TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CULIACÁN

MATERIA: INTELIGENCIA ARTIFICIAL

MAESTRO: JOSE MARIO RIOS FELIX

ALUMNO: GUILLERMO EDUARDO PADILLA NORIEGA

Dijkstra

El algoritmo del camino más corto de Dijkstra fue inventado en 1956 por el científico informático holandés Edsger W. Dijkstra durante una pausa para el café de veinte minutos, mientras estaba de compras con su prometida en Ámsterdam.

La razón para inventar el algoritmo fue probar una nueva computadora llamada ARMAC.

Cómo funciona:

1. Establezca distancias iniciales para todos los vértices: 0 para el vértice de origen e infinito para todos los demás.
2. Seleccione el vértice no visitado con la distancia más corta desde el inicio como vértice actual. De esta forma, el algoritmo siempre comenzará con el origen como vértice actual.
3. Para cada uno de los vértices vecinos no visitados del vértice actual, calcule la distancia desde la fuente y actualice la distancia si la nueva distancia calculada es menor.
4. Ya hemos terminado con el vértice actual, así que lo marcamos como visitado. Un vértice visitado no se vuelve a verificar.
5. Regrese al paso 2 para elegir un nuevo vértice actual y siga repitiendo estos pasos hasta que haya visitado todos los vértices.
6. Al final nos queda el camino más corto desde el vértice de origen hasta todos los demás vértices del gráfico.

```
package IA;
import java.util.*;

public class Dijkstra {
    static class Arista {
        int destino;
        int peso;

        Arista(int destino, int peso) {
            this.destino = destino;
            this.peso = peso;
        }
    }

    static class Nodo implements Comparable<Nodo> {
        int id;
        int distancia;

        Nodo(int id, int distancia) {
            this.id = id;
            this.distancia = distancia;
        }
```

```
    public int compareTo(Nodo otro) {
        return Integer.compare(this.distancia, otro.distancia);
    }
}

public static void dijkstra(List<List<Arista>> grafo, int origen) {
    int n = grafo.size();
    int[] distancias = new int[n];
    Arrays.fill(distancias, Integer.MAX_VALUE);
    distancias[origen] = 0;

    PriorityQueue<Nodo> cola = new PriorityQueue<>();
    cola.add(new Nodo(origen, 0));

    while (!cola.isEmpty()) {
        Nodo actual = cola.poll();
        int u = actual.id;

        for (Arista arista : grafo.get(u)) {
            int v = arista.destino;
            int peso = arista.peso;
```

```
                if (distancias[u] + peso < distancias[v]) {
                    distancias[v] = distancias[u] + peso;
                    cola.add(new Nodo(v, distancias[v]));
                }
            }
        }

        System.out.println("Distancias desde el nodo " + origen + ":");
        for (int i = 0; i < n; i++) {
            System.out.println("→ Nodo " + i + ": " + distancias[i]);
        }
    }

    public static void main(String[] args) {
        int nodos = 5;
        List<List<Arista>> grafo = new ArrayList<>();

        for (int i = 0; i < nodos; i++) {
            grafo.add(new ArrayList<>());
        }
```

```
        grafo.get(0).add(new Arista(1, 10));
        grafo.get(0).add(new Arista(2, 3));
        grafo.get(1).add(new Arista(2, 1));
        grafo.get(1).add(new Arista(3, 2));
        grafo.get(2).add(new Arista(1, 4));
        grafo.get(2).add(new Arista(3, 8));
        grafo.get(2).add(new Arista(4, 2));
        grafo.get(3).add(new Arista(4, 7));
        grafo.get(4).add(new Arista(3, 9));

        dijkstra(grafo, 0);
    }
}
```