16. Sea D un digrafo conexo que no tiene ciclos dirigidos, v el único vértice de D con grado de entrada 0 (Dicho vértice siempre existe, ver guías 2 y 3) y $c : E(D) \to \mathbb{Z}$ una función de pesos.

a. Definir una función recursiva $d : V(D) \to \mathbb{Z}$ tal que $d(w)$ es el peso del camino mínimo de v a w para todo $w \in V(D)$. Ayuda: considerar que el camino mínimo de v a w se obtiene yendo de v hacia z y luego tomando la arista $z \to w$, para algún vecino de entrada z de w; notar que la función recursiva está bien definida porque D no tiene ciclos.

$$d(w) = \begin{cases} 0 & \text{si } w = v \\ \min_{\text{vecino } z \in N^-(w)} (d(z) + c(z, w)) & \text{si } w \neq v \end{cases}$$

b. Diseñar un algoritmo de programación dinámica top-down para el problema de camino mínimo en digrafos sin ciclos y calcular su complejidad.

Como nuestra función tiene como máximo n argumentos (uno por cada vértice en el grafo) y claramente hay superposición de subproblemas, entonces podemos usar programación dinámica.

Construimos un vector de n posiciones:

$$d(w) = \begin{cases} \text{memo}[w] & \text{si memo}[w] \neq \text{INDEF} \\ 0 & \text{si } w = v \\ \min_{\text{vecino } z \in N^-(w)} (d(z) + c(z, w)) & \text{si } w \neq v \end{cases}$$

La complejidad es $O(n + m)$, ya que recorremos todos los nodos y aristas.

c. (Integrador y opcional) Diseñar un algoritmo de programación dinámica bottom-up para el problema. Ayuda: computar d de acuerdo a un orden topológico $v = v_1, \ldots, v_n$ donde $v_i \to v_j$ solo si $i < j$. Este orden se puede computar en $O(n + m)$ (guía 3).

Algorithm 1 Computación de caminos mínimos en un digrafo acíclico (bottom-up)

Computar un orden topológico $v = v_1, \ldots, v_n$ del grafo D
Inicializar vector d con distancias infinitas: $d[w] = \infty$ para todo $w \in V(D)$
$d[v] = 0$
for $i = 1$ **to** n **do**
    for cada vecino z de v_i en $N^-(v_i)$ **do**
        $d[v_i] = \min(d[v_i], d[z] + c(z, v_i))$
    end for
end for