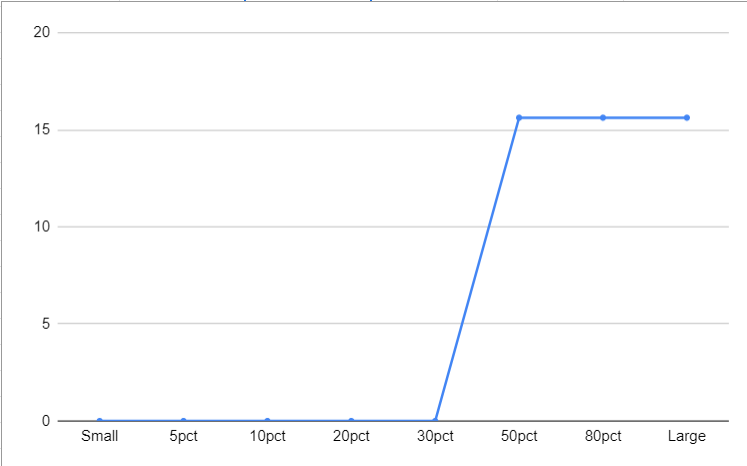
# **Requerimiento 2:**

|  |  |
| --- | --- |
| ARCHIVO | TIEMPO DE REQUERIMIENTO (ms) |
| Small | 0 |
| 5pct (%) | 0 |
| 10pct (%) | 0 |
| 20pct (%) | 0 |
| 30pct (%) | 0 |
| 50pct (%) | 0 |
| 80pct (%) | 0 |
| Large (todos los datos) | 0 |

La complejidad es de O(m), m siendo el número de jugadores dentro de todos los rangos dados, m también sería el recorrido for de cada elemento en la lista. Dentro de ese for se coge el elemento i en la lista (la info de un jugador), y desde ese elemento i se cogen elementos p y w (potencial y salario) y se comparan con los parámetros dado por el usuario. Si cumplen con las restricciones, el id de la info de ese jugador se añade a una sublista.

# **Requerimiento 3:**

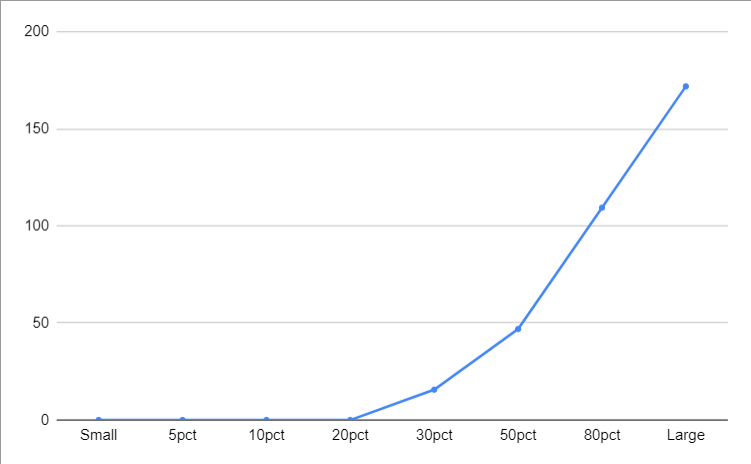
|  |  |
| --- | --- |
| ARCHIVO | TIEMPO DE REQUERIMIENTO (ms) |
| Small | 0 |
| 5pct (%) | 0 |
| 10pct (%) | 0 |
| 20pct (%) | 0 |
| 30pct (%) | 0 |
| 50pct (%) | 15,625 |
| 80pct (%) | 15,625 |
| Large (todos los datos) | 15,625 |



La complejidad es de O((n\*m) + c), n es el recorrido de la lista de tags hasta que se encuentre el tag ingresado por el usuario, donde se saca el valor de la llave que le corresponde al tag, el valor siendo un árbol de los salarios, y de ese árbol se crea una lista ordenada de los salarios. m (que ocurre en el recorrido de n) es el recorrido de la lista de salarios para encontrar los salarios dentro del rango dado por el usuario, donde se saca el valor de los salarios apropiados, el valor siendo el id los jugadores, y se ponen en una lista. c es el recorrido de la lista de ids de jugadores, en el cual se usa la función translator para convertir cada id de la lista en la información de los jugadores, y esa información se añade a otra lista que es ordenada usando merge sort y finalmente se retorna.

# **Requerimiento 4:**

|  |  |
| --- | --- |
| ARCHIVO | TIEMPO DE REQUERIMIENTO (ms) |
| Small | 0 |
| 5pct (%) | 0 |
| 10pct (%) | 0 |
| 20pct (%) | 0 |
| 30pct (%) | 15,625 |
| 50pct (%) | 46,875 |
| 80pct (%) | 109,375 |
| Large (todos los datos) | 171,875 |



La complejidad es de O(m). Cogiendo el valor de la llave (trait) que es igual al trait dado por el usuario, se consigue un árbol donde las llaves son las fechas de nacimiento de los jugadores con ese trait, y los valores son los ids de esos jugadores. Usando om.values se consigue una lista de todos los ids de los jugadores que nacieron dentro del rango dado por el usuario. m es el recorrido de la lista para añadir cada id con getElement a una lista filtrada.

# **Requerimiento 5:**

|  |  |
| --- | --- |
| ARCHIVO | TIEMPO DE REQUERIMIENTO (ms) |
| Small | 0 |
| 5pct (%) | 0 |
| 10pct (%) | 0 |
| 20pct (%) | 0 |
| 30pct (%) | 0 |
| 50pct (%) | 0 |
| 80pct (%) | 0 |
| Large (todos los datos) | 0 |

Texto

Descripción generada automáticamente

La complejidad es de O(). donde n es la duración del recorrido que sigue siempre que i sea menor que N (número de segmentos en que se divide el rango). Dentro del while hay un for, el cual se ejecuta en el size de los jugadores dentro del rango, pero como el numero de segmentos dados en el while es tan pequeño se realiza muy rápidamente.