1. Documentación de los requerimientos funcionales (incluyendo los datos de entrada, la descripción, los datos de salida y estimación de complejidad temporal de cada requerimiento funcional).

Nota: Mencionar y justificar las estructuras de datos utilizadas para los requerimientos del proyecto.

1. Diseño de las Estructuras de Datos a utilizar (diagrama de clases UML, imagen)
2. Diseño de la Solución al proyecto (diagrama de clases UML, imagen)

**Documentación de los requerimientos funcionales**

**Requerimiento 0:** Cargar el Grafo No Dirigido (grafo más grande) de la malla vial de la ciudad completa de Bogotá creado en el taller 7. Informar el total de vértices y el total de arcos que definen el grafo cargado (cada arco debe contarse una única vez). Solo es permitido leer una vez la información de los archivos.

Datos de Entrada:

* Void

Datos de Salida:

* Int: Numero de vértices cargados
* Int: Numero de arcos cargados.

Estimación complejidad temporal:

* O(V)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Requerimiento 1:** Al grafo creado, se debe agregar la información de costo. El grafo va a tener 3 tipos de costo en sus arcos:

a. El primer tipo de costo de un arco es la distancia Harversine (en kilómetros) entre las localizaciones geográficas de los vértices que conecta. Esta distancia ya fue calculada en el taller 7.

b. El segundo tipo de costo de un arco es el tiempo de viaje entre sus vértices. Este se calcula tomando los promedios de tiempos de viaje (segundos) entre las zonas Uber (MOVEMENT\_ID) de sus vértices.   
Para tal fin, el costo en tiempo entre los vértices de un arco se define de la siguiente forma:

i. Si los vértices del arco pertenecen a la misma zona Uber, el peso es igual al tiempo promedio de los viajes Uber reportados en el trimestre donde la zona origen y destino es la misma. Si no existen tiempos promedios de viajes Uber para esa misma zona origen y destino, se asignará un costo predefinido de 10 segundos.

ii. Si los vértices del arco están en diferentes zonas Uber, el peso es igual al tiempo promedio de los viajes Uber reportados en el trimestre desde la zona del vértice origen hasta la zona del vértice destino. Si no existen tiempos promedios de viajes Uber entre la zona origen y la zona destino, se asignará un costo predefinido de 100 segundos.

C. El tercer tipo de costo es la velocidad del arco, calculada como su distancia dividida por su tiempo.

Datos de Entrada:

Datos de Salida:

Estimación complejidad temporal:

* O(V+E)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Requerimientos -Parte Inicial (ambos estudiantes)**

**Requerimiento 2:** Completar el grafo con los tres costos para cada arco a partir de los datos de Uber y la malla vial de Bogotá. Crear un archivo JSON para guardarlo y un método que permita cargar el nuevo grafo JSON generado.

Datos de Entrada:

Datos de Salida:

Estimación complejidad temporal:

* O(V+E)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 3:** Dada una localización geográfica con latitud y longitud, encontrar el Id del Vértice de la malla vial más cercano por distancia Haversine. Esta solución se deberá utilizar en algunos requerimientos de la parte A y parte B del proyecto.

Datos de Entrada:

* Double: Localización geográfica con latitud y longitud.

Datos de Salida:

* Int: Id del Vértice de la malla vial más cercano por distancia Haversine.

Estimación complejidad temporal:

* O(E)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimientos -Parte A (estudiante 1 de cada grupo)**

**Requerimiento 4:** Encontrar el camino de costo mínimo (menor tiempo promedio según Uber en la ruta) para un viaje entre dos localizaciones geográficas de la ciudad ((lat,long) origen, (lat, long) destino), ingresados por el usuario. Estas localizaciones deben aproximarse a las localizaciones más próximas en la malla vial.   
**Respuesta en consola:** Muestre en la consola de texto el camino a seguir, informando el total de vértices, sus vértices (Id, latitud, longitud), el costo mínimo (menor tiempo promedio en segundos) y la distancia estimada (sumatoria de distancias Haversine en Km).  
**Visualización mapa:** Muestre el camino resultante en Google Maps (incluyendo la ubicación de inicio y la ubicación de destino).

Datos de Entrada:

* Double: (lat,long) origen
* Double: (lat, long) destino

Datos de Salida:

* Camino de costo minimo.

Estimación complejidad temporal:

* O(E\*Log(V))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 5:** Determinar los n vértices con menor velocidad promedio en la ciudad de Bogotá. Siendo la velocidad promedio de un vértice v, el promedio de las velocidades de todos sus arcos. El parámetro n es un dato de entrada dado por el usuario.   
**Respuesta en consola:** Mostrar los n vértices resultantes en la consola de texto (su identificador, su ubicación (latitud, longitud), ordenados de menor a mayor por la velocidad promedio del vértice. Informar el número de componentes conectados (subgrafos) que se definen entre estos vértices en el grafo original. Por cada componente informar los identificadores de los vértices que la componen.   
**Visualización mapa:** marque la localización de los n vértices resultantes en un mapa en Google Maps usando un color 1. Destaque la componente conectada más grande (con más vértices) usando un color 2. Para esta componente muestre sus vértices y sus arcos.

Datos de Entrada:

* Int: Cantidad de vértices a consultar

Datos de Salida:

* Vértices con:
  + Id
  + Ubicación

Ordenados de menor a mayor por velocidad promedio de vértice.

* Numero de componentes conectados
  + Por cada componente id de vértices que la componen.

Estimación complejidad temporal:

* O(V\*E)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 6:** Calcular un árbol de expansión mínima (MST) con criterio distancia, utilizando el algoritmo de Prim, aplicado al componente conectado (subgrafo) más grande de la malla vial de Bogotá.   
**Respuesta en consola:** Muestre en la consola de texto el tiempo que toma el algoritmo en encontrar la solución (en milisegundos), y la siguiente información del árbol generado: el total de vértices en el componente, los vértices (identificadores), los arcos incluidos (Id vértice inicial e Id vértice final) y el costo total (distancia en Km) del árbol.  
**Visualización mapa:** Muestre árbol generado resultante en Google Maps: sus vértices y sus arcos.

Datos de Entrada:

* void

Datos de Salida:

* Double: Tiempo que tarda en encontrar un MST.
  + Número total de vértices componente
  + Id de cada vértice
  + Arcos incluidos
  + Costo total árbol

Estimación complejidad temporal:

* O(E\*Log(V))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimientos -Parte B (estudiante 2 de cada grupo)**

**Requerimiento 7:** Encontrar el camino de menor costo (menor distancia Haversine) para un viaje entre dos localizaciones geográficas de la ciudad ((lat,long) origen, (lat, long), destino), ingresadas por el usuario. Estas localizaciones deben aproximarse a las localizaciones más próximas en la malla vial.   
**Respuesta en consola:** Muestre en la consola de texto el camino a seguir, informando el total de vértices, sus vértices (Id, latitud, longitud), el tiempo estimado (la sumatoria de los tiempos de sus arcos) y la distancia Haversine estimada (sumatoria de distancias Haversine en Km).  
**Visualización mapa:** Muestre el camino resultante en Google Maps (incluyendo la ubicación de inicio y la ubicación de destino).

Datos de Entrada:

* Localizaciones geográficas:
  + Origen: (Double Lat, Double Long)
  + Destino: (Double Lat, Double Long)

Se aproximan a la localización más cercana en malla vial.

Datos de Salida:

* Camino a Seguir:
  + Número total de vértices
  + Id, Latitud, Longitud de cada vertice
  + Tiempo estimado
  + Distancia Haversine

Estimación complejidad temporal:

* O(N)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 8:** A partir de las coordenadas de una localización geográfica de la ciudad (lat, lon) de origen, indique cuáles vértices son alcanzables para un tiempo T (en segundos) dado por el usuario. La localización de origen debe aproximarse a la localización más próxima en la malla vial.  
**Respuesta en consola:** Muestre en la consola los identificadores y la ubicación (lat, lon)de los vértices alcanzables en un tiempo T a partir de la localización de origen.   
**Visualización mapa:** Marque la localización de origen en un color 1 y las localizaciones de los vértices alcanzables en un color 2 en Google Maps.

Datos de Entrada:

* Localización geográfica:
  + Double Lat, Long de origen
* Double tiempo t en segundos

Datos de Salida:

* Para cada vértice alcanzable en t segundos:
  + Int ID
  + Double Latitud, Longitud

Estimación complejidad temporal:

* O(E)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 9:** Calcular un árbol de expansión mínima (MST) con criterio distancia, utilizando el algoritmo de Kruskal, aplicado al componente conectado (subgrafo) más grande de la malla vial de Bogotá. Respuesta en consola: Muestre en la consola de texto el tiempo que toma el algoritmo en encontrar la solución (en milisegundos), y la siguiente información del árbol generado: el total de vértices en la componente, los vértices (identificadores), los arcos incluidos (Id vértice inicial e Id vértice final) y el costo total (distancia en Km) del árbol.  
**Visualización mapa:** Muestre el árbol generado resultante en Google Maps: sus vértices y sus arcos.

Datos de Entrada:

* Void

Datos de Salida:

* Double tiempo que demora el algoritmo en calcular la solución
* Del árbol generado:
  + Total de vértices en el componente
  + Id de cada vertice
  + Arcos incluidos (id vert inicial y final)

Estimación complejidad temporal:

* O(E\*Log(V))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Parte C (trabajo en grupo)**

**Requerimiento 10:** Construir un nuevo grafo simplificado No dirigido de las zonas Uber, donde cada zona (MOVEMENT\_ID) es representada con un único vértice y los enlaces entre ellas representan su vecindad dentro de la malla vial. Es decir, si existe al menos un arco entre los vértices idA y idB en la malla vial y estos vértices pertenecen a la zonaX y la zonaY de Uber respectivamente, entonces las zonas zonaX y zonaY son vértices del grafo de zonas y estas zonas son adyacentes (vecinas) en este grafo. NO considerar los arcos que definan auto-ciclos entre una misma zona. Para cada zona seleccione una única localización (latitud, longitud) de referencia de manera que pueda luego visualizarse en el mapa. Adicionalmente, el costo del arco entre dos zonas vecinas corresponde al tiempo de viaje promedio reportado por Uber entre dichas zonas, teniendo en cuenta los tiempos promedio para los días (domingo, lunes, ..., sábado) en que Uber haya reportado viajes entre dichas zonas. En caso de NO existir ningún tiempo de viajes en los datos Uber entre las dos zonas, asumir un valor predefinido de 200 segundos. Entre dos vértices (zonas) adyacentes solo debe crearse un arco, sin importar que pueda haber muchas intersecciones en común entre dichas zonas.  
**Respuesta en consola:** Al final de la construcción del grafo de zonas, reportar la cantidad de vértices y arcos (cada arco debe contarse una única vez).  
**Visualización mapa:** Muestre el grafo resultante en Google Maps: sus vértices (usando su localización de referencia) y sus arcos. Los arcos entre zonas conectan sus localizaciones de referencia.

Datos de Entrada:

Datos de Salida:

Estimación complejidad temporal:

* O(N)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Requerimiento 11:** Calcular el camino de costo mínimo (algoritmo de Dijkstra) basado en el tiempo promedio entre una zona de origen y una zona de destino sobre el grafo de zonas.   
**Respuesta en consola:** Muestre en la consola de texto el tiempo que toma el algoritmo en encontrar la solución (en milisegundos) y del camino resultante: su secuencia de vértices/zonas (MOVEMENT\_ID) y su costo total (sumatoria de tiempos de los arcos en segundos).

Adicionalmente, mostrar el tiempo promedio desde la zona origen hasta la zona destino reportado por el archivo de resolución semanal de Uber (teniendo en cuenta los tiempos promedio para los días (domingo, lunes, ..., sábado) en que Uber haya reportado viajes entre dichas zonas).Esto con el propósito que el usuario pueda comparar el tiempo estimado en el grafo de zonas y el tiempo estimado con los datos de Uber entre la zona origen y la zona destino.  
  
**Visualización mapa:** Muestre el camino de costo mínimo en Google Maps: sus vértices (usando su localización de referencia) y sus arcos. Los arcos entre zonas conectan sus localizaciones de referencia.

Datos de Entrada:

Datos de Salida:

Estimación complejidad temporal:

* O(E\*Log(V))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Requerimiento 12:** A partir de una zona origen, calcular los caminos de menor longitud (cantidad de arcos) a todas sus zonas alcanzables. De estos caminos, seleccionar el camino más largo (mayor cantidad de arcos); este será el camino desde la zona origen a su zona más distante (teniendo en cuenta el número de arcos) en el grafo de zonas.  
   
**Respuesta en consola:** Muestre en la consola de texto el tiempo que toma el algoritmo en encontrar la solución (en milisegundos) y del camino resultante más largo: su secuencia de vértices/zonas (MOVEMENT\_ID) y su número total arcos. Si hay múltiples caminos con la mayor longitud (número de arcos), mostrar entre estos caminos aquel que llegue al vértice destino con menor MOVEMENT\_ID.  
  
**Visualización mapa:** Muestre el camino más distante desde la zona de origen en Google Maps: sus vértices (usando su localización de referencia) y sus arcos. Los arcos entre zonas conectan sus localizaciones de referencia.

Datos de Entrada:

Datos de Salida:

Estimación complejidad temporal:

* O(E\*Log(V))