

# Video39 (1)

---

## Conceptos Principales

---

1. a
2. b
3. c

## Notas

---

### Resumen del algoritmo de Warshall (matriz de caminos)

La idea general es la misma del video anterior: queremos construir **la matriz de caminos** de un grafo. La matriz de adyacencia solo dice si hay un **arco directo** entre dos nodos; la matriz de caminos dice si existe **cualquier camino**, directo o usando nodos intermedios.

Warshall resuelve justo eso: ver si se puede ir del nodo  $i$  al nodo  $j$  pasando por cualquier cantidad de nodos intermedios, de manera eficiente y sin multiplicar matrices.

---

### Cómo piensa Warshall

Warshall parte de la matriz de adyacencia, pero trabaja sobre una copia; esa copia se va convirtiendo en la matriz de caminos.

La pregunta que responde para cada par  $(i, j)$  es:

**¿Existe un camino directo o puedo llegar pasando por algún otro nodo  $k$ ?**

Entonces evalúa dos opciones:

1. Ya había camino entre  $i$  y  $j \rightarrow$  queda en 1.
2. No había camino directo, pero existe un nodo  $k$  tal que:
  - se puede ir de  $i$  a  $k$
  - y se puede ir de  $k$  a  $j$

Si eso pasa, entonces pone 1 en  $(i, j)$ .

Para que esto funcione, el algoritmo recorre todos los  $k$  posibles. Es básicamente:

para cada  $k$

para cada  $i$

para cada  $j$

actualice si hay camino usando  $k$  como puente.

---

## Por qué se basa en lógica booleana

El algoritmo usa la idea de OR y AND:

- **AND:** para ver si  $i \rightarrow k$  y  $k \rightarrow j$  existen.
- **OR:** para decir "si ya existía camino directo, lo dejo; si aparece uno nuevo usando k, también lo marco".

Por eso a veces se usan operaciones tipo "mínimo(1, ...)" para que nunca aparezca un 2 cuando se suman dos 1; todo se mantiene booleano.

---

## Qué problema resuelve

Warshall sirve cuando querés saber:

### ¿Se puede ir del nodo i al nodo j con algún camino posible?

No te da cuántos saltos lleva, solo si existe o no. Eso basta para muchos problemas de conectividad, análisis de grafos, o verificar dependencias entre partes de un programa.

---

## Por qué es práctico para programar

- Solo requiere **tres ciclos anidados**, nada más.
- No hay multiplicación de matrices ni cálculos raros.
- Es más liviano que el método del cierre transitivo por potencias.
- Funciona muy bien si ya tenés el grafo como **matriz de adyacencia**.

En C es directo: solo estás revisando condiciones y poniendo ceros/unos según si aparece camino pasando por k.

---

## Lo que te conviene recordar para examen / tareas

- Warshall transforma la matriz de adyacencia en la matriz de caminos.
- La regla clave es:

$$M[i][j] = M[i][j] \text{ OR } (M[i][k] \text{ AND } M[k][j])$$

- Se recorren todos los k intermedios.
- Resultado: una matriz que dice si existe *algún* camino, sin importar cuántos pasos.