Laura Maya

201423854

Carlos Peñaloza

201531973

Al menos tres requerimientos funcionales para cada parte:

Parte A

Nombre	R1: Ward con más alarmas Slight en un rango de años
Resumen	Buscar el barrio que tenga más alarmas Slight en determinado rango de años
Entradas	
El rango de años en el cual se quiere hacer la comparación	
Resultados	
Un reporte con los wards con más alarmas Slight en el rando de años que entró por parámetro	

Nombre	R2: Historial de colisiones Slight para un barrio
Resumen	Devuelve las colisiones Slight de un barrio
Entradas	
Barrio del que se quiere obtener el historial	
Resultados	
Reporte con las colisiones Slight de un barrio	

Nombre	R3: Promedio de colisiones Slight para un barrio
Resumen	Devuelve el promedio de colisiones Slight de un barrio
Entradas	
Barrio del que se quiere obtener el promedio	
Resultados	
Devuelve un double con el promedio de colisiones que tiene el barrio que entra por parámetro	

Laura Maya

201423854

Carlos Peñaloza

201531973

Nombre	R4: Obtener el último reporte de colisiones Slight de un barrio
Resumen	Poder acceder al último reporte de colisiones Slight que entró al sistema
Entradas	
Resultados	
Reporte con el último dato de las colisiones Slight	

Nombre	R5: Reportar al local authority las colisiones por encima del máximo permitido
Resumen	Reporta a un local authority las colisiones que están por encima del máximo permitido. El reporte incluye el nombre del barrio, el año en el que sucedieron y el número de colisiones.
Entradas	
Resultados	
Un reporte con las colisiones que superan el máximo permitido	

Nombre	R6: Almacenar los datos del archivo json
Resumen	Almacena los datos del archivo json en una lista
Entradas	
Resultados	
Una lista con los reportes que contiene el archivo json	

Nombre	R7: Ordenar las listas
Resumen	Debe ordenar las listas por año

Laura Maya

201423854

Carlos Peñaloza

201531973

Entradas
Resultados
La lista de colisiones queda ordenada por año de manera ascendente

Parte B

Nombre	R1: Almacenar información de las áreas
Resumen	Almacena los datos del archivo json en una lista
Entradas	
Archivo Boroughs.json	
Resultados	
Lista con la información de las áreas	

Nombre	R2: Organizar lista
Resumen	Se organiza la lista por año en el que ocurrió la colisión
Entradas	
Resultados	
La lista se encuentra ordenada por año	

Nombre	R3: Crear lista colisiones serias
Resumen	Se tomará la información de la lista de colisiones por área para crear una nueva lista de colisiones serias organizada por año
Entradas	
Resultados	

Laura Maya 201423854

Carlos Peñaloza 201531973

Nueva lista con las colisiones serias

Nombre	R4: Reporte colisiones serias
Resumen	Se realizará un reporte que incluya las alertas por número de colisiones serias que superan el número máximo permitido
Entradas	
Resultados	
Nuevo reporte (Serious collisions by cities) que incluye ciudades y/o áreas que superan el máximo recomendado, número de colisiones Serious presentadas y año en el que ocurrió.	

Nombre	R5: Respuesta solicitudes
Resumen	Se responderá a cada alcalde (en el orden en el que solicitaron): el área con el índice de colisiones Serious más alto en un rango de fechas. Historial de colisiones Serious para un área. Promedio de colisiones Serious para una área. El valor más reciente de colisiones Serious de una área.
Entradas	
Archivo Solicitudes_alcaldes.csv	
Resultados	
Respuesta a cada alcalde	

Parte C

Nombre	R1: Almacenar información entrada al parqueadero
Resumen	Se guarda la información del archivo en una cola
Entradas	
Archivo Entrada_al_parqueadero.json	

Laura Maya

201423854

Carlos Peñaloza

201531973

Resultados
Se guardó la información de la entrada al parqueadero en una cola

Nombre	R2: Almacenar información salida del parqueadero
Resumen	Se guarda la información del archivo en una cola
Entradas	
Archivo Salida_del_parqueadero.json	
Resultados	
Se guardó la información de la salida del parqueadero en una cola	

Nombre	R3: Parquear carros en cola
Resumen	Debe poder parquear un carro en uno de los 5 parqueaderos disponibles. Se debe optimizar para que el carro pueda salir de la forma más óptima posible.
Entradas	
Resultados	
Los carros de la cola quedan parqueados	

Laura Maya 201423854

Carlos Peñaloza 201531973

Estimación de Complejidad temporal de requerimientos funcionales a partir del uso de Estructuras de datos:

El primer requerimiento funcional para las primeras dos partes consiste en almacenar la información de los archivos .json en listas sencillamente encadenadas, lo cual tiene una complejidad de O(n) al tener que hacer un recorrido de tamaño n para agregar cada elemento a la lista. Posteriormente, es necesario organizar las listas según los criterios requeridos para llevar a cabo las búsquedas de información solicitadas, para esto se hará uso de merge sort, que para un arreglo tiene una complejidad temporal de O(n log n), dado que se hará uso de una lista sencillamente encadenada se demorara un poco más al gastar un tiempo extra calculando el tamaño de la estructura, pero en la implementación realizada se actualiza el atributo tamaño cada vez que se agrega o remueve un objeto, por lo cual el peor caso seguirá siendo de O(n log n).

Por otra parte, en el punto C se hará uso de colas y pilas, cuyos métodos básicos (pop, push, peek, isempty, enqueue y dequeue) todos tienen una complejidad temporal constante, es decir O(1) por lo que permiten resolver el problema de una manera eficiente en tiempo, lo que hace falta es encontrar la manera óptima para manipular los carros en las distintas pilas.

Explicación/Justificación de las estructuras de datos:

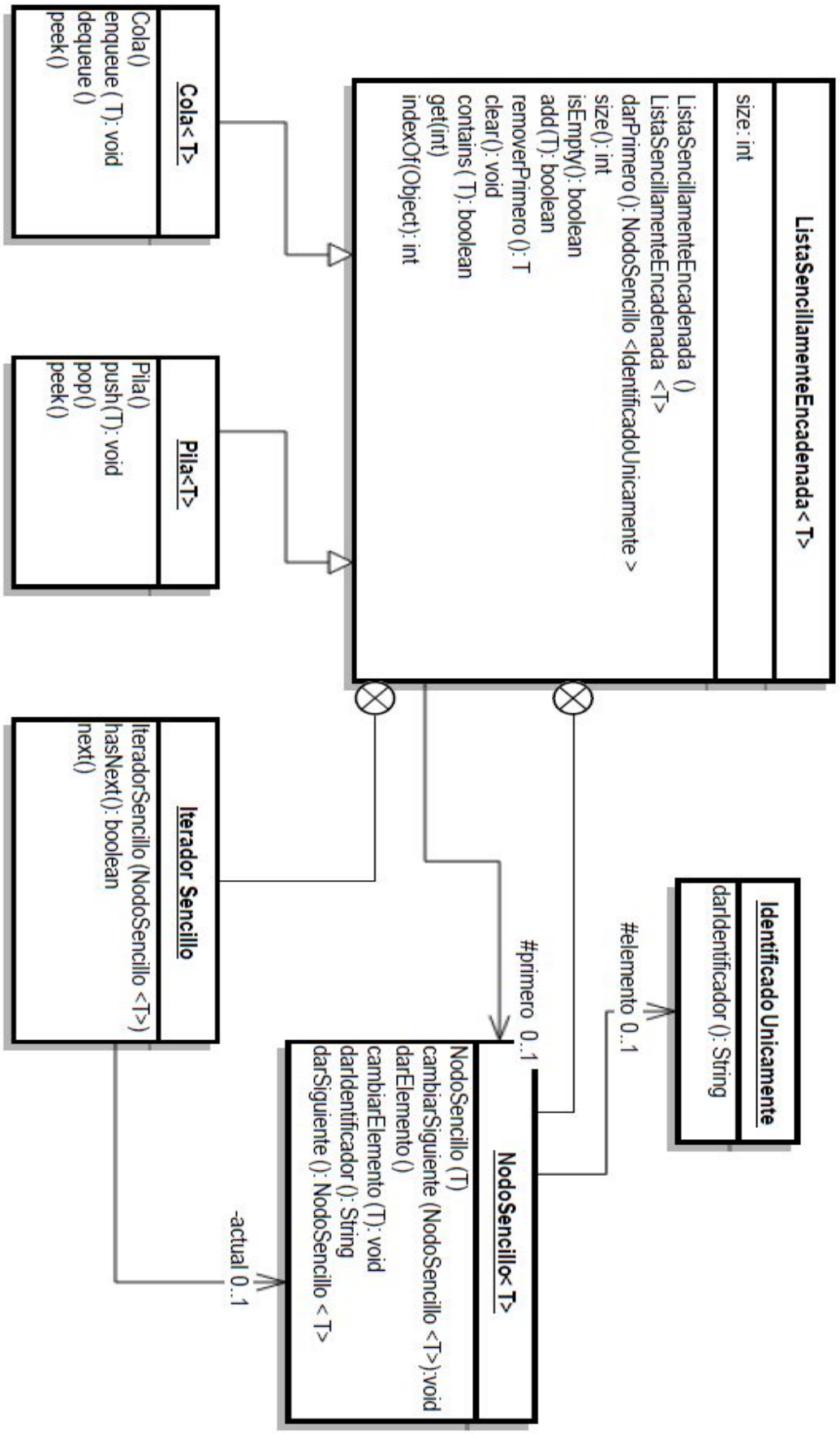
El uso de lista sencillamente encadenada nos permitirá organizar los datos que llegan del archivo Json, de una manera más eficiente ya que estas no tienen un tamaño definido, por el contrario el tamaño se va adaptando a las necesidades que tengamos. Además no necesita tener el espacio consecutivo en la memoria ya que puede buscar donde hay espacio y almacenar ahí los datos, por lo cual resulta menos costosa en espacio.
Para manejar los datos, las colas y las pilas son estructuras que nos permitirán organizar mejor los procesos que tenemos que hacer, la ventaja principal de la cola es que se pueden agregar y remover elementos fácilmente (al comienzo y al final), lo cual nos permitirá guardar las peticiones de manera en que puedan ser resueltas en el orden de llegada, y con las pilas podremos ir guardando temporalmente los datos cuando necesitemos ordenar o buscar un elemento en particular de una forma eficiente en tiempo.

Laura Maya

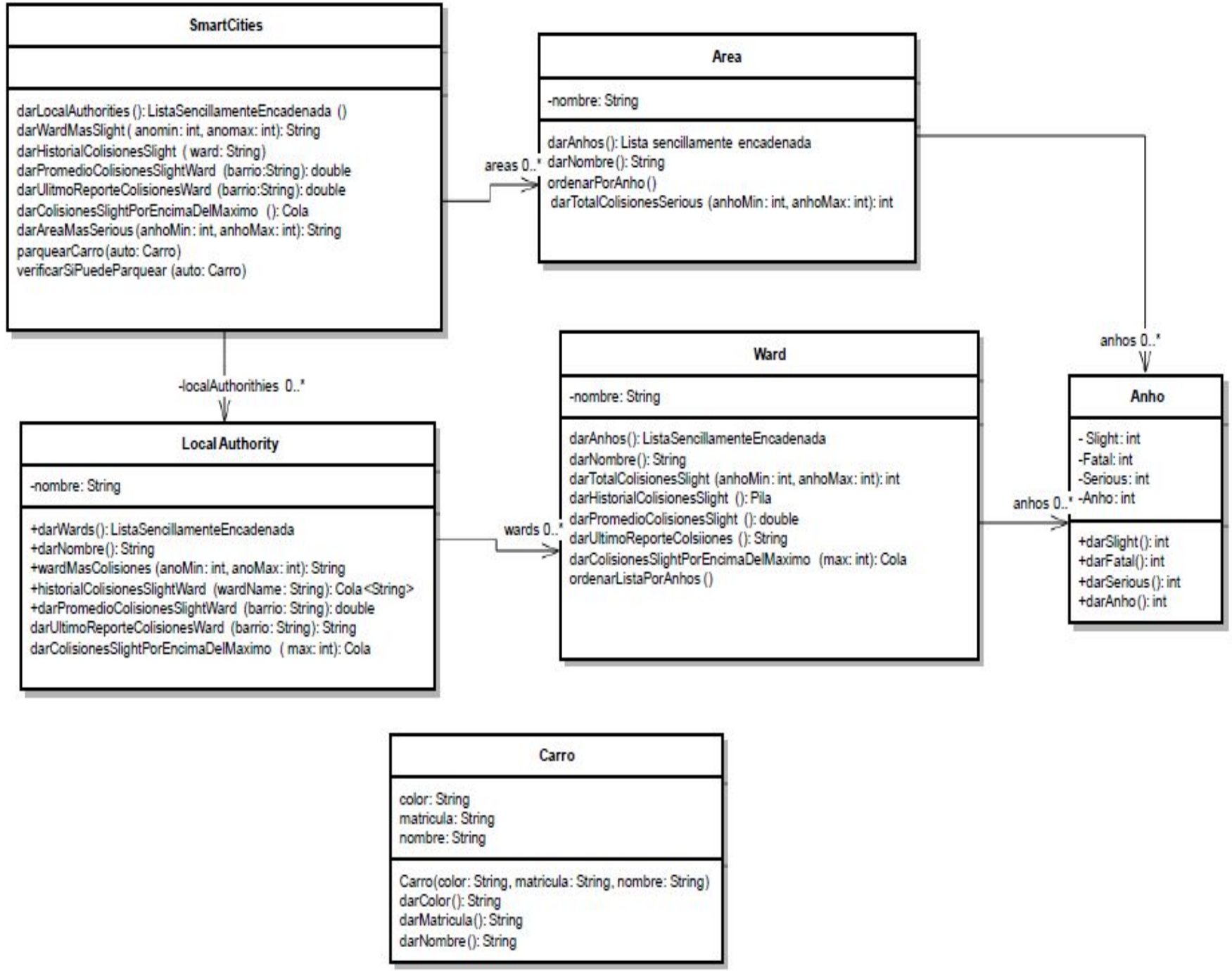
201423854

Carlos Peñaloza

201531973



Carlos Peñaloza
201531973



Laura Maya
201423854