**Diseño de proyecto**

**Carga de datos**

* Implementación: Utilizar lectura de archivos Json para cada comparendo y almacenar la información recibida en objetos Infraccion (objeto referente a un comparente) que se irán añadiendo a una lista encadenada. A medida que se van creando los comparendos se van comparando para encontrar aquel con el mayor OBJECT\_ID mediante una variable. A la vez se va revisando los valores máximos y mínimos de longitud y latitud.
* Parámetros de entrada: ruta del archivo Json a leer
* Retorno: Comparendo can mayor OBJECT\_ID y arreglo de Minimax
* Orden del algoritmo: N

**Parte A**

1A.

* Implementación: Recorrer la lista de los comparendos hasta encontrar el primero con una localidad recibida por parámetro.
* Parámetros de entrada: String de la localidad a buscar
* Retorno: Primer comparendo con dicha localidad
* Orden del algoritmo: N

2A.

* Implementación: Recorrer toda la lista revisando aquellos comparendos con la FECHA\_HORA recibida por parámetro y añadir estos a una nueva lista encadenada para retornar. La adición a la nueva lista se hace en orden de código infracción de menor a meyor.
* Parámetros de entrada: String de la FECHA\_HORA a buscar
* Retorno: Lista Encadenada con los comparendos encontrados en orden ascendente de código de infracción
* Orden del algoritmo: N²

3A.

* Implementación: Se recorre toda la lista buscando los comparendos en alguna de las fechas recibidas por parámetro. Estos se van añadiendo en orden ascendente de código de infracción en una nueva lista encadenada. Luego se recorre esta lita nueva con dos contadores que calculan la cantidad de comparendos en cada fecha y se adiciona un String a una cola de Strings para retornar, con el código de infracción y el número de comparendos para cada fecha separados por una coma.
* Parámetros de entrada: las dos FECHA\_HORAs que se desea buscar
* Retorno: Cola de Strings con la información calculada
* Orden del algoritmo: N³

**Parte B**

1B.

* Implementación: Recorrer la lista de comparendos hasta encontrar el primero con una infracción encontrada por parámetro.
* Parámetros de entrada: el String con el código de infracción a buscar
* Retorno: Comparendo encontrado
* Orden del algoritmo: N

2B.

* Implementación: Recorrer toda la lista buscando los comparendos con un código de infracción recibido por parámetro y añadir estos a una nueva lista encadenada para retornar. La adición se hace de manera ordenada por FECHA\_HORA de manera ascendente
* Parámetros de entrada: String del código de infracción a buscar
* Retorno: Lista encadenada con los comparendos encontrados
* Orden del algoritmo: N²

3B.

* Implementación: Ordenar la lista en orden ascendente por código de infracción. Posteriormente usar dos contadores para contador el número de comparendos en cada tipo de vehículo por cada infracción y agregar un String a una cola de Strings con el código de infracción y el número de comparendo por cada tipo separados por una coma. En la adición se verifica si alguno de los contadores es 0, en cuyo caso no se agrega la información.
* Parámetros de entrada: Ninguno
* Retorno: Cola de Strings con la información calculada
* Orden del algoritmo: N³

**Parte C**

1C.

* Implementación: Se recorre toda la lista buscando aquellos comparendos en la localidad recibida por parámetro y que estén entre las dos fechas recibidas por parámetro. Estos se adicionan en orden alfabético de código de infracción en una nueva lista encadenada. Luego se recorre la nueva lista contando la cantidad de comparendos por cada código de infracción y se añade un String a una cola de Strings con el código de infracción y el número de comparendos separados por una coma. Se retorna esta última cola.
* Parámetros de entrada: String de la localidad a buscar y fechas límite de la búsqueda
* Retorno: Cola de Strings con la información calculada
* Orden del algoritmo: N³

2C.

* Implementación: Se recorre toda la lista de comparendos buscando aquellos entre las dos fechas límite recibidas por parámetro. Estos se agregan en orden ascendente de de código de infracción en una nueva lista encadenada. Posteriormente se recorre esta nueva lista contando el número de comparendos para cada infracción y se añade un String a otra lista encadenada de Strings con el código y el número de comparendos separados por una coma. Está adición se hace en orden descendente según el número de comparendos. Por último se añaden los primeros N elementos de esta última lista de Strings a un acola de Strings para retornar.
* Parámetros de entrada: Número N de comparendos a registrar y fechas límite dela búsqueda
* Retorno: Cola de String con la información calculada
* Orden del algoritmo: N⁴

3C.

* Implementación: Se ordena la lista de comparendos en orden alfabético por localidad. Posteriormente se recorre la lista ordenada contando el número de comparendos y se añade un String a una cola de Strings con la localidad y el número de comparendos. Por ultimo durante la impresión en consola se imprime el nombre de la localidad y, la parte entera del número de comparendos dividido 50, veces un asterisco.
* Parámetros de entrada: Ninguno
* Retorno: Cola con la información registrada
* Orden del algoritmo: N⁴