# **Requerimientos funcionales:**

|  |  |
| --- | --- |
| Nombre: | **R1 – Cargar el archivo de datos GeoJSON** |
| Descripción: | Los datos que se encuentran en el archivo de comparendos son leídos uno por uno y guardados en estructuras de datos, en este caso tablas de hash Linear Probing y Separate Chaining, Cola de prioridad y árbol balanceado. **Los archivos solo deben poder cargarse una vez.** |
| Datos de Entrada: | se debe realizar un input en la consola, el input UNICAMENTE DEBE SER EL NUMERO 1 para que se realice la carga. |
| Datos de Salida: | Se muestra el total de comparendos leídos y la información del comparendo con mayor ObjectID. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | En este caso se utilizará un Queue para guardar los datos, este Queue se iterará en los requerimientos funcionales las diferentes estructuras que se requieran |
| Complejidad Temporal y justificación: | O(n) – linear. Agregar un elemento a una Queue tiene una complejidad constante, pero como se agregan n elementos la complejidad tiene a n. |

|  |  |
| --- | --- |
| Nombre: | **R2 – Mostrar Los datos de un comparendo** |
| Descripción: | A partir de un comparendo dado, se muestran los datos que componen a sus atributos |
| Datos de Entrada: | Comparendo del cual se desea obtener los datos |
| Datos de Salida: | Se muestran los OBJECTID, FECHA\_HORA, INFRACCION, CLASE\_VEHICULO, TIPO\_SERVICIO y LOCALIDAD del comparendo |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Ninguna |
| Complejidad Temporal y justificación: | O(1) – constant. Solo se la información del comparendo |

|  |  |
| --- | --- |
| Nombre: | **1A- Obtener los M comparendos con mayor gravedad.** |
| Descripción: | Se muestran los M comparendos con mayor gravedad según la infracción |
| Datos de Entrada: | Numero M de comparendos que se desea ver. |
| Datos de Salida: | Se debe mostrar en consola los M comparendos de mayor a menor gravedad. De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora y clase de vehículo. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de tipo de servicio y si estos son iguales se utilizará como criterio de prioridad el codigo de infracción. |
| Complejidad Temporal y justificación: | Se tiene que hacer un recorrido por la cola de prioridad con los M primeros comparendos, encontrar estos comparendos tiene una complejidad constante puesto que se encuentran en las primeras M posiciones de un arreglo que sirve de cola. O(1) – constant |

|  |  |
| --- | --- |
| Nombre: | **2A- Buscar los comparendos por mes y día de la semana.** |
| Descripción: | Muestra los comparendos que cumplan con los datos de mes y día de la semana ingresados |
| Datos de Entrada: | número del mes (1-12) y el día de la semana (L, M, I, J, V, S, D). |
| Datos de Salida: | Se debe mostrar en consola los N comparendos que cumplan los criterios de búsqueda. De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora y clase de vehículo. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Tabla de hash LinearProbing<String, Comparendo> tendrá como llave (número del mes/ día de la semana) |
| Complejidad Temporal y justificación: | Constante, asumiendo que hay un hashing uniforme, la complejidad seria de (3-5) siendo estos los números aproximados de llaves que se pueden encontrar una posición del SeparateChaining. O(1) – constant. |

|  |  |
| --- | --- |
| Nombre: | **3A- Buscar los comparendos que tienen una fecha-hora en un rango y que son de una localidad dada.** |
| Descripción: | Muestra los comparendos que cumplan con los datos de fecha-hora y localidad ingresados |
| Datos de Entrada: | El usuario ingresa un rango de fecha-hora [limite\_bajo, limite\_alto] en formato “YYYY/MM/DD-HH:MM:ss” y una localidad. |
| Datos de Salida: | Se debe mostrar en consola los N comparendos que cumplan los criterios de búsqueda. De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora y clase de vehículo. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | En este caso se utilizará un RedBlackBST<Date, Comparendo>  La llave es la fecha del comparendo |
| Complejidad Temporal y justificación: | O(2Log(N) se compara con los hijos de menor a mayor y se hace un descarte de dos en dos básicamente. Como este proceso se hace dos veces entre el limite inferior y el limite superior entonces la complejidad resulta en 2Log(N) |

|  |  |
| --- | --- |
| Nombre: | **1B- Buscar los M comparendos más cercanos a la estación de policía.** |
| Descripción: | Muestra los M comparendos más cercanos a la estación de policía del Campin |
| Datos de Entrada: | Numero M de comparendos que se desea ver. |
| Datos de Salida: | De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora, clase de vehículo, longitud y latitud geográficas. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | la estructura que mejor sirve para esta implementación es Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de distancia de menor a mayor. |
| Complejidad Temporal y justificación: | Constante, dado que siempre los primeros M comparendos los encontraré en la misma cantidad de tiempo. O(1) – constant |

|  |  |
| --- | --- |
| Nombre: | **2B- Buscar los comparendos por medio de detección, clase de vehículo, tipo de servicio y localidad.** |
| Descripción: | Muestra los N compararendos que cumplan los criterios de búsqueda ordenados por fecha |
| Datos de Entrada: | El usuario ingresa el medio de detección, clase de vehículo, tipo de servicio y localidad. |
| Datos de Salida: | De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora, clase de vehículo, longitud y latitud geográficas. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Tabla de hash SeparateChaining<String, Comparendo> tendrá como llave (medio de detección / clase de vehículo/ tipo de servicio/ localidad)  Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de fecha de menor a mayor. |
| Complejidad Temporal y justificación: | Constante, dado que siempre los primeros N comparendos de la hash table los encontraré en la misma cantidad de tiempo. O(1) – constant  Estos comparendos encontrados los ordenare con criterio de comparación de fecha con una MaxHeapCP y el mostrarlos todos tendrá una complejidad lineal O(n) – linear. |

|  |  |
| --- | --- |
| Nombre: | **3B- Buscar los comparendos que tienen una latitud en un rango dado y que involucraron un tipo de vehículo particular.** |
| Descripción: | Muestra los N compararendos que cumplan los criterios de búsqueda |
| Datos de Entrada: | El usuario ingresa un rango de latitudes [limite\_bajo, limite\_alto] y una clase de vehículo (automovil, camioneta, moto, etc.). |
| Datos de Salida: | De cada comparendo se debe mostrar el id, tipo de servicio, infracción, fecha-hora, clase de vehículo y latitud geográfica. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de latitud de menor a mayor. |
| Complejidad Temporal y justificación: | Puede que necesite recorrer toda la cola para encontrar los resultados, entonces la complejidad será O(n) – linear. |

|  |  |
| --- | --- |
| Nombre: | **1C- Visualizar Datos en una Tabla ASCII** |
| Descripción: | Muestra los N compararendos que cumplan los criterios de búsqueda |
| Datos de Entrada: | número de días D |
| Datos de Salida: | Se muestra una tabla con el número de comparendos en el rango de días D. El valor de cada “\*” representa un numero especifico de comparendos y es lo que se muestra en la consola |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de fecha de menor a mayor. |
| Complejidad Temporal y justificación: | Puede que necesite recorrer toda la cola para encontrar los resultados, entonces la complejidad será O(n) – linear. |

|  |  |
| --- | --- |
| Nombre: | **2C- El costo de los tiempos de espera hoy en día (cola)** |
| Descripción: | Muestra los N compararendos que cumplan los criterios de búsqueda |
| Datos de Entrada: | ninguno |
| Datos de Salida: | •El costo total que generan las penalizaciones en 2018  •El número de días en promedio que debe esperar un comparendo  • Un histograma ASCII con el número comparendos procesados por día y el número de comparendos que están esperando. Recuerde que por día se pueden procesar manualmente máximo 1500 comparendos. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Cola Queue<Comparendo> |
| Complejidad Temporal y justificación: | Puede que necesite recorrer toda la cola para encontrar los resultados, entonces la complejidad será O(n) – linear. |

|  |  |
| --- | --- |
| Nombre: | **3C- El costo de los tiempos de espera usando el nuevo sistema** |
| Descripción: | Muestra los N compararendos que cumplan los criterios de búsqueda |
| Datos de Entrada: | ninguno |
| Datos de Salida: | •El costo total que generan las penalizaciones en 2018  •El número de días en promedio que debe esperar un comparendo  • Un histograma ASCII con el número comparendos procesados por día y el número de comparendos que están esperando. Recuerde que por día se pueden procesar manualmente máximo 1500 comparendos. |
| Parametrización de Estructura(s) Genérica(s) de Datos a utilizar: | Cola de prioridad MaxHeapCP<Comparendo> con criterio de prioridad de fecha de menor a mayor. |
| Complejidad Temporal y justificación: | Puede que necesite recorrer toda la cola para encontrar los resultados, entonces la complejidad será O(n) – linear. |