UNIVERSIDAD AUTONOMA DE CHILE

DUC IN ALTUM

MÁS UNIVERSIDAD

ANÁLISIS DE ALGORITMOS

Mg. Marcos Contreras F.

SER PUNTUAL

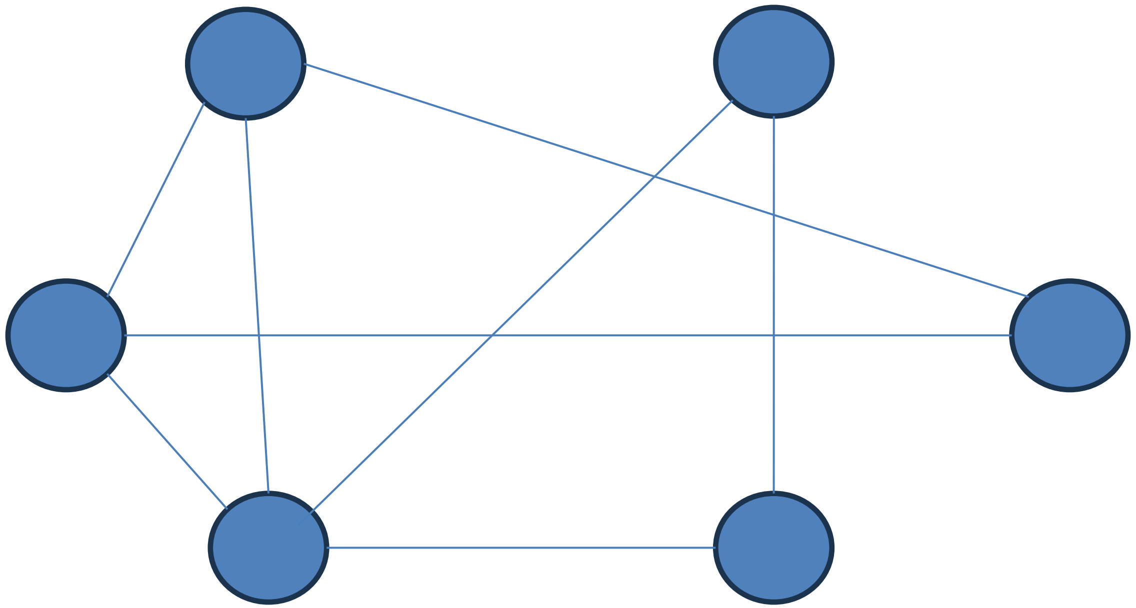
"es valorar tu tiempo y el mío"

Teoría de Grafos

Un **grafo** es una estructura que consta de un conjunto de **vértices** (o nodos) y un conjunto de **aristas** (o enlaces) que conectan pares de vértices.

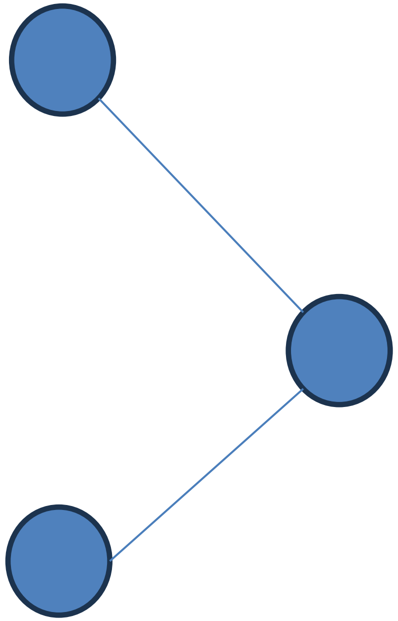
Formalmente, un grafo G se define como G = (V, E), donde:

- V es el conjunto de vértices.
- E es el conjunto de aristas.

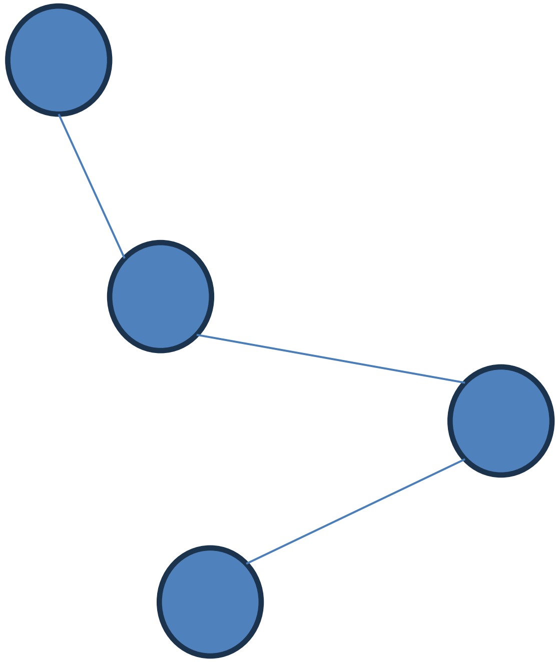


Un grafo está formado por un conjunto de vértices y otro de aristas que los conectan.

El grado de un vértice o nodo es el número de aristas o enlaces que están conectadas a él.

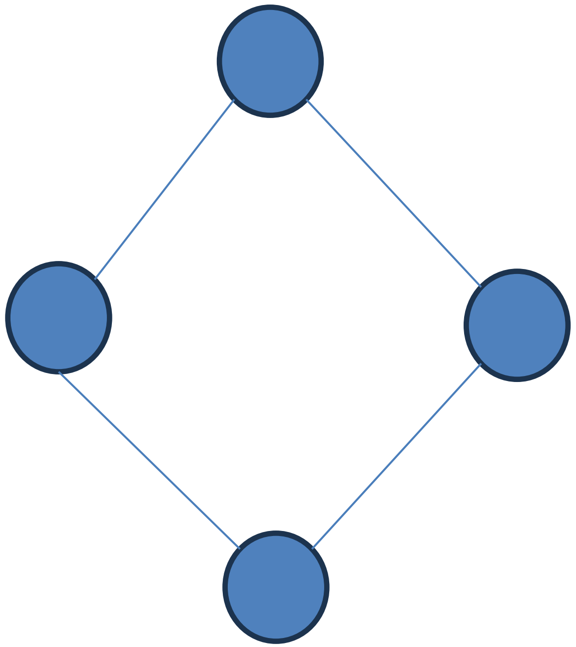


P3

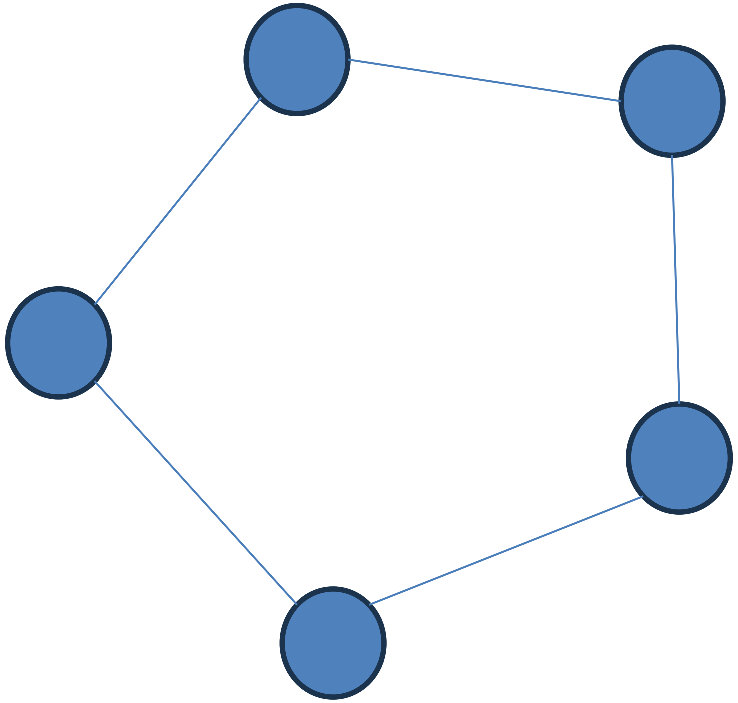


P4

Los Grafos camino: los vértices se conectan uno detrás de otro formando un camino sin conectar los extremos. Se usa la letra P de Path (Camino).



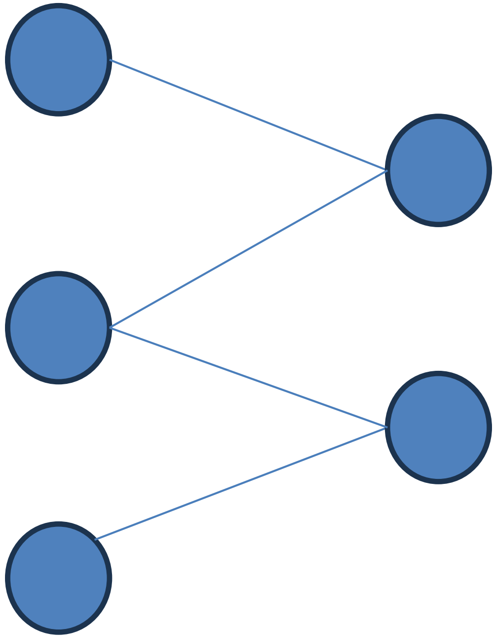
C4



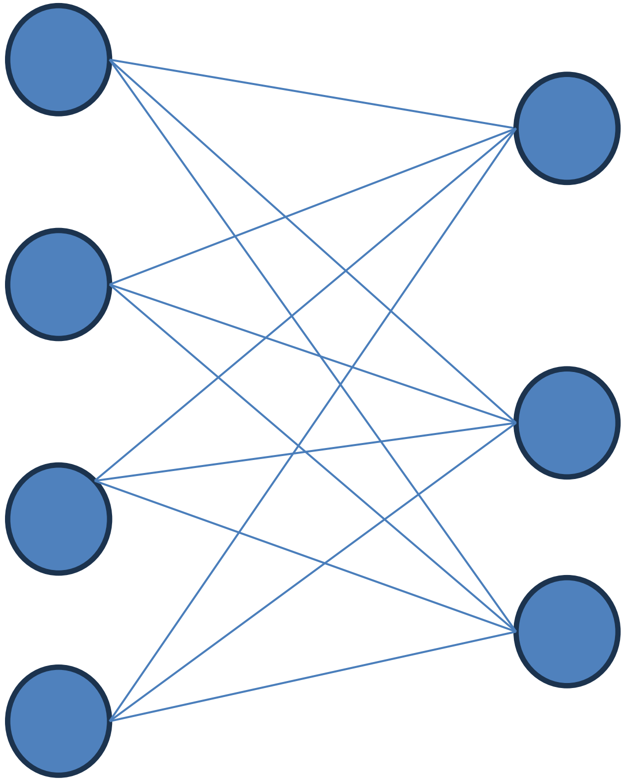
C5

Grafos ciclo: Cada vértice se conecta con otros dos formando un ciclo. Se usa la letra C de Cycle.

Grafos Bipartito y Bipartito Completo

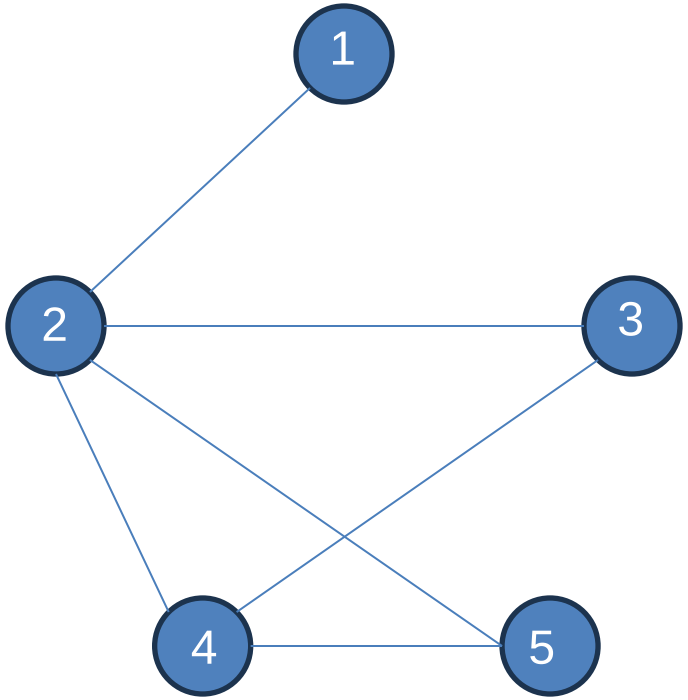


K3,2



K4,3

La formal, nombrando los vértices y cada una de las aristas con llaves.



V = {1,2,3,4,5}

A = { {1,2}, {2,3} , {2,4} , {2,5} , {3,4} , {4,5} }

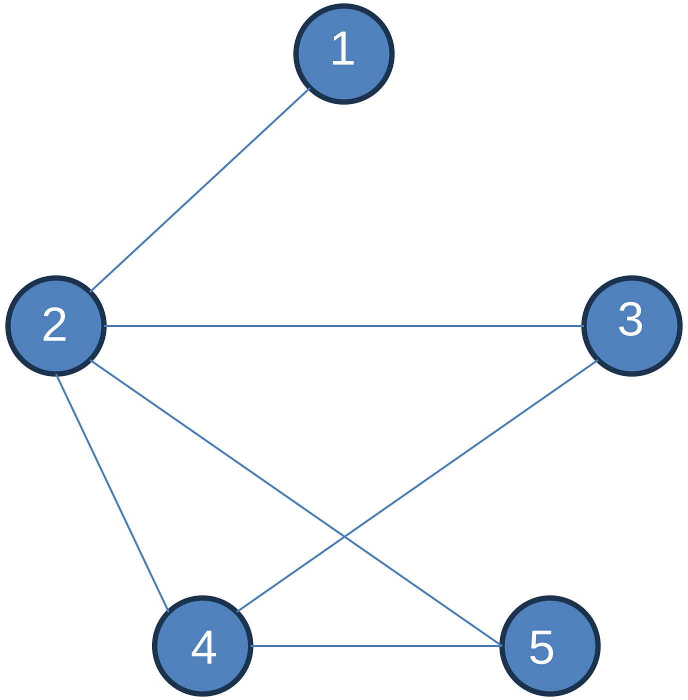
La definición formal de un grafo es la composición de dos conjuntos finitos: el conjunto de vértices y el conjunto de aristas.

Definición formal

- Un grafo G está formado por dos conjuntos finitos: N y A.
- N es el conjunto de vértices o nodos.
- A es el conjunto de aristas o arcos, que son las conexiones que relacionan los nodos.

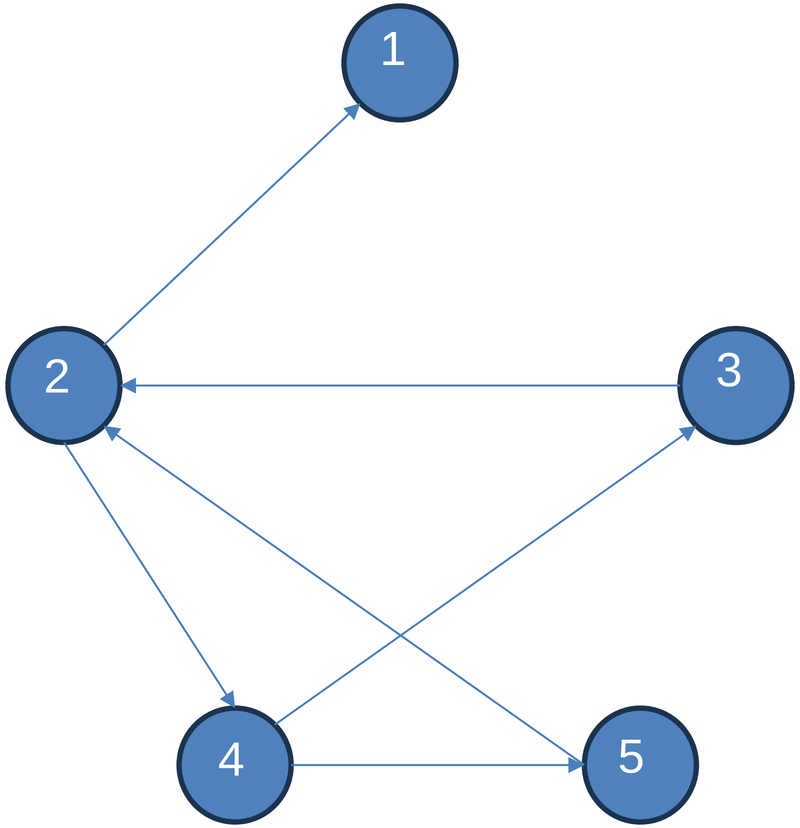
Lista de Adyacencias, una lista con las adyacencias de cada vértice.

[[2], [1,3,4,5], [2,4], [2,3,5], [2,4]]



MATRIZ DE ADYACENCIAS

	1	2	3	4	5
1	0	1	0	0	0
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	0	1	0	1	0



Los grafos pueden ser dirigidos, en vez de aristas tendríamos flechas indicando la dirección, conocidas como arcos y esto afectaría en la forma de representarlo ya que ahora no es lo mismo {1,2} que {2,1}, para ello usamos un par ordenado, usando paréntesis en vez de llaves.

FORMAL

V = {1,2,3,4,5}

A = { (1,2), (2,3), (2,4), (2,5), (3,4), (4,5) }

MATRIZ DE ADYACENCIAS

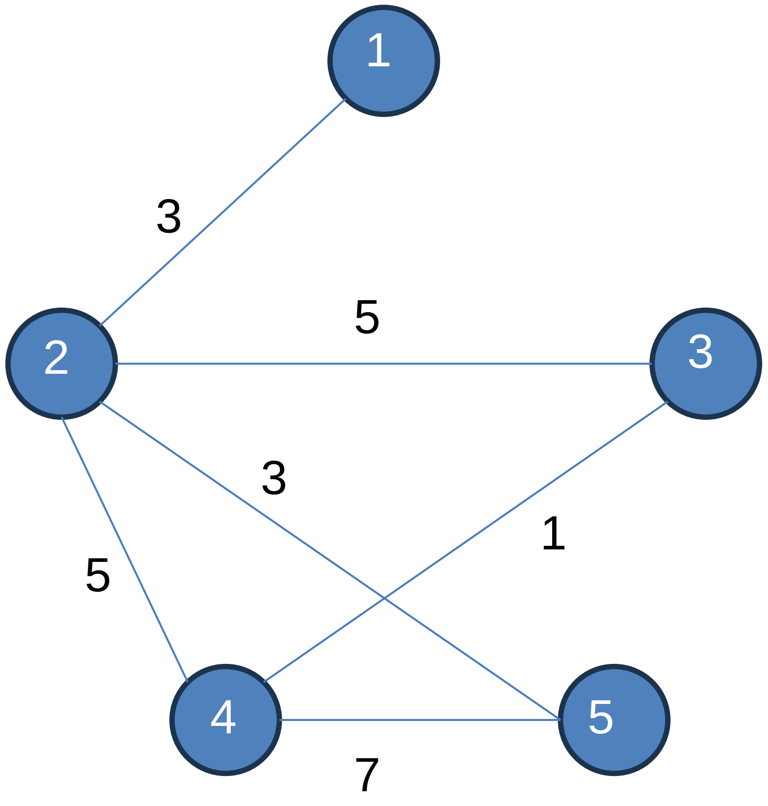
	1	2	3	4	5
1	0	0	0	0	0
2	1	0	0	1	0
3	0	1	0	0	0
4	0	0	1	0	1
5	0	1	0	0	0

Un grafo puede ser Ponderado, esto significa que podemos asociar un número a cada arista al que llamaremos peso y se suele usar la **w** de weight.

V = {1,2,3,4,5}

A = { {1,2}, {2,3} , {2,4} , {2,5} , {3,4} , {4,5} }

W = { {3} , {5}, {5}, {3}, {1}, {7} }



MATRIZ DE ADYACENCIAS

	1	2	3	4	5
1	0	3	0	0	0
2	3	0	5	5	3
3	0	5	0	1	0
4	0	5	1	0	7
5	0	3	0	7	0

Tipos de Grafos

- **Grafo dirigido**: Las aristas tienen una dirección.
- **Grafo no dirigido**: Las aristas no tienen dirección.
- **Grafo ponderado**: Las aristas tienen un peso o costo asociado.
- **Grafo no ponderado**: Las aristas no tienen peso.

¿Qué es un grafo dirigido?

Un **grafo dirigido** (también llamado **dígrafo**) es un tipo de grafo en el cual **cada arista tiene una dirección**. Es decir, las conexiones entre los vértices no son de ida y vuelta necesariamente.

G=(V,A)

donde:

- V es un conjunto de **vértices** o **nodos**.
- A es un conjunto de **aristas dirigidas** (también llamadas **arcos**), donde cada arco es un **par ordenado** (u,v), que indica una conexión **desde** el vértice u **hacia** el vértice v.

Ejemplo sencillo

Imagina que tienes tres vértices: A, B y C, y las siguientes conexiones:

- De A a B
- De B a C
- De A a C

Esto **no** implica que puedas ir de B a A o de C a A automáticamente, porque la dirección importa.

Grado de Entrada (In-degree)

Número de aristas que **entran** a un vértice.

Ejemplo:

- In-degree de C = 2 (viene desde A y B).

Grado de Salida (Out-degree)

Número de aristas que **salen** de un vértice.

Ejemplo:

- Out-degree de A = 2 (hacia B y C).

Aplicaciones de Grafos Dirigidos

- Redes sociales: Un usuario puede seguir a otro, pero no necesariamente ser seguido de vuelta (como en Twitter).
- Mapas de carreteras: Algunas calles son de sentido único.
- Tareas dependientes: Planificación de proyectos donde ciertas tareas deben completarse antes que otras (Diagramas de precedencia).

Ejercicio Rápido

Dado este conjunto de aristas:

- (A, B)
- (B, A)
- (B, C)
- (C, A)

1. ¿Cuál es el grado de entrada de A?
2. ¿Cuál es el grado de salida de B?

Respuesta:

1. Entrada a A: Desde B y desde C → **2**
2. Salida de B: Hacia A y hacia C → **2**

Matriz de Adyacencias

	A	B	C
A	0	1	0
B	1	0	1
C	1	0	0

Problema 1: Ruta óptima en un grafo dirigido

Supongamos el siguiente grafo:

- A → B (peso 2)
- A → C (peso 5)
- B → C (peso 1)
- B → D (peso 2)
- C → D (peso 3)

Pregunta: ¿Cuál es el camino más corto de A a D?

Matriz de Adyacencias

	A	B	C	D
A	0	2	5	0
B	0	0	1	2
C	0	0	0	3
D	0	0	0	0

Solución

1. **Inicialización:**

   1. **Distancia a A = 0**
   2. **Distancia a todos los demás = ∞**

2. **Desde A:**

   1. **A → B = 2 (mejorar distancia a B)**
   2. **A → C = 5 (mejorar distancia a C)**

3. **Desde B (distancia actual 2):**

   1. **B → C: 2 + 1 = 3 (mejorar distancia a C, antes era 5)**
   2. **B → D: 2 + 2 = 4 (actualizar distancia a D)**

4. **Desde C (distancia actual 3):**

   1. **C → D: 3 + 3 = 6 (pero ya tenemos un camino a D de peso 4, así que no actualizamos)**

5. **Resultado final:**
6. **Camino más corto de A a D es A → B → D, con distancia total 4.**

Problema 2: Detectar ciclo en un grafo dirigido

Dado el siguiente conjunto de aristas:

- A → B
- B → C
- C → A

¿Tiene ciclo?

Solución:
Si partimos de A:

- A lleva a B
- B lleva a C
- C regresa a A

Sí, tiene un ciclo (A → B → C → A)

Algoritmo de Dijkstra (para encontrar el camino más corto)

¿Qué hace?

Busca el **camino más corto** desde un vértice origen a todos los demás vértices en un grafo con **pesos no negativos**.

Pasos del Algoritmo:

1. **Inicializar**:

    1. Asignar distancia 0 al nodo de origen.
    2. Asignar distancia infinita (∞) a todos los otros nodos.
    3. Marcar todos los nodos como no visitados.

2. **Seleccionar** el nodo no visitado con **menor distancia** actual.
3. **Actualizar** la distancia de los nodos vecinos:

    1. Si el camino a través del nodo actual es más corto, actualizar.

4. **Marcar** el nodo actual como visitado.
5. **Repetir** hasta que todos los nodos estén visitados o alcanzados.

Ejemplo:

Grafo:

- A → B (peso 2)
- A → C (peso 4)
- B → C (peso 1)
- B → D (peso 7)
- C → D (peso 3)

Encuentra el camino más corto de A a D.

Nodo	Distancia Inicial
A	0
B	∞
C	∞
D	∞

Proceso:

1.Desde A:

    1. A → B: distancia 2 (mejor que ∞)
    2. A → C: distancia 4 (mejor que ∞)

2.Nodo con menor distancia: **B (2)**.

    1. B → C: 2 + 1 = 3 (mejor que 4, actualizamos)
    2. B → D: 2 + 7 = 9

3.Nodo con menor distancia: **C (3)**.

    1. C → D: 3 + 3 = 6 (mejor que 9, actualizamos)

4.Nodo D (6): Llegamos.

Camino más corto: A → B → C → D

Costo total: 6


consultas
dudas
¿?


THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
DANKSCHEEN
TASHAKKUR ATU
YAQHANYELAY
SUKSAMA
EKHMET
BÏYAN
SHUKRIA
TINGKI
GRAZIE
MEHRBANI
PALDIES
BOLZÏN
MERCI
GOZAIMASHITA
EFCHARISTO
JUSPAXAR
KOMAPSUMNIDA
MAAKE

¡¡¡Gracias por la asistencia… Éxito!!!