**Estructuras de Datos 201810**

Francisco Javier González 201713245 – Carlos Fernando Infante 20171345

**ESTIMACIÓN DE COMPLEJIDAD REQUERIMIENTOS PROYECTO 1**

**Requerimiento 1A:** *Generar una Cola con todos los servicios de taxi que se prestaron en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final de consulta. El inicio y terminación del servicio debe estar incluido dentro del periodo de consulta. Los servicios deben mostrarse en orden cronológico de su fecha/hora inicial.*

Se piensa crear un método que reciba por parámetro el rango de fecha/hora que pide el usuario de manera que este mismo pueda retornar una lista de cola donde estén todos los servicios de taxi organizados de manera cronológica.

Primero, dentro del método como llegan los datos ordenados, toca es delimitar por medio del rango los datos que van a ser elegidos. Estos están dentro de un ArrayList implementado por nosotros y el cual va a ir agregando los elementos que están dentro del rango a una lista temporal dentro del método para finalmente retornarla. Se estima que la complejidad de este método es O(n) porque tiene que meter todos los elementos dentro del rango establecido en una lista y eso implica recorrer cada uno de dichos datos.

**Requerimiento 2A:** *Buscar el taxi de una compañía dada que más servicios inició en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final de consulta.*

El método puede tener por parámetro la compañía de la que se quiere el taxi y el rango de fecha/hora de los servicios. Ya con esto, se crea una temporal que guarde el mayor, mientras se recorre toda la lista haciendo una verificación que mire si algún taxi tiene más servicios que el temporal mayor, entonces lo asigna como ese taxi y así sucesivamente. Como se recorre toda la lista, la complejidad puede estimarse en O(n).

**Requerimiento 3A:** *Buscar la información completa de un taxi, a partir de su identificador, en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final de consulta. Incluye el nombre de su compañía y los valores totales de plata ganada, de servicios prestados, de distancia recorrida y de tiempo total de servicios.*

En este método se recibe por parámetro el ID del taxi a buscar y el rango fecha/hora de los servicios que se quieren de dicho taxi. Para eso, toca encontrar la compañía que posea dicho ID y dentro de él, recoger los datos de todos los servicios dentro del rango que posee el taxi, para así retornar finalmente toda su información. Se estima que la complejidad de todo esto es ya que toca encontrar el taxi dentro de las compañías y luego ir mirando cuales servicios de este último están dentro del rango delimitado por parámetro.

**Requerimiento 4A:** *Retornar una lista de rangos de distancia recorrida, en la que se encuentran todos los servicios de taxis servidos por las compañías, en una fecha dada y en un rango de horas especificada. La información debe estar ordenada por la distancia recorrida, así la primera posición de la lista tiene a su vez una lista con todos los servicios cuya distancia recorrida esta entre [0 y 1) milla. En la segunda posición, los recorridos entre [1 y 2) millas, y así sucesivamente.*

En este método se tiene por parámetro la fecha, hora inicial y final para saber de que manera se van a organizar los rangos de distancia pues estos están delimitados en un cierto tiempo. Se estima que la complejidad es de debido a que toca recorrer todos los servicios delimitados con las características que se mencionaron anteriormente, meterlos en una lista y luego estos meterlos dentro de otra lista que los secciona por rangos de millas, entonces se pueden hacer a partir de ciclos anidados.

**Requerimiento 1B:** *Mostrar la información de las compañías de taxi consistente en: El total de compañías que tienen al menos un taxi inscrito y el total de taxis que prestan servicio para al menos una compañía. Adicionalmente, generar la lista alfabética de compañías a las cuales aparecen inscritos los servicios de taxi de la fuente de datos de consulta. Por cada compañía debe informarse su nombre y el número de taxis que tiene registrados.*

El método asociado a este requerimiento puede tener una complejidad de ya que tiene que ir recopilando todos los datos de todas las compañías y dentro de ellas mirando la información de todos los taxis para así retornar una lista con todos estos datos. Por eso al hacer dos recorridos en ciclos se termina dictando la complejidad mencionada anteriormente.

**Requerimiento 2B:** *Buscar el taxi de una compañía dada que mayor facturación ha generado en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final de consulta*

En este caso, se busca por parámetro el nombre de la compañía y se da un rango de hora inicial y final desde donde se busca el taxi con mayor facturación. Con respecto a dichas especificaciones, se puede estimar que el método poseerá una complejidad de porque primero tiene que encontrar la compañía del parámetro, luego mirar los servicios de cada taxi e ir guardando en una temporal el que haya tenido mayor facturación. Esto requiere de un ciclo dentro de otro más uno anidado.

**Requerimiento 3B:** *Buscar la información completa de una zona de la ciudad en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final de consulta. El número total de servicios que se recogieron en la zona de consulta y terminaron en otra zona y el valor total pagado por los usuarios; el número total de servicios que se recogieron en otra zona y terminaron en la zona de consulta y el valor total pagado por los usuarios, y el total de servicios que iniciaron y terminaron en la misma zona de consulta y el valor total pagado por los usuarios.*

Como hay una clase que guarda todos los servicios que tiene una zona, tocaría buscar la zona con el ID que se da por parámetro, luego por medio de variables ir recopilando la información de todos los servicios, eso implica recorrer toda esa lista de la zona, por eso la complejidad se estima de gracias que hay dos ciclos anidados.

**Requerimiento 4B:** *Retornar una lista con todas las zonas de la ciudad (ordenadas por su identificador). Cada zona debe tener el total de servicios iniciados en dicha zona en un rango de fechas. Por ejemplo, la primera posición de la lista tiene todos los servicios de la primer zona, en dicha posición, se tiene una lista de fechas (ordenadas cronológicamente) con el total de servicios asociados a dicha fecha.*

Para este requerimiento se van a meter dentro de una lista todas las zonas y sus características. Por dicha razón, toca recorrer toda la lista e irla metiendo con todos sus datos, para así retornarla, por eso que la complejidad estimada es de O(n).

**Requerimiento 1C:** *Cargar toda la información de una fuente de datos seleccionada por el usuario (día, semana, mes o trimestre) y generar los VO y las estructuras de datos necesarias.*

Gracias a que todo está dentro de un solo archivo JSON y se quiere cargar a las clases, toca recorrer todos los objetos que posea este archivo. Por dicha razón, toca ir creando todas las compañías; dentro de ellas meter los taxis, luego los servicios con todas sus características. Esto implicaría hacer una complejidad de por cada ciclo que se vaya haciendo dentro del cargar. Es decir, la idea es poder almacenar la información dentro de las listas de tal manera que no sea necesario realizar 3 o más ciclos uno dentro del otro correspondientemente.

**Requerimiento 2C:** *Identificar el top X de compañías que más servicios iniciaron en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final. El valor X es un dato de consulta. El resultado debe mostrar el Top X de compañías ordenadas por el número de servicios de mayor a menor. Por cada compañía debe informarse su nombre y su número de servicios de respuesta*

Aquí toca revisar cada una de las compañías en un ciclo. Como ya se ha cargado la lista ordenadamente con respecto al número de servicios, solo toca retornar el número n de elementos que pida el usuario por parámetro dentro del rango establecido dentro de una nueva lista. En este caso la complejidad es O(n) al tener que ir recorriendo todos los datos de la lista.

**Requerimiento 3C:** *Buscar el taxi más rentable de cada compañía. El taxi más rentable de una compañía es aquel cuya relación de plata ganada y distancia recorrida en los servicios prestados es mayor.*

Toca empezar a revisar todos los servicios de cada uno de los taxis e ir revisando con un temporal cual es el que tiene la mayor relación de ganancia y distancia recorrida con sus servicios. Para esto toca mirar cada uno de los taxis y ahí mismo mirar todos sus servicios. Como hay dos ciclos, entonces se estima que la complejidad puede ser .

**Requerimiento 4C:** *Dada la gran cantidad de datos que requiere el proyecto, se desea poder compactar información asociada a un taxi particular. Para ello usted debe guardar en una pila todos los servicios generados por el taxi en orden cronológico, entre una hora inicial y una hora final, en una fecha determinada. Para comprimir la información, usted debe ir guardando los servicios reportados por el taxi, siempre y cuando la distancia acumulada de dichos servicios sea inferior a un valor dado (por ejemplo 10 millas). Cuando la sumatoria de las distancias de los servicios acumulados en la pila sobrepasen el valor límite, usted deberá retirar los servicios de la pila y generar uno nuevo, que resuma los servicios analizados. Este servicio resumen, tiene como hora inicial, la menor hora de los servicios, como hora final la mayor hora de los servicios, como distancia la sumatoria de todas las distancias de los servicios; como duración, la sumatoria de las duraciones de los servicios; y como valor total de la carrera, la sumatoria de todos los valores de los servicios. Los demás campos deben quedar en el valor por defecto dependiendo del tipo de dato.*

En este método se están revisando todas las características que posee un taxi con respecto a sus servicios dentro de un rango limitado por el usuario, el cual busca el ID del taxi, la hora inicial, final y la fecha de los datos que se quieren recopilar en una pila. Para realizar todo esto, se busca el taxi dentro de una lista, y se empiezan a recopilar todos los datos en el rango requerido. Esto implicaría dos ciclos, por lo que se estima que la complejidad puede ser de