En esta evaluación a usted se le pide programar dos implementaciones para resolver el **problema de rutas mínimas** en una red de transporte. La información disponible de esta red es:

- Conjunto de nodos
- Conjunto de arcos, dados como pares de nodos
- Costo de transporte de cada arco (también llamado distancia).

Una ruta mínima entre dos nodos de la red es la ruta que minimiza la distancia total de viaje. Esto es, la suma de las distancias de los arcos que componen la ruta.

En el problema de rutas mínimas que resolveremos se requiere conocer todas las rutas mínimas desde un nodo fijo y conocido (que llamaremos *nodo fuente*) hacia todos los otros nodos de la red (*nodos sumidero*). Por las propiedades del problema de rutas mínimas, estas rutas siempre pueden formar un árbol de rutas mínimas. Este árbol puede ser representado de forma tal que cada nodo tiene un predecesor en la ruta mínima al nodo fuente, de modo que la ruta mínima entre dos nodos se puede componer desde el nodo sumidero hacia atrás, hasta llegar al nodo fuente.

A usted se le pide programar y analizar la ejecución de dos implementaciones de algoritmos para resolver el problema de rutas mínimas. Ambas implementaciones son variaciones del Algoritmo de Dijkstra. Específicamente, se le pide programar:

- El algoritmo original de Dijkstra. En este algoritmo todos los nodos tienen una etiqueta compuesta por la mejor distancia que se conoce desde el nodo fuente hasta el nodo y el predecesor del nodo si se siguiera la mejor ruta conocida. A lo largo del algoritmo se va fijando el valor de esta etiqueta, una vez que se conoce la mejor ruta (ruta mínima) y la distancia asociada a esta ruta (ruta mínima). En cada iteración del algoritmo se revisa los nodos con etiqueta temporal y se fija la etiqueta del nodo que tiene un menor valor temporal para la etiqueta de distancia. Se actualiza las etiquetas y se sigue con la siguiente iteración hasta que se fija las etiquetas de todos los nodos. La ruta mínima de cada nodo se conoce desde el nodo sumidero hasta el nodo fuente, a través de los predecesores de las etiquetas ya fijadas. Este algoritmo se detalla en el capítulo 4.5 del texto adjunto[1].

- La Implementación de Dial del algoritmo de Dijsktra, también conocido como Algoritmo de Dial. Se sabe que el cuello de botella del Algoritmo de Dijkstra es la selección del nodo que se fija en cada iteración, ya que se debe revisar todos los nodos cuyas etiquetas no se han fijado aún. La implementación de Dial permite guardar los nodos en una lista de baldes (buckets), de modo que no es necesario revisarlos todos en cada iteración. Concretamente, el hecho de que el valor de la etiqueta de distancia que se fija en cada iteración nunca decrece (sólo sube o se mantiene con el tiempo) permite guardar los nodos en baldes con distintas etiquetas de distancia. En cada iteración se busca el siguiente balde con nodos sin

[1] Capítulo 4 del libro Network Flows, de Ahuja, Magnanti y Orlin.

fijar y se va actualizando el balde de los nodos asociados al nodo que se está fijando. Tal algoritmo se detalla en el capítulo 4.6 del texto adjunto.

Es sabido que la complejidad del algoritmo de Dial es menor que la del de Dijkstra. Sin embargo, esto sólo garantiza menor número de operaciones en "peores casos". En una instancia cualquiera, cualquiera de los algoritmos puede ser más rápido. Para efectos de análisis, a usted se le pide que compare el desempeño de ambos algoritmos, en términos de tiempo de resolución, para el conjunto de instancias adjuntas.

Su implementación debe ser capaz de leer una instancia cualquiera del problema de rutas mínimas, desde dos archivos de texto plano:

- Un archivo llamado "nodos.txt" que contiene una línea con la cantidad de nodos de la instancia. En lo que sigue, los nodos serán nombrados correlativos desde 1 hasta la cantidad de nodos.

- Un archivo llamado "arcos.txt" que contiene tantas líneas como arcos tiene la instancia. Cada línea contiene 3 valores enteros: el número del nodo inicial, el número del nodo final y el costo del arco. Después de la línea correspondiente al último nodo, el archivo contendrá una línea con el string "EOF", indicando el fin del archivo.

Al ejecutar cada programa, éste debe solicitar al usuario (a través de un archivo, consola o lo que usted estime conveniente) el nodo fuente desde donde se debe calcular las rutas mínimas. Al final de la ejecución, el programa debe entregar un archivo de salida llamado "salida.txt" que debe contener una línea inicial que indique el nodo fuente y, en seguida, una línea por cada nodo de la instancia. Cada una de estas líneas debe contener 3 valores enteros: el número del nodo, su predecesor y la etiqueta de distancia de ruta mínima. En el caso que algún nodo no sea alcanzable desde el nodo fuente, el valor del predecesor debe ser 0 y el valor de la etiqueta de distancia debe ser -1. Para el nodo fuente, su distancia será obviamente 0 y su predecesor será él mismo.

Otras indicaciones:

- Usted debe programar los algoritmos utilizando python. Para ello, debe tomar en cuenta las estructuras de datos más adecuadas para las operaciones que requiere cada algoritmo.
- En el caso que usted lo requiera, puede almacenar o consultar datos fuera de su implementación (por ejemplo, en BB.DD). Si fuera el caso, por favor entregar la justificación de la tecnología seleccionada.
- Se valora el orden y el estilo de programación.