Grafos
Dijkstra
Floyd
Recorrido en profundidad

Dra. MPuerto Paule Ruiz

Estos apuntes son notas de clase realizadas por el profesor. El lector ha de complementarlas con las sesiones impartidas de manera presencial

Dijkstra

- **public** *DijkstraDataClass* **dijkstra (T source)**→ Método que implementa el algoritmo de Dijkstra desde un nodo origen.
  - Devuelve una DijkstraDataClass con los vectores D y P correspondientes si se aplica o null si no se puede aplicar
- *DijkstraDataClass*→ Clase con el vector D (costes) y vector P (Path)
- Pruebas unitarias

Floyd

- ***public boolean floyd()*** → Método que implementa el algoritmo de Floyd
  - Devuelve true si lo aplica y false en caso contrario
- ***protected double[][] getFloydA()***

  Método que devuelve la matriz A de Floyd
- ***public double minCostPath(T nodoOrigen, T nodoDestino)*** →

  Devuelve el coste del camino de coste mínimo entre origen y destino según Floyd.
  - Si los nodos source o target no existen, lanza una excepcion de tipo ElementNotPresentException
- **public String path(T origen, T destino)** → Indica el camino entre los nodos que se le pasan como parámetros con el formato siguiente: Origen<tab>(coste0)<tabr>Inter1<tab>(coste1)<tabulador>InterN<tab>(costeN)<tab>Destino<tab>
  - Si no hay camino: Origen<tab>(Infinity)<tab>Destino<tab>
  - Si Origen y Destino coinciden: Origen<tab>
  - Si no existen los 2 nodos devuelve una cadena vacía

Recorrido en profundidad

- public String recorridoProfundidad(T nodo)→ Lanza el recorrido en profundidad de un grafo desde un nodo determinado
  - Al recorrer cada nodo añade el toString del nodo y un tabulador
  - Si no existe el nodo devuelve una cadena vacía