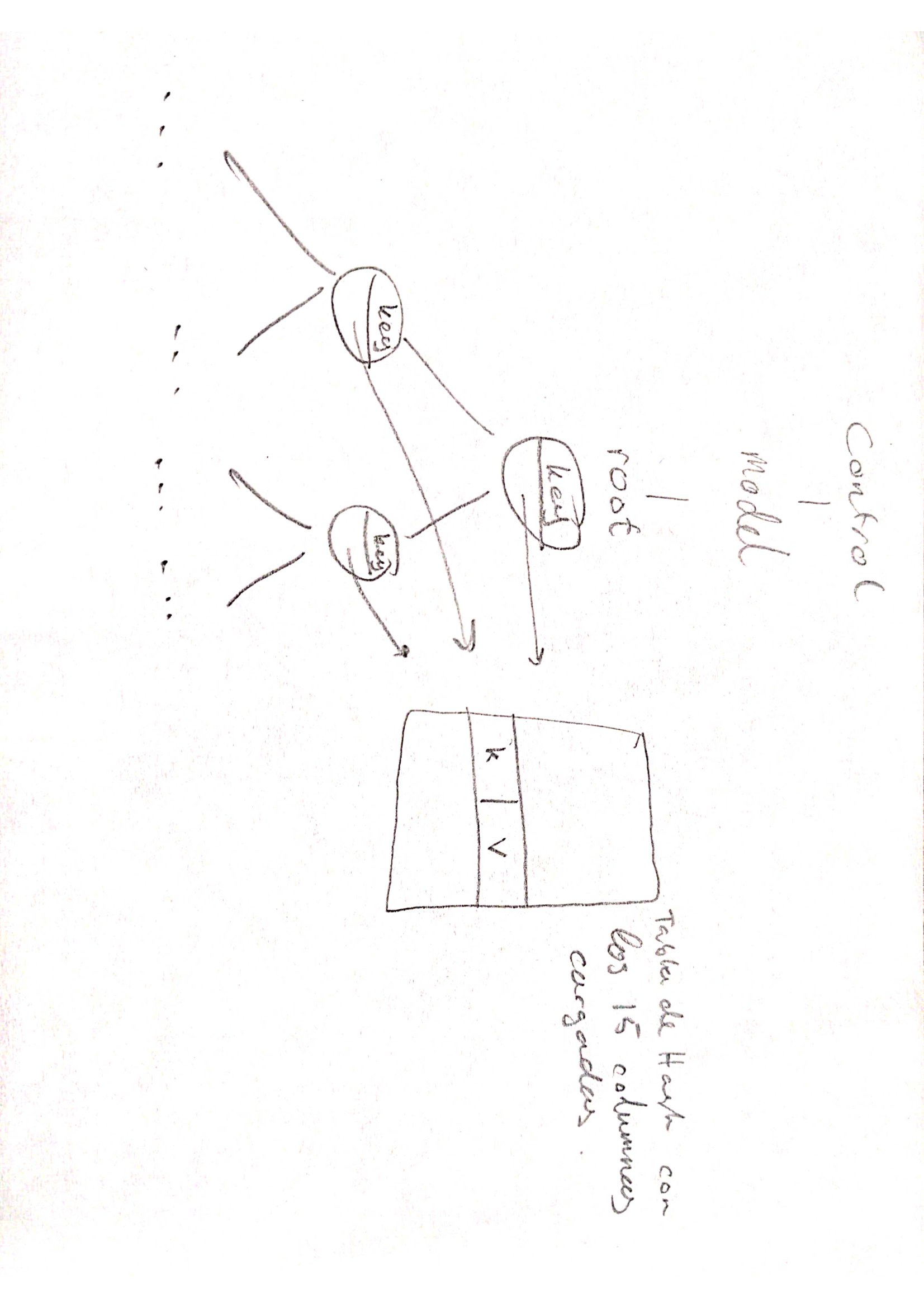
**ANÁLISIS DEL RETO 3**

*Simón Calderón López, 202113559,* [*s.calderon@uniandes.edu.co*](mailto:s.calderon@uniandes.edu.co)

*Alejandro Torres Rodríguez, 202013743,* [*a.torres48@uniandes.edu.co*](mailto:a.torres48@uniandes.edu.co)

*Johannes Almas, 202224983,* [*j.almas@uniandes.edu.co*](mailto:j.almas@uniandes.edu.co)

# **Diagrama de Estructura**



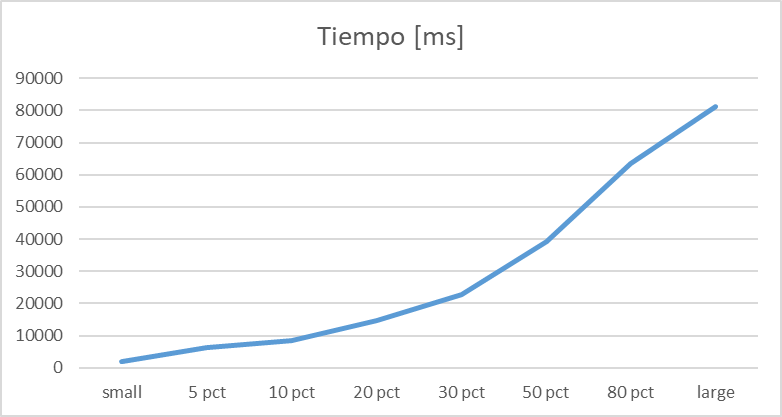
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

# **Tabla de datos Carga**

|  |  |
| --- | --- |
| CARGA | Tiempo [ms] |
| small | 1882.121 |
| 5 pct | 6317.7295 |
| 10 pct | 8588.9388 |
| 20 pct | 14571.2139 |
| 30 pct | 22652.8309 |
| 50 pct | 39253.0903 |
| 80 pct | 63562.7743 |
| large | 81297.9836 |

**Graficas**



# **Requerimiento <<1>>**

Reportar todos los accidentes dado un rango de fechas.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Fecha inicial del intervalo y fecha final del intervalo. |
| **Salidas** | Accidentes registrados en el rango de fechas. |
| **Implementado (Sí/No)** | Sí/ El grupo. |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Merge sort | O( n\*log(n) ) |
| Loop para hacer la tabla | O(k\* n ) |
| ***TOTAL*** | ***O( n\*log(n) )*** |

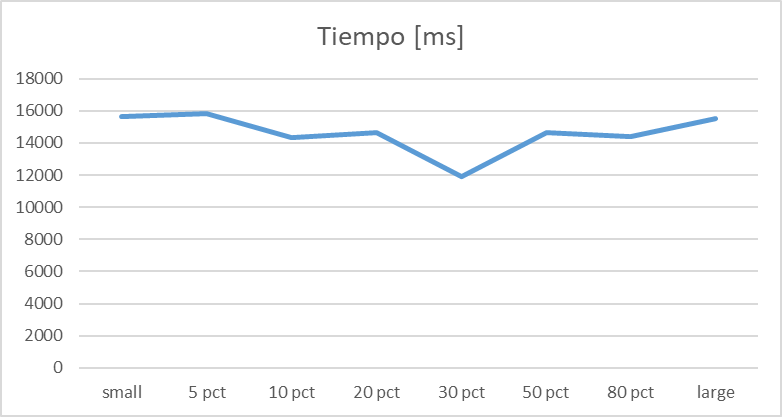
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ1 | Tiempo [ms] |
| small | 15612.6616 |
| 5 pct | 15843.2138 |
| 10 pct | 14340.0298 |
| 20 pct | 14642.3515 |
| 30 pct | 11886.3526 |
| 50 pct | 14661.2321 |
| 80 pct | 14423.6712 |
| large | 15488.5043 |

**Gráficas Análisis**



**Análisis**

El requerimiento 1 solo hace un merge sort para tratar los datos. Este tiene una complejidad temporal de O (n\*log(n)). Luego hace una tabla para y la hace print, pero nada tiene una complejidad mayor a O(k\*n) así que la complejidad total es de O(n\*log(n)).

# **Requerimiento <<2>>**

Reportar todos los accidentes en un intervalo de horas del día para un mes y año dados.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Hora y minutos iniciales y finales del intervalo, año de consulta y mes de consulta |
| **Salidas** | El número total de accidentes ocurridos en el intervalo de tiempo, año y mes dados. |
| **Implementado (Sí/No)** | Sí/ El grupo. |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Loop para agregar elementos a una lista | O(n) |
| Merge sort | O(n\*log(n)) |
| Loop para hacer la tabla | O(k\*n) |
| ***TOTAL*** | ***O(n\*log(n))*** |

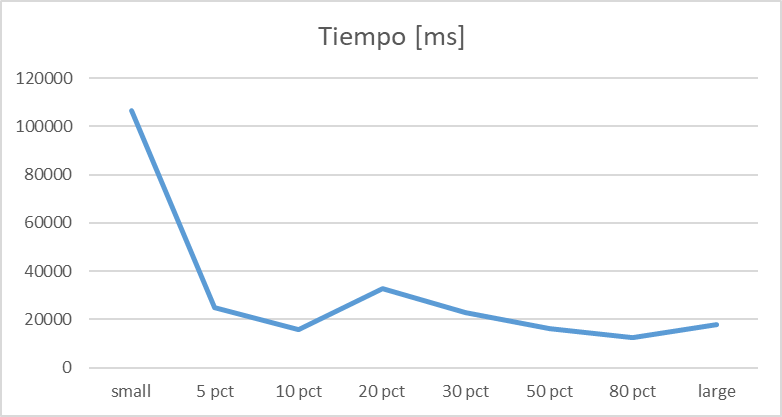
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ2 | Tiempo [ms] |
| small | 106493.2972 |
| 5 pct | 24812.0533 |
| 10 pct | 15721.9074 |
| 20 pct | 32878.6051 |
| 30 pct | 22673.5258 |
| 50 pct | 16311.8742 |
| 80 pct | 12631.7144 |
| large | 17907.099 |

**Gráficas**



**Análisis**

El requerimiento 2 solo agrega ciertos elementos a una lista antes de ordenarlos con un merge sort. Estos pasos tienen una complejidad temporal de O(n) y O (n\*log(n)) respectivamente. Luego hace una tabla para y la hace print, pero nada tiene una complejidad mayor a O(k\*n) así que la complejidad total es de O(n\*log(n)).

# **Requerimiento <<3>>**

Reportar todos los accidentes dado un rango de fechas.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** |  |
| **Salidas** |  |
| **Implementado (Sí/No)** |  |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Paso 1 | O(...) |
| Paso 2 | O(...) |
| Paso …. | O(...) |
| ***TOTAL*** | ***O(...)*** |

## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ1 | Tiempo [ms] |
| small |  |
| 5 pct |  |
| 10 pct |  |
| 20 pct |  |
| 30 pct |  |
| 50 pct |  |
| 80 pct |  |
| large |  |

**Gráficas Análisis**

**Análisis**

# **Requerimiento <<4>>**

Reportar los 5 accidentes más recientes dada una gravedad y un rango de fechas.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Fecha inicial del intervalo, fecha final del intervalo y gravedad del accidente. |
| **Salidas** | Accidentes registrados en el rango de fechas. |
| **Implementado (Sí/No)** | Sí/ Alejandro Torres Rodríguez |

## **Análisis de complejidad**

Análisis de complejidad de cada unos de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Loop para agregar elementos a una lista | O(n) |
| Merge sort | O(n\*log(n)) |
| Loop para hacer la tabla | O(k\*n) |
| ***TOTAL*** | ***O(n\*log(n))*** |

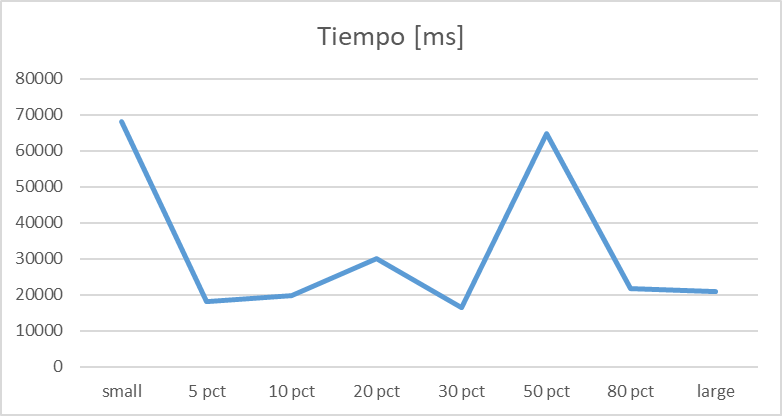
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ4 | Tiempo [ms] |
| small | 68006.1664 |
| 5 pct | 18026.6497 |
| 10 pct | 19805.238 |
| 20 pct | 30216.8302 |
| 30 pct | 16496.3855 |
| 50 pct | 64815.4276 |
| 80 pct | 21737.0134 |
| large | 20882.3064 |

**Gráficas Análisis**



**Análisis**

El requerimiento 4 agrega ciertos elementos a una lista antes de ordenarlos con un merge sort. Estos pasos tienen una complejidad temporal de O(n) y O (n\*log(n)) respectivamente. Luego hace una tabla para y la hace print, pero nada tiene una complejidad mayor a O(k\*n) así que la complejidad total es de O(n\*log(n)).

# **Requerimiento <<5>>**

Reportar los 10 accidentes menos recientes ocurridos en un mes y año para una localidad de la ciudad.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Localidad de Bogotá en la que ocurrieron accidentes, un mes y un año entre 2015 y 2022. |
| **Salidas** | El número total de accidentes ocurridos en la localidad durante la fecha indicada y los 10 registros de accidentes más recientes en la localidad. |
| **Implementado (Sí/No)** | Sí/ Johannes Almas. |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Loop para agregar elementos a una lista | O(n) |
| Merge sort | O(n\*log(n)) |
| Loop para hacer la tabla | O(k) |
| ***TOTAL*** | ***O(n\*log(n))*** |

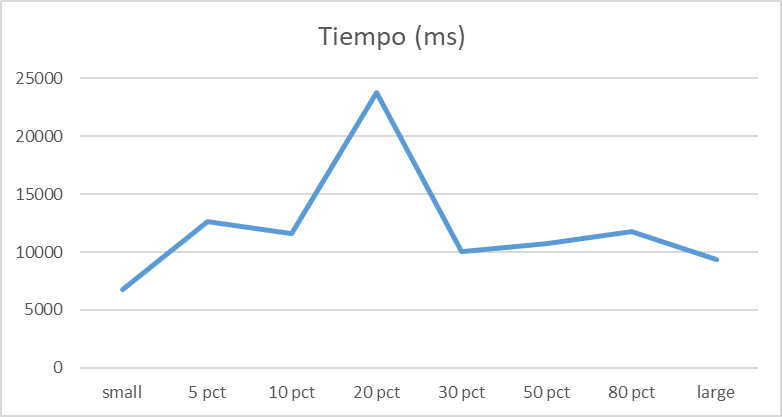
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ5 | Tiempo [ms] |
| small | 6754.2222 |
| 5 pct | 12653.2719 |
| 10 pct | 11613.6233 |
| 20 pct | 23771.1257 |
| 30 pct | 10067.4184 |
| 50 pct | 10698.374 |
| 80 pct | 11773.0676 |
| large | 9320.6129 |

**Gráficas**



**Análisis**

El requerimiento 5 agrega ciertos elementos a una lista antes de ordenarlos con un merge sort como tratamiento de los datos. Estos pasos tienen una complejidad temporal de O(n) y O (n\*log(n)) respectivamente. Luego hace una tabla para y la hace print, pero nada tiene una complejidad mayor a O(k) (como la tabla tiene un tamaño máximo de 10) así que la complejidad total es de O(n\*log(n)).

# **Requerimiento <<6>>**

Mostrar los N accidentes ocurridos dentro de una zona específica para un mes y un año.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Mes, año entre 2015 y 2022, coordenada del centro del área (Latitud y Longitud), radio del área en Km y número de accidentes. |
| **Salidas** | Los N accidentes ocurridos, organizados por su cercanía al centro de la zona y los registros de todos los N accidentes ocurridos en la zona. |
| **Implementado (Sí/No)** | Sí/ El grupo. |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Loop para agregar elementos a una lista | O(n) |
| Merge sort | O(n\*log(n)) |
| Loop para hacer la tabla | O(k\*n) |
| ***TOTAL*** | ***O(n\*log(n))*** |

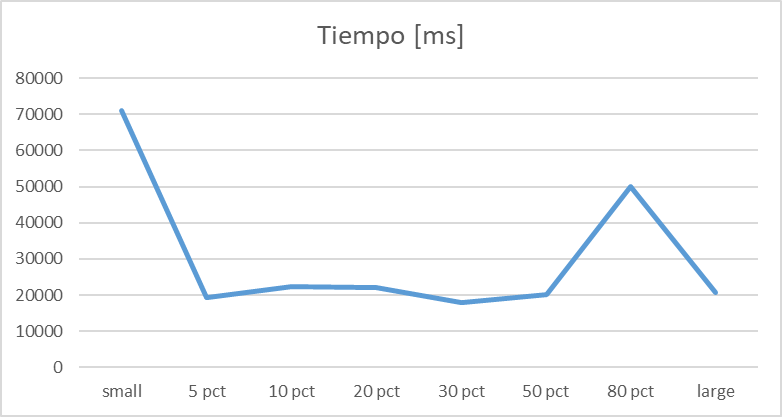
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ6 | Tiempo [ms] |
| small | 71115.5659 |
| 5 pct | 19392.3336 |
| 10 pct | 22322.5743 |
| 20 pct | 22164.1121 |
| 30 pct | 17847.3204 |
| 50 pct | 20185.048 |
| 80 pct | 50010.6298 |
| large | 20578.9271 |

**Gráficas**



**Análisis**

El requerimiento 6 agrega ciertos elementos a una lista antes de ordenarlos con un merge sort. Estos pasos tienen una complejidad temporal de O(n) y O (n\*log(n)) respectivamente. Luego hace una tabla para y la hace print, pero nada tiene una complejidad mayor a O(k\*n) así que la complejidad total es de O(n\*log(n)).

# **Requerimiento <<7>>**

Reportar los accidentes más temprano y más tarde para cada día de un mes y año dado, y graficar el histograma de frecuencias de accidentes por hora para ese mismo mes y año.

## **Descripción**

|  |  |
| --- | --- |
| **Entrada** | Mes y año entre 2015 y 2022. |
| **Salidas** | Para cada día del mes en el año dados, dar la información del accidente ocurrido más temprano y más tardío y la gráfica de barras con una representación (simplificada) de los accidentes ocurridos en cada franja horaria durante el mes y año dados, entendiendo una franja horaria como el tiempo entre una hora y justo antes de la hora anteriorp |
| **Implementado (Sí/No)** | Sí/ El grupo. |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Loop para agregar elementos a una lista | O(n) |
| Merge sort | O(n\*log(n)) |
| Loop para agregar listas a un mapa | O(k\*n) |
| Loop para encontrar datos para el gráfico | O(k\*n) |
| ***TOTAL*** | ***O(n\*log(n))*** |

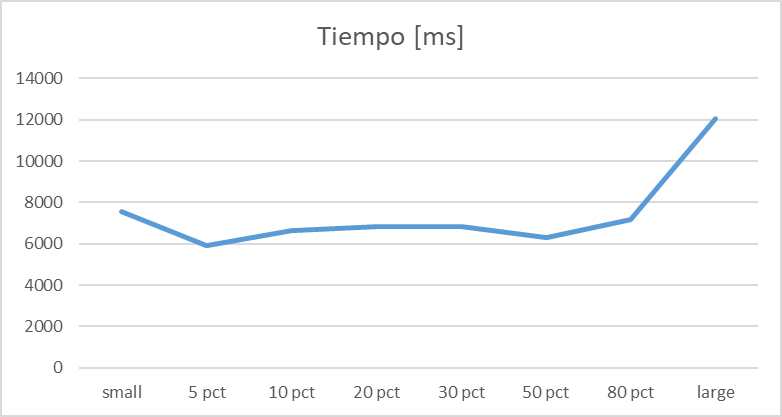
## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-9300H @2.40Ghz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 Home |

**Tabla de datos**

|  |  |
| --- | --- |
| REQ7 | Tiempo [ms] |
| small | 7544.7106 |
| 5 pct | 5904.1313 |
| 10 pct | 6613.4961 |
| 20 pct | 6815.4666 |
| 30 pct | 6850.0056 |
| 50 pct | 6283.8005 |
| 80 pct | 7186.7849 |
| large | 12064.328 |

**Gráficas**



**Análisis**

El requerimiento 7 agrega ciertos elementos a una lista antes de ordenarlos con un merge sort como tratamiento de los datos. Estos pasos tienen una complejidad temporal de O(n) y O (n\*log(n)) respectivamente. También hace un doble for loop dos veces. Los dos son loops de los n elementos de la lista k veces, entonces los dos tienen una complejidad de O(k\*n). La complejidad total es de O(n\*log(n)).