Proyecto Final

Parte A:

Para este requerimiento la carga de datos se hizo en un mapa de compañías, donde las llaves eran el nombre de la compañía y los valores eran una lista normal de Python con dos entradas: un TAD Lista con todos los taxis de la compañía y un entero con la suma total de los viajes.

Número de taxis: La función es O(n) sobre las compañías (a lo sumo 55) y se hace la sumatoria de la size de cada TAD Lista (O(1)).

Número de Compañías: O(1) pues es simplemente sacar la size del Mapa Compañías

Top Compañías Taxis: La función es O(n) pues itera sobre todas las compañías del mapa para luego sacar la cantidad de afiliados de cada una; si bien luego hay otro ciclo, éste también es O(n) pero más pequeño y el orden permanece igual.

Top Compañías Servicios: La función es exactamente igual a la anterior, O(n)

Parte B:

En esta parte se crea un árbol RBT organizado por fechas y cuyos valores son un TAD lista con todos los viajes que se hicieron en esa fecha

Top Taxis Puntaje: Se hace la búsqueda de la fecha (O(log(n)) y se itera sobre la lista de viajes (O(n)) para encontrar el taxi al que corresponde el viaje y guardar en un mapa los valores necesarios para sacar el puntaje. Una vez se tienen todos los taxis y sus datos en el mapa, se itera sobre el mapa para sacar los puntajes y encontrar los mejores. Por lo tanto, el orden es O(n).

Top Taxis Puntaje Rango: La función hace lo mismo que la anterior pero para cada fecha en el rango. Como se hace una función O(n) hasta n veces, el orden es $O(n^2)$

Parte C:

Para la carga de datos de esta parte se creó un árbol de horas para poder sacar más fácilmente el rango y con los datos del rango de horas se crea el grafo.

Mejor Ruta: Se utiliza Dijsktra para buscar la mejor ruta en el grafo creado a partir de las horas y con la duración como peso; por lo tanto, la complejidad es O((V+E)log(V))