1. Con base en la información del archivo, ¿qué elegirían vértices y qué como aristas en el grafo?

En el grafo que representa el sistema de buses, los vértices deben ser las combinaciones de parada y ruta, es decir, cada vértice tendrá el formato BusStopCode-ServiceNo. Esto se debe a que una misma parada puede pertenecer a varias rutas, y es necesario distinguirlas para mantener la estructura del grafo coherente. Las aristas representan las conexiones dirigidas entre paradas consecutivas dentro de una misma ruta y dirección, y el peso de cada arista será la distancia entre esas paradas.

2. ¿Qué tipo de grafo es más adecuado para representar la información?

El tipo de grafo más adecuado para esta representación es un grafo dirigido y ponderado. Es dirigido porque las rutas de los buses tienen un sentido específico de recorrido, indicado por el campo Direction, y es ponderado porque las distancias entre paradas no son uniformes, y por lo tanto es importante tenerlas en cuenta como pesos de las aristas. Este tipo de grafo permite modelar fielmente las rutas de los buses y facilita la aplicación de algoritmos de caminos mínimos o de análisis de conectividad.

3. ¿Se deben crear varios vértices para una misma parada, o se usarán las aristas para diferenciar las rutas?

Sí, se deben crear varios vértices para una misma parada cuando esta pertenece a múltiples rutas. No es suficiente con usar aristas para distinguir las rutas, ya que esto podría mezclar trayectos distintos y dificultar el análisis. Al representar cada combinación de parada y ruta como un vértice único, se asegura que las conexiones entre paradas respeten la lógica de cada ruta individual, lo que es fundamental para analizar trayectorias, transferencias y recorridos específicos.

4. ¿Es útil un grafo para analizar redes de transporte? Justifiquen su respuesta.

Un grafo es una herramienta muy útil para analizar redes de transporte porque permite representar las paradas como nodos y las conexiones entre ellas como aristas, lo que facilita el análisis de conectividad, accesibilidad y eficiencia. Además, permite aplicar algoritmos clásicos como Dijkstra para encontrar rutas más cortas, o DFS y BFS para explorar posibles trayectorias y detectar ciclos o puntos de transbordo. También permite identificar cuellos de botella, optimizar rutas y simular escenarios de operación o emergencia.

5. ¿Qué información del archivo les permitiría identificar si hay una conexión directa (una arista) entre dos paradas específicas?

Para identificar una conexión directa entre dos paradas específicas, se deben revisar los campos ServiceNo, Direction y StopSequence. Si dos paradas pertenecen a la misma ruta (ServiceNo) y a la misma dirección (Direction), y si sus secuencias (StopSequence) son consecutivas, entonces existe una conexión directa entre ellas. La distancia entre las paradas se puede deducir a partir de la diferencia de los valores del campo Distance entre ambas filas del archivo.

6. Si encuentran ciclos en el grafo, ¿qué podrían representar en el contexto de las rutas de autobuses?

Los ciclos en el grafo podrían representar rutas circulares dentro del sistema de transporte. Estos ciclos indican que un bus puede retornar a una parada anterior dentro de la misma ruta sin necesidad de transbordos, lo cual es común en rutas que operan en forma de lazo o circuito cerrado. Detectar estos ciclos es útil tanto para los usuarios que desean regresar a su punto de origen, como para los operadores que quieren analizar la eficiencia y cobertura de sus rutas.

Al finalizar las modificaciones y hacer pruebas funcionales ejecutando ambos algoritmos de BFS y DFS considere las siguientes preguntas y respóndalas en el documento de Observaciones Lab 11:

a) ¿Existe alguna diferencia entre los resultados encontrados por BFS y DFS?
Tras ejecutar completamente el laboratorio e implementar las funciones relacionadas con la búsqueda en profundidad (DFS) y en amplitud (BFS), se realizaron varias pruebas utilizando diferentes estaciones base y de destino. Pudimos observar que, aunque ambos algoritmos cumplen con el propósito de identificar componentes conexos y rutas dentro del grafo de estaciones, existen diferencias en el orden de recorrido de los vértices y en el camino específico que trazan desde una estación base a una estación de destino.

En el caso de DFS, el algoritmo tiende a profundizar en un camino lo más posible antes de retroceder, lo que puede llevar a encontrar rutas más largas o con mayor cantidad de saltos si no se encuentra una conexión directa temprana. Esto hace que el orden en que se visitan los vértices y el recorrido registrado sea más dependiente de la estructura del grafo y de la posición de las conexiones más profundas.

Por otro lado, BFS realiza una exploración por niveles, es decir, primero visita todos los vecinos inmediatos de un vértice antes de continuar con los vecinos de esos vecinos. Gracias a esto, BFS tiende a encontrar rutas más cortas en número de arcos entre la estación base y una estación destino, siempre que tales rutas existan, ya que prioriza las conexiones más cercanas.

A pesar de estas diferencias en la forma en que cada algoritmo recorre el grafo, los resultados respecto a la existencia de rutas o de componentes fuertemente conexos fueron consistentes entre ambos. Es decir, si una estación era alcanzable desde otra con DFS, también lo era con BFS, y viceversa. Lo mismo sucedió al identificar componentes conexos: aunque los caminos pueden diferir, los conjuntos de estaciones agrupadas dentro de un mismo componente no cambiaron entre ambos algoritmos.

En resumen, sí hay diferencias en la estrategia de recorrido y en los caminos específicos que se generan, pero no hubo contradicciones en cuanto a la conectividad general del grafo ni en los resultados funcionales requeridos para el laboratorio.