Grafos - Introducción

Curso OIA UNLaM - Edición 2021

Agenda

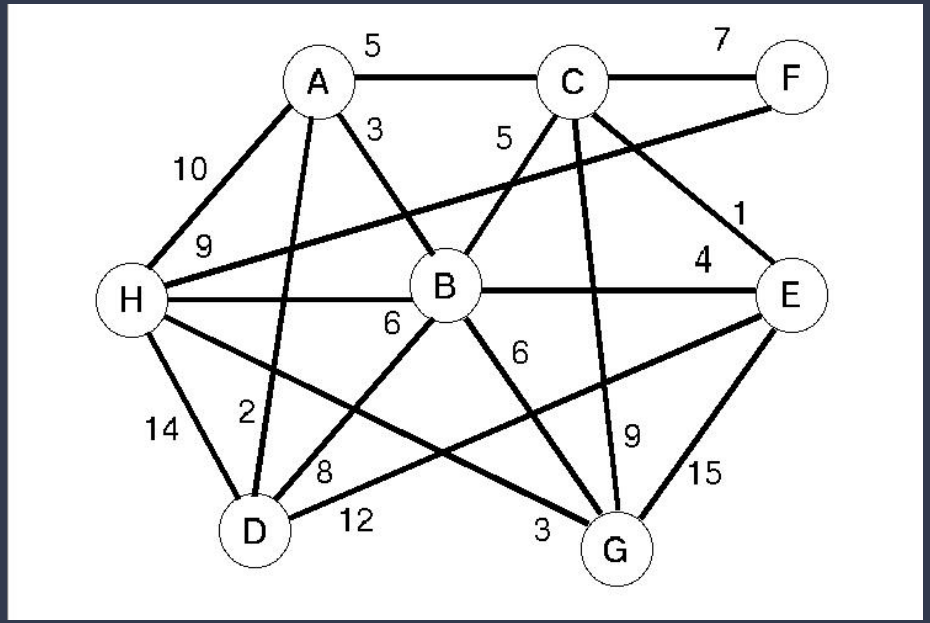
- Conceptos básicos
- Representación
- Más definiciones
- Algoritmos para recorrer grafos

Conceptos básicos

¿Qué es un grafo?

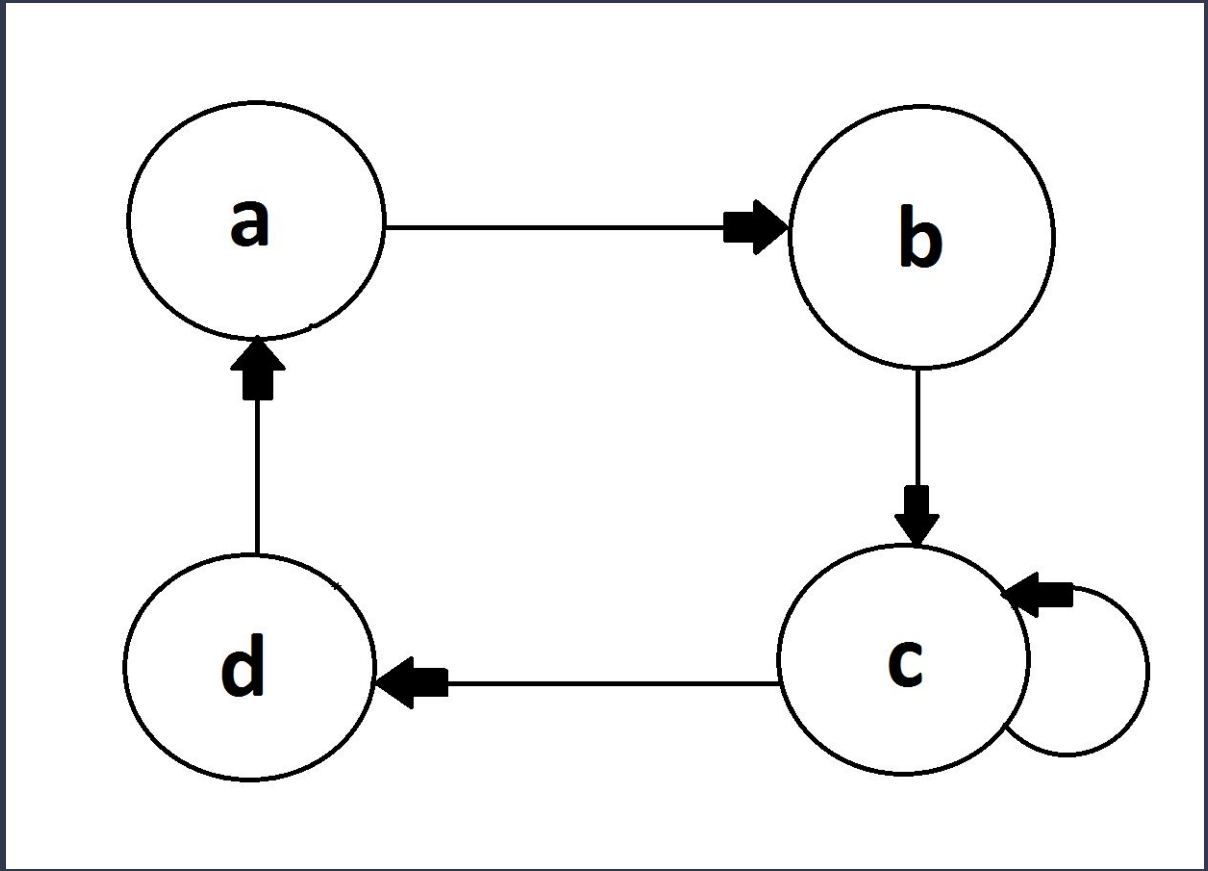
- Es un conjunto de objetos llamados nodos unidos por enlaces llamados aristas que representan una relación binaria.

- **Orden**: cantidad de nodos



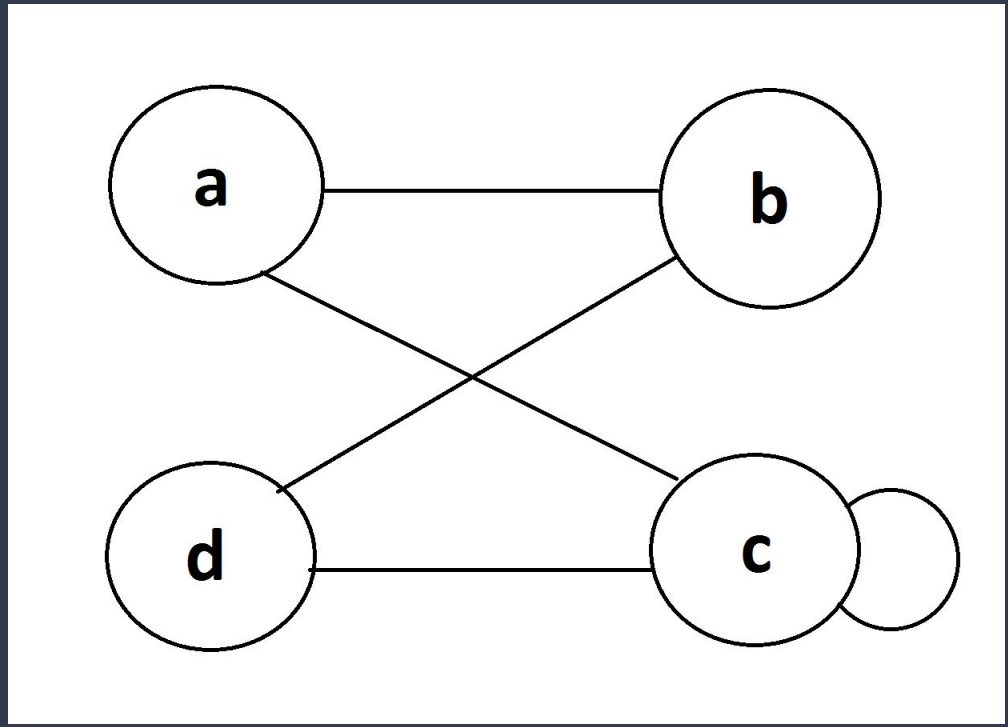
Grafo dirigido

Las aristas tienen un sentido definido desde un nodo origen hacia un nodo destino.



Grafo no dirigido

Las aristas representan relaciones simétricas y no tienen un sentido definido (conectan a los nodos en ambos sentidos)

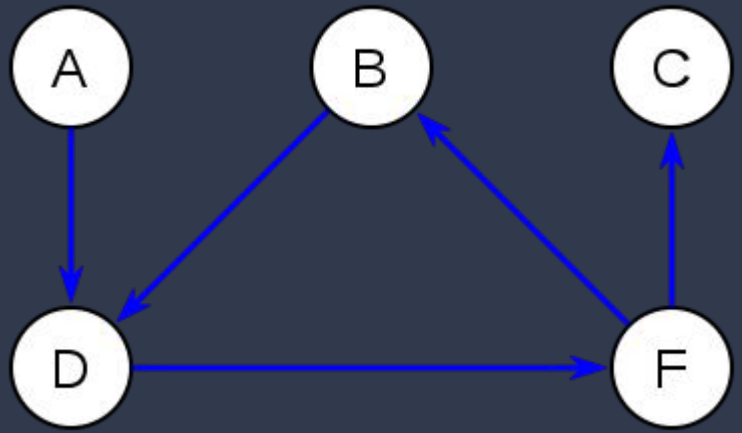


Representación

Matriz de adyacencia

- Matriz de n x n donde n es la cantidad de nodos del grafo.

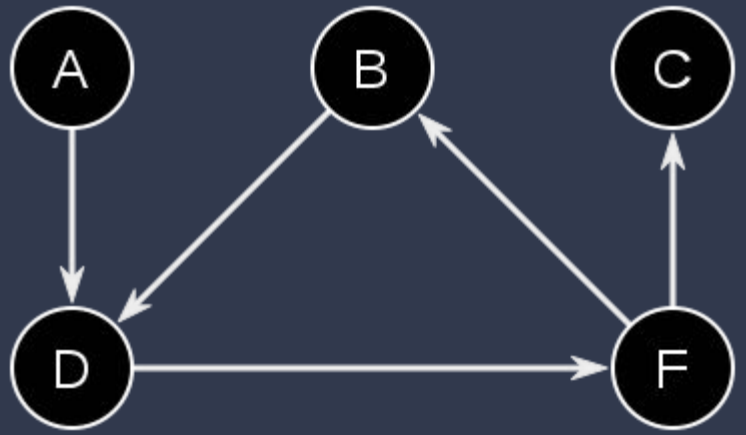
- Un 1 representa una arista en el par



	A	B	C	D	F
A	0	0	0	1	0
B	0	0	0	1	0
C	0	0	0	0	0
D	0	0	0	0	1
F	0	1	1	0	0

Lista de adyacencia

- Es un vector de vectores de enteros que en el i-ésimo vector tiene el nodo j si existe una arista entre i y j.

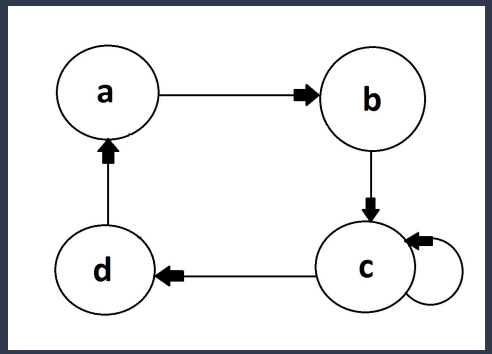


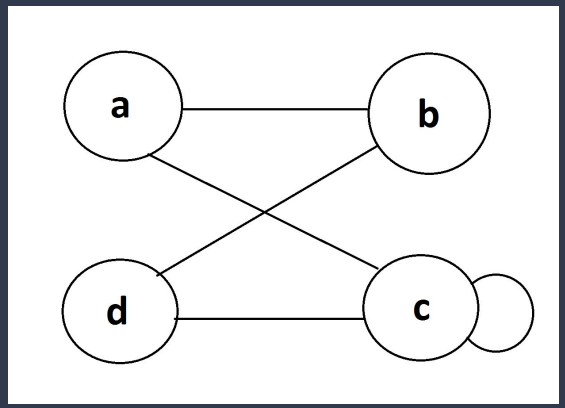
A: {D}
B: {D}
C:
D: {F}
F: {B; C}

Más definiciones

Vecino y grado

- ***a*** es vecino de ***b*** si tenemos una arista que une a los dos vértices (y ***b*** es vecino de ***a***)
- La cantidad de vecinos de ***a*** se llama grado de ***a***.





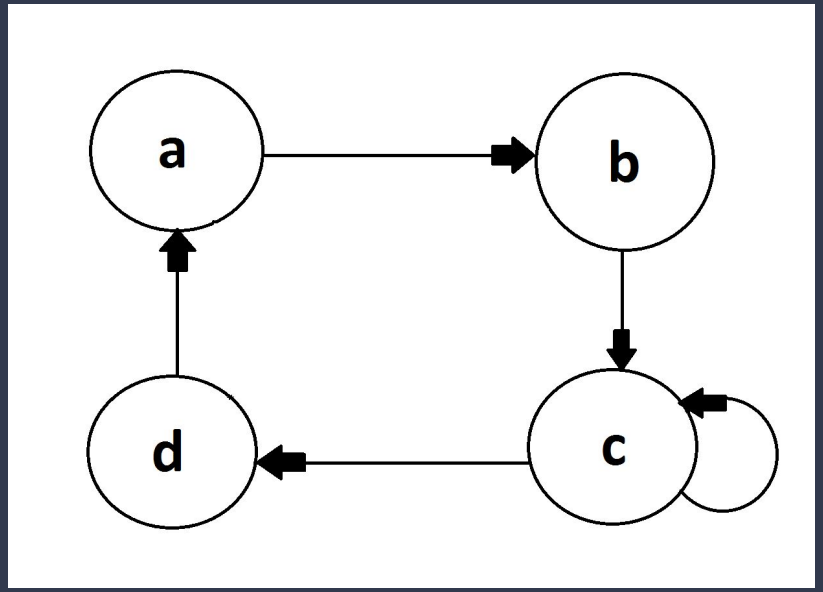
Camino y distancia

- **Camino**: Sucesión de vértices y aristas dentro de un grafo que empieza y termina en vértices.
- Dos vértices están conectados si existe un camino para llegar del uno al otro.
- La longitud de un camino es su número de aristas.
- La menor longitud es la **distancia**

Bucle y ciclo

- **Bucle**: Es una arista que conecta un vértice consigo misma

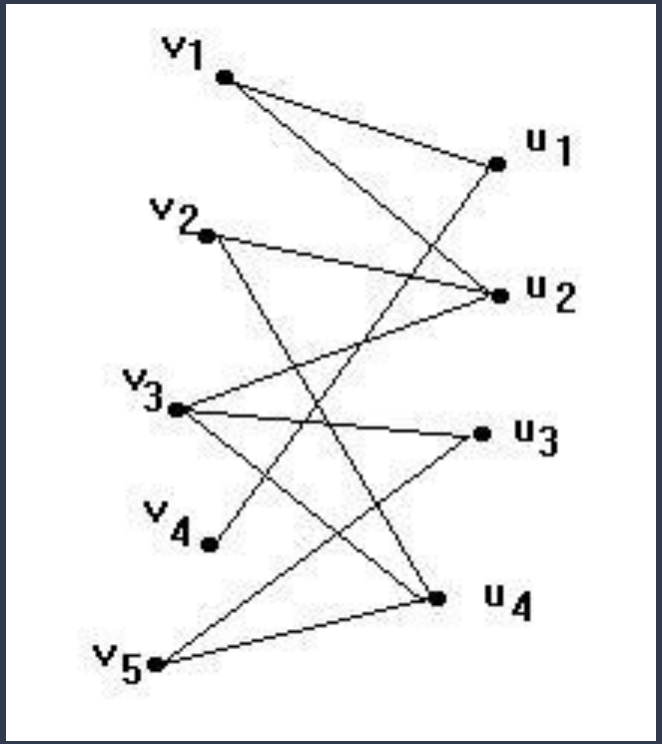
- **Ciclo**: camino donde el origen es igual al destino



No menos importante...

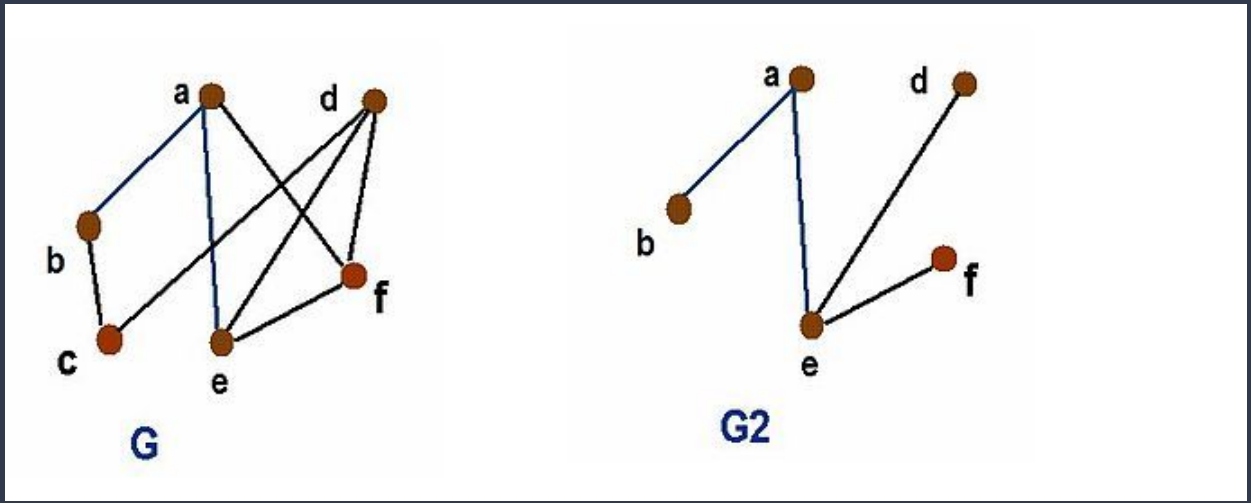
Grafo bipartito

- Es un grafo que puede ser dividido en dos partes o conjuntos
- Las aristas unen los vértices de un conjunto con los vértices del otro.



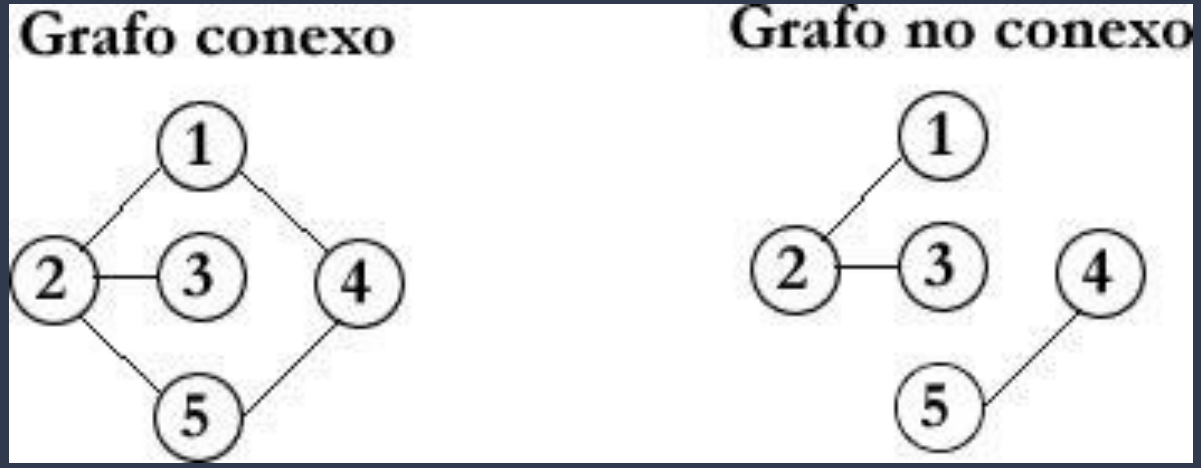
Subgrafos

Grafo con vértices y aristas que son un subconjunto de un grafo padre.



Grafo conexo

Todos los vértices están conectados por un camino



Componente conexas

Son los subgrafos conexos máximos de un grafo no dirigido

Algoritmos para recorrer grafos

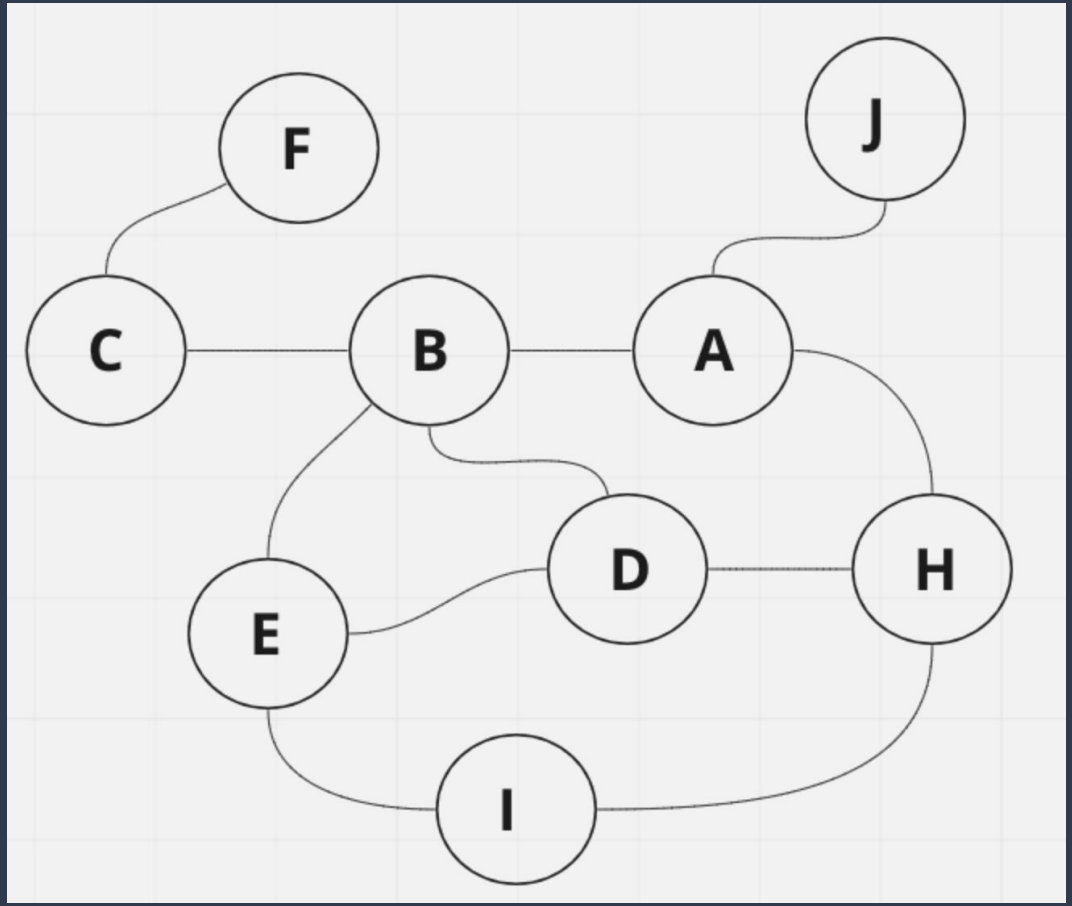
BFS

- Búsqueda en anchura: recorre el grafo y calcula la **mínima distancia** desde un nodo a cada uno de los otros.

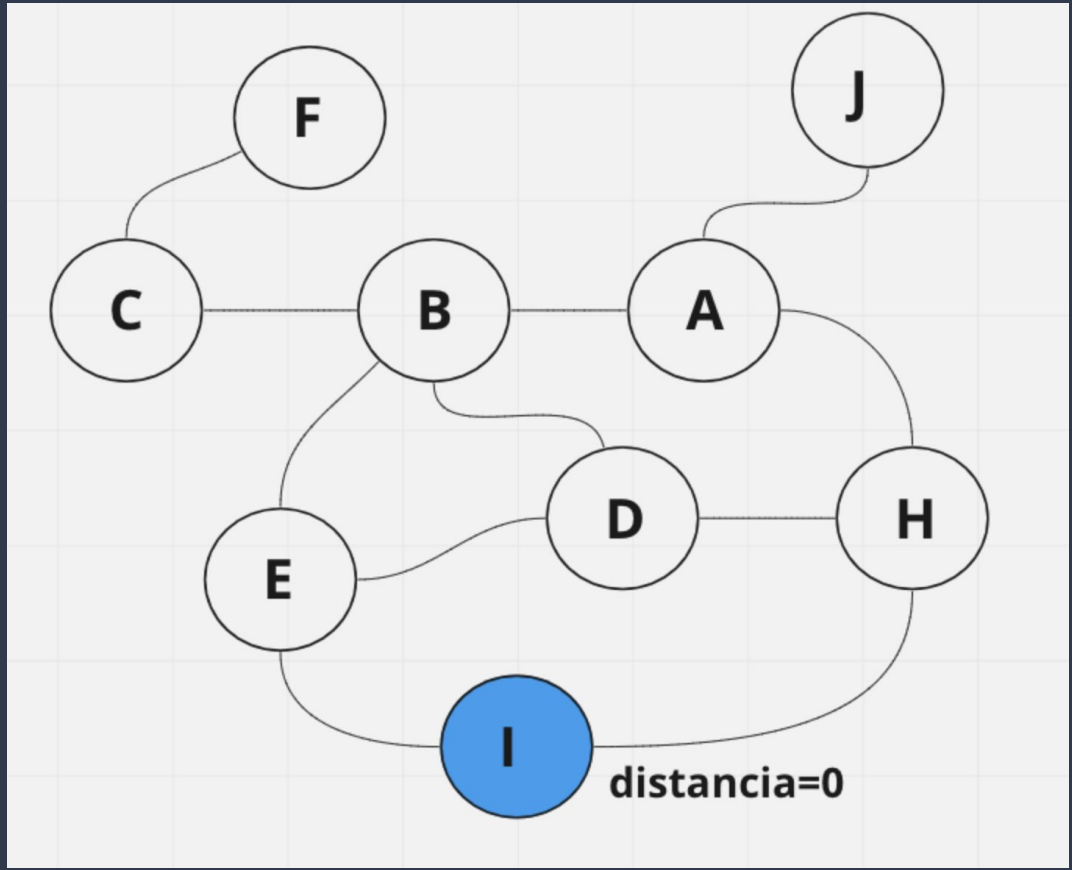
  Pasos:

  - Se comienza en un nodo y se exploran sus vecinos
  - Para cada uno de los vecinos (que aún no se hayan visitado) se exploran sus vecinos
  - Y así sucesivamente...

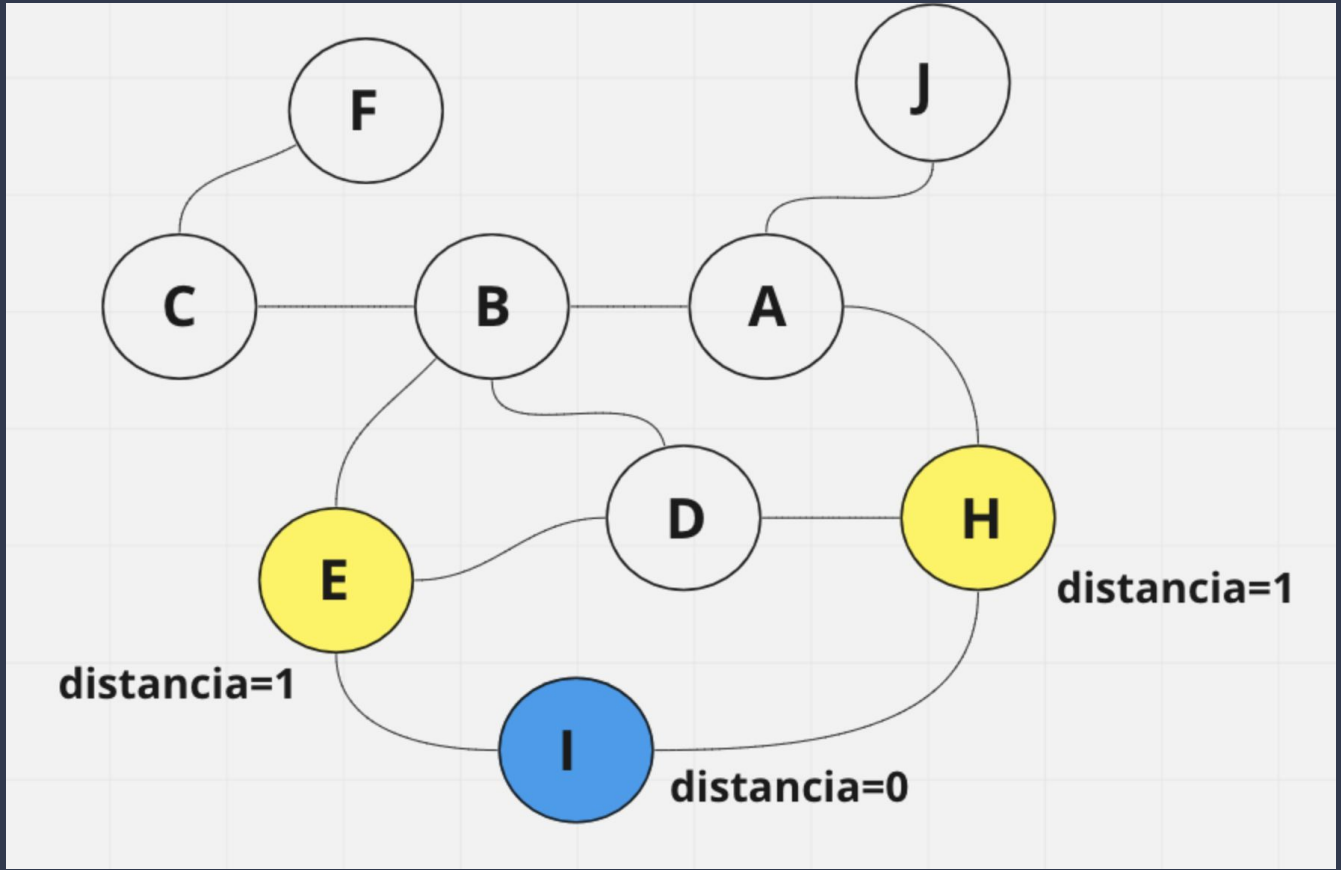
BFS



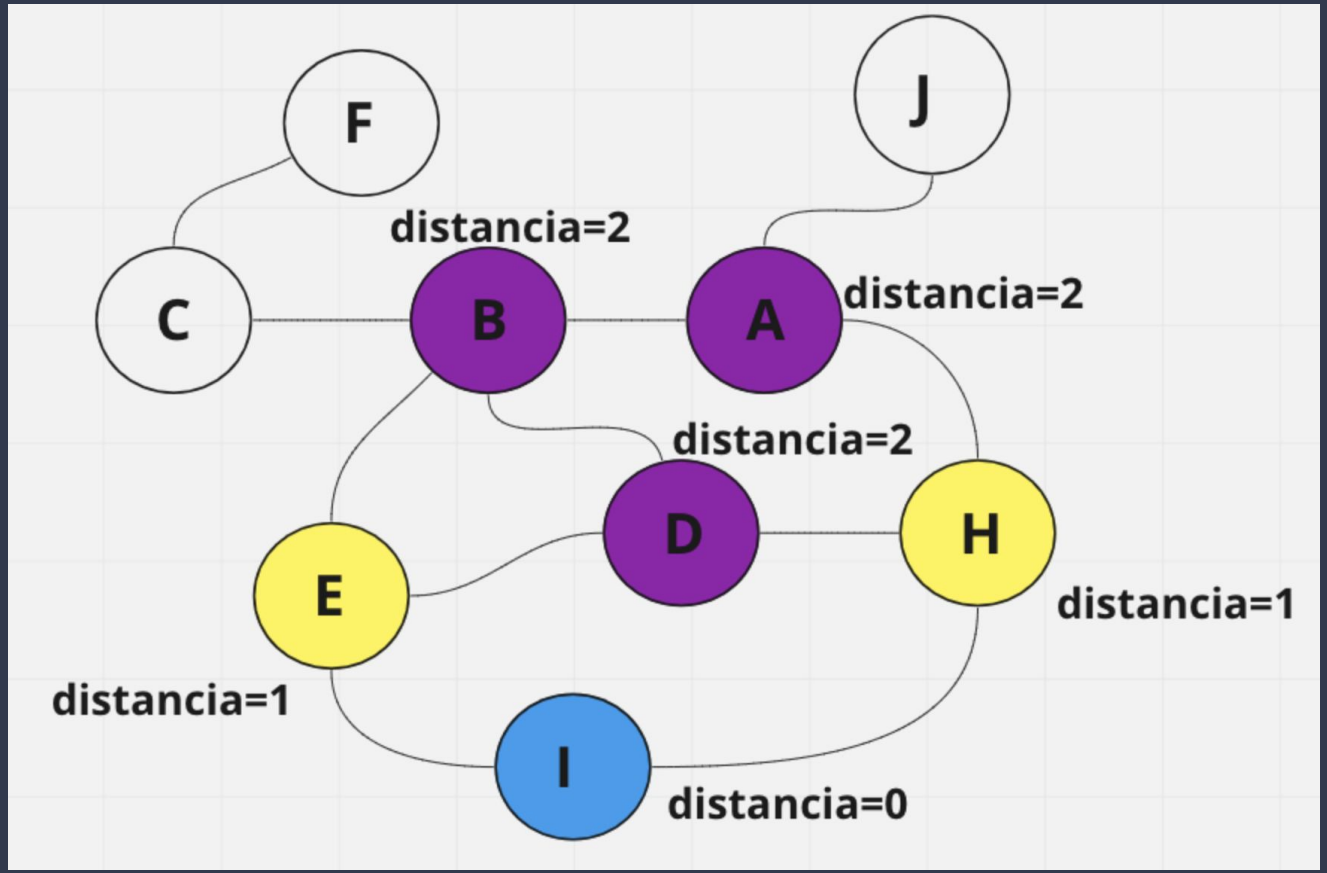
BFS



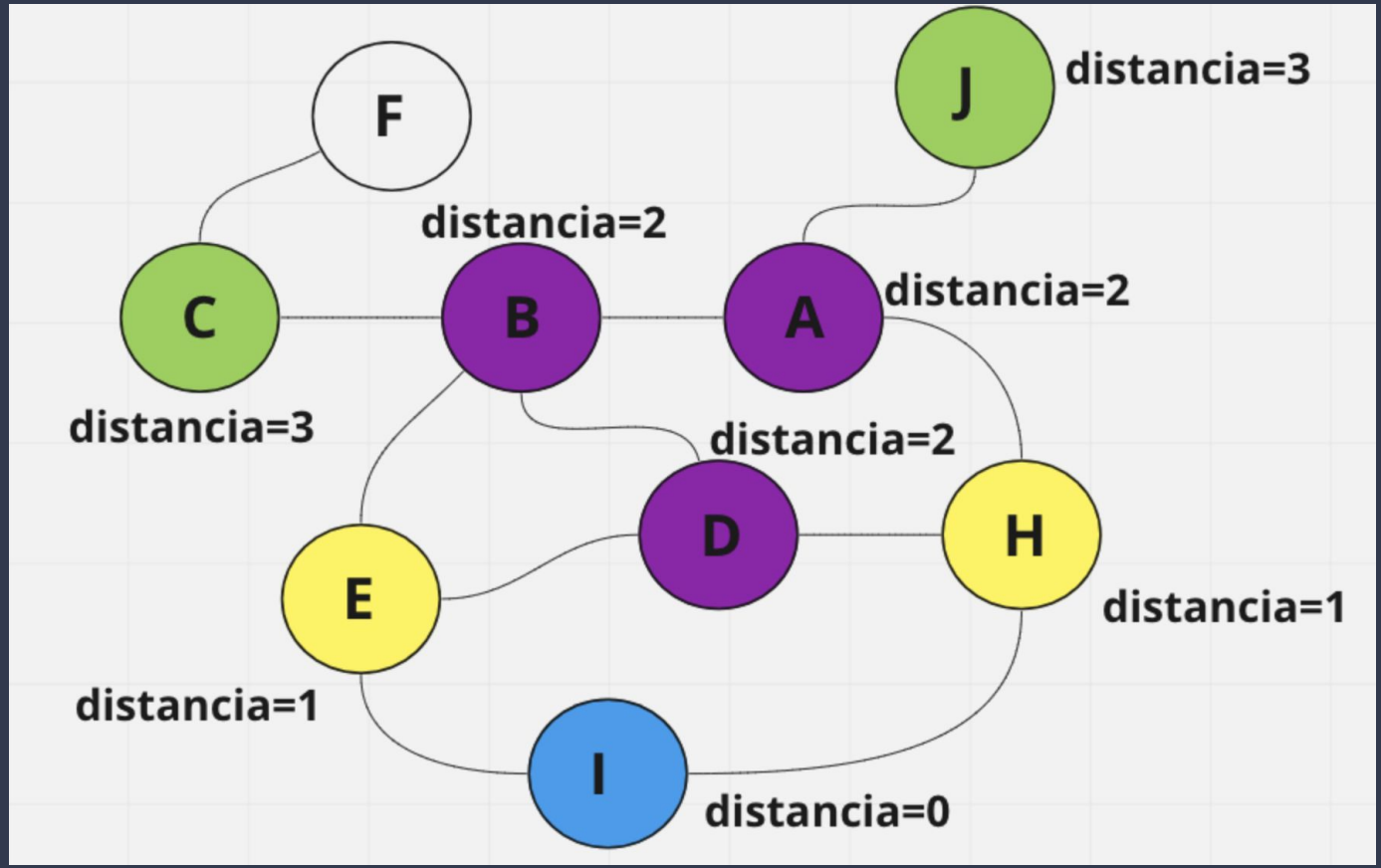
BFS



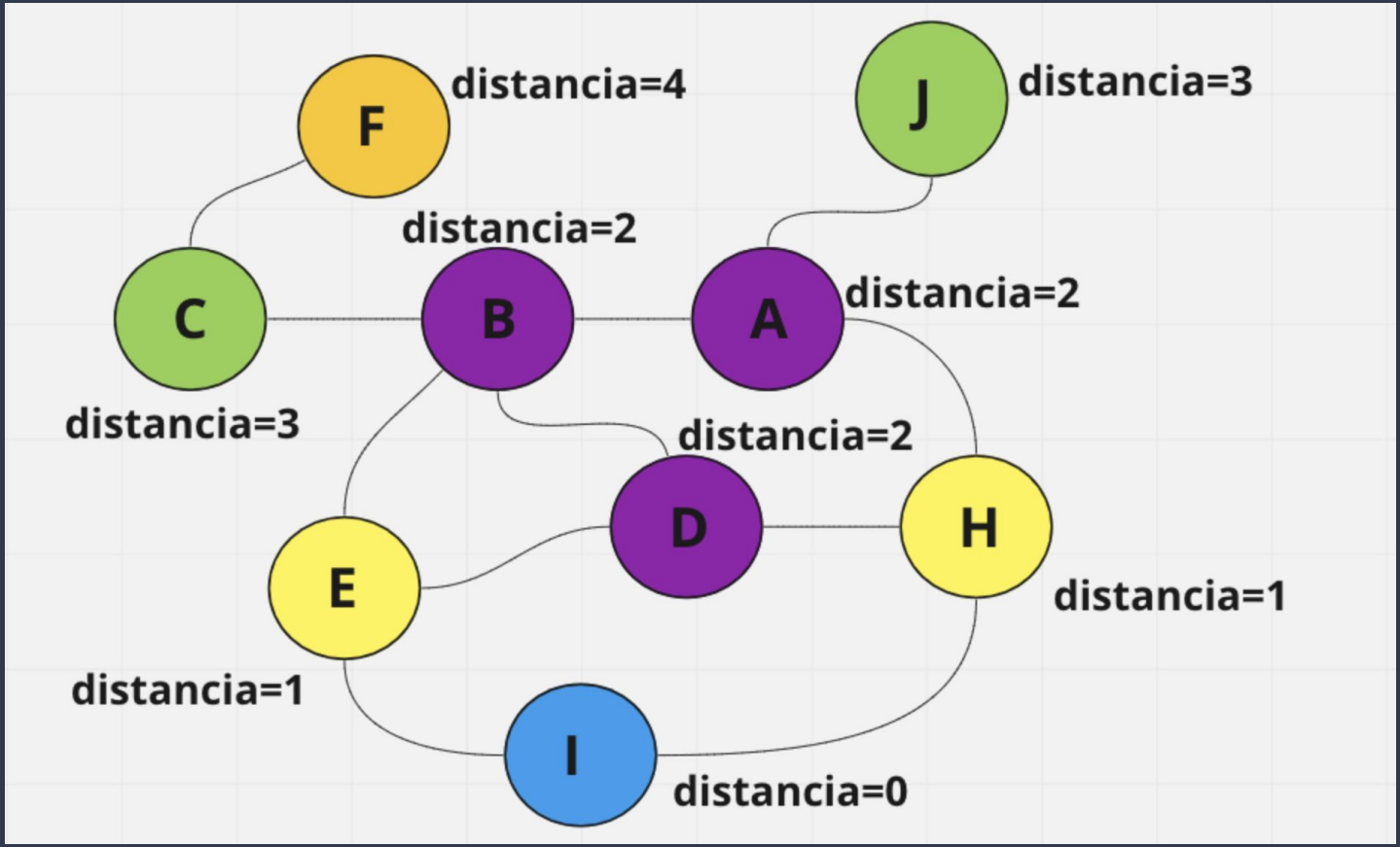
BFS


F
J
distancia=2
C
B
A
distancia=2
distancia=2
D
H
E
distancia=1
distancia=1
I
distancia=0

BFS


J
distancia=3
F
distancia=2
C
B
A
distancia=2
distancia=3
distancia=2
D
H
E
distancia=1
distancia=1
I
distancia=0

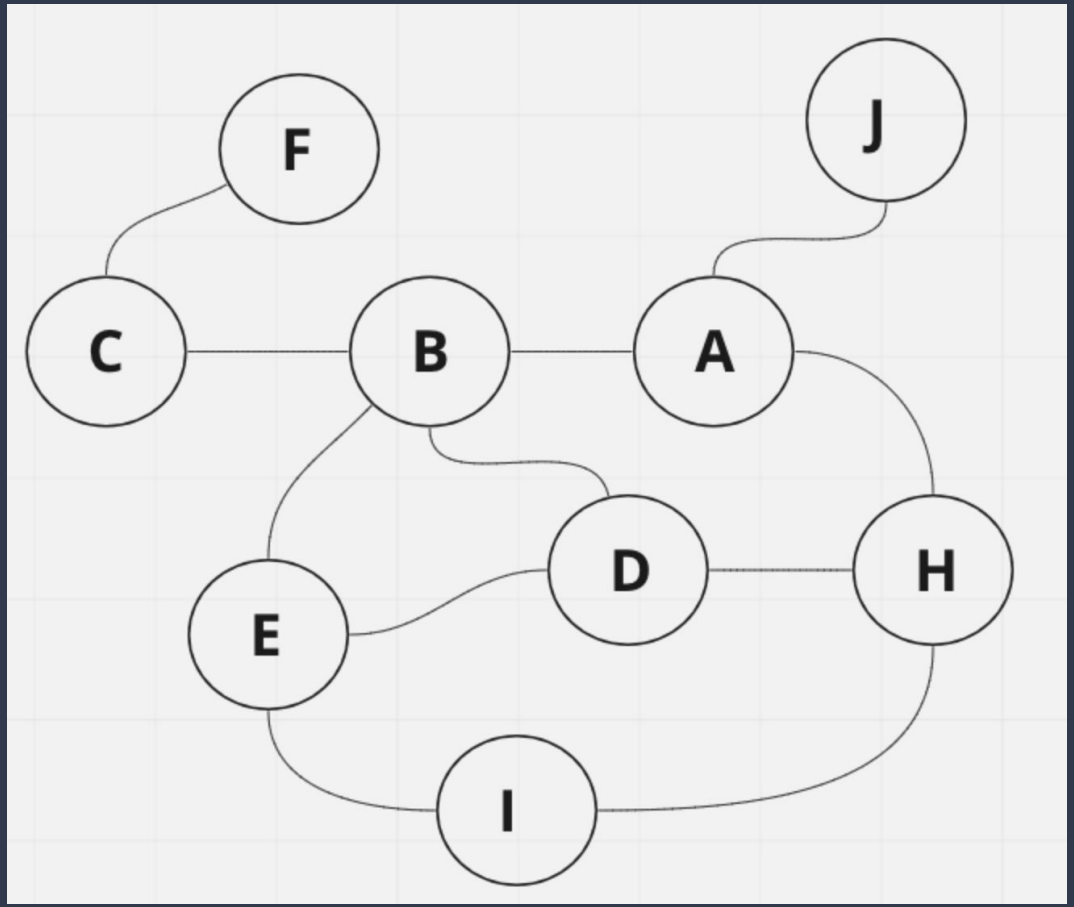
BFS


F
distancia=4
J
distancia=3
distancia=2
C
B
A
distancia=2
distancia=3
distancia=2
D
H
E
distancia=1
distancia=1
I
distancia=0

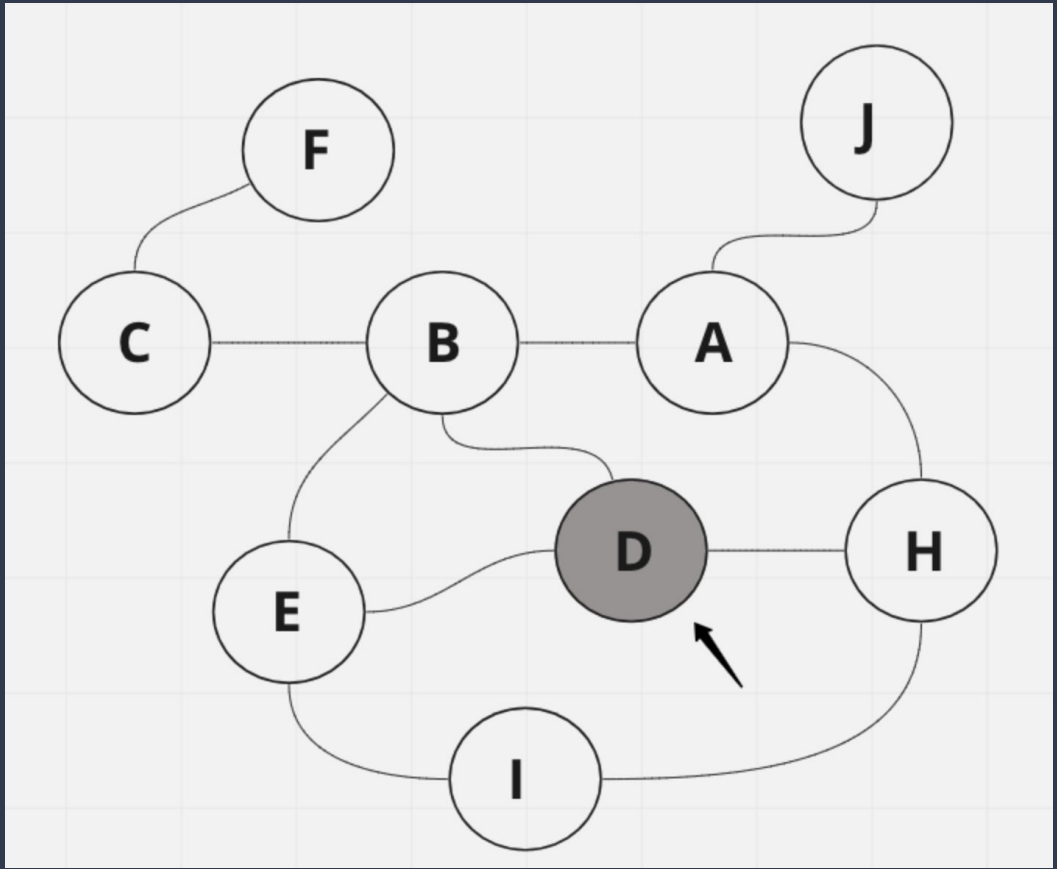
DFS

- Búsqueda en profundidad: recorre todos los nodos de un grafo de manera ordenada.
- Expande cada nodo que va localizando en un camino en concreto.
- Cuando ya no quedan vecinos por visitar, retorna y repite el proceso por cada hermano del nodo visitado.

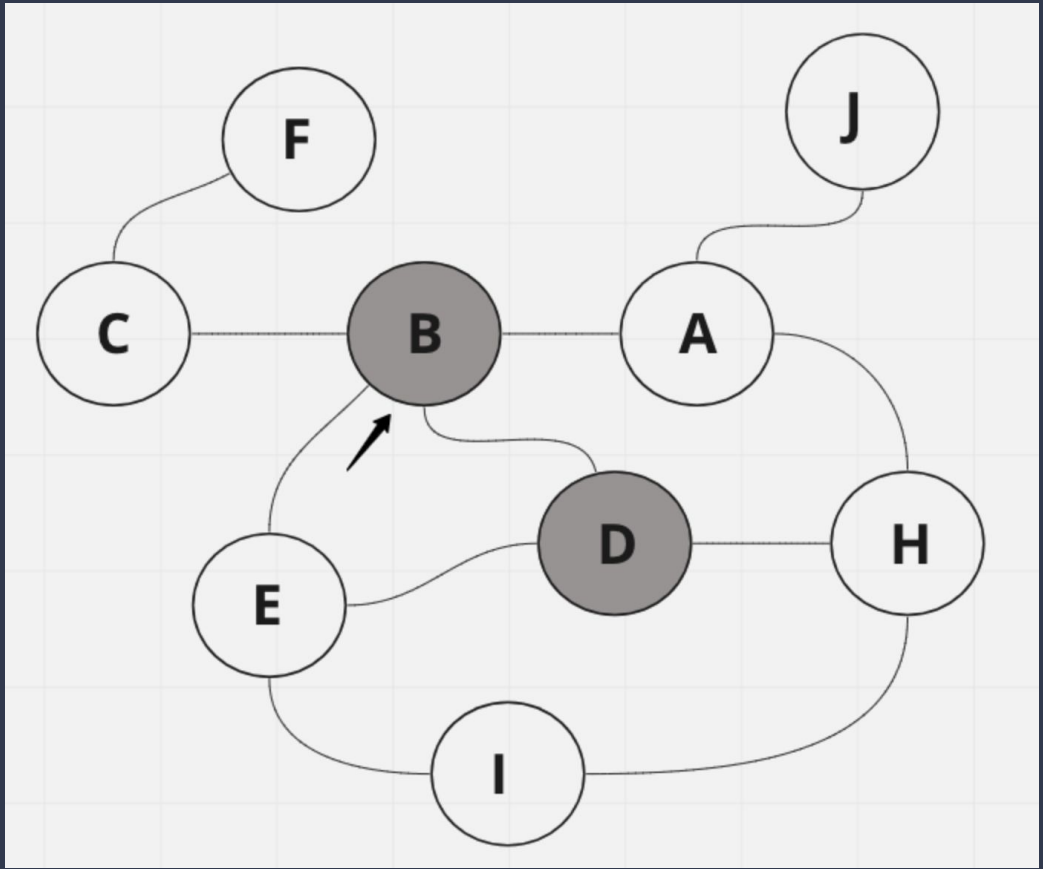
DFS



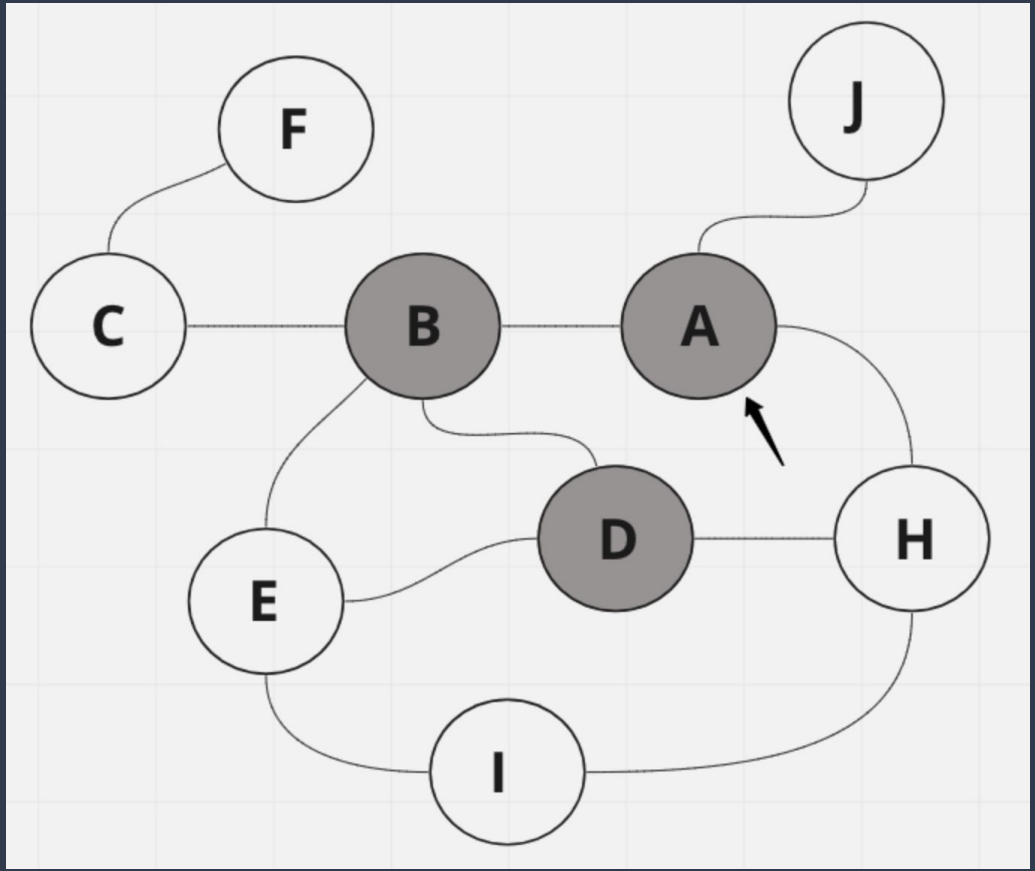
DFS



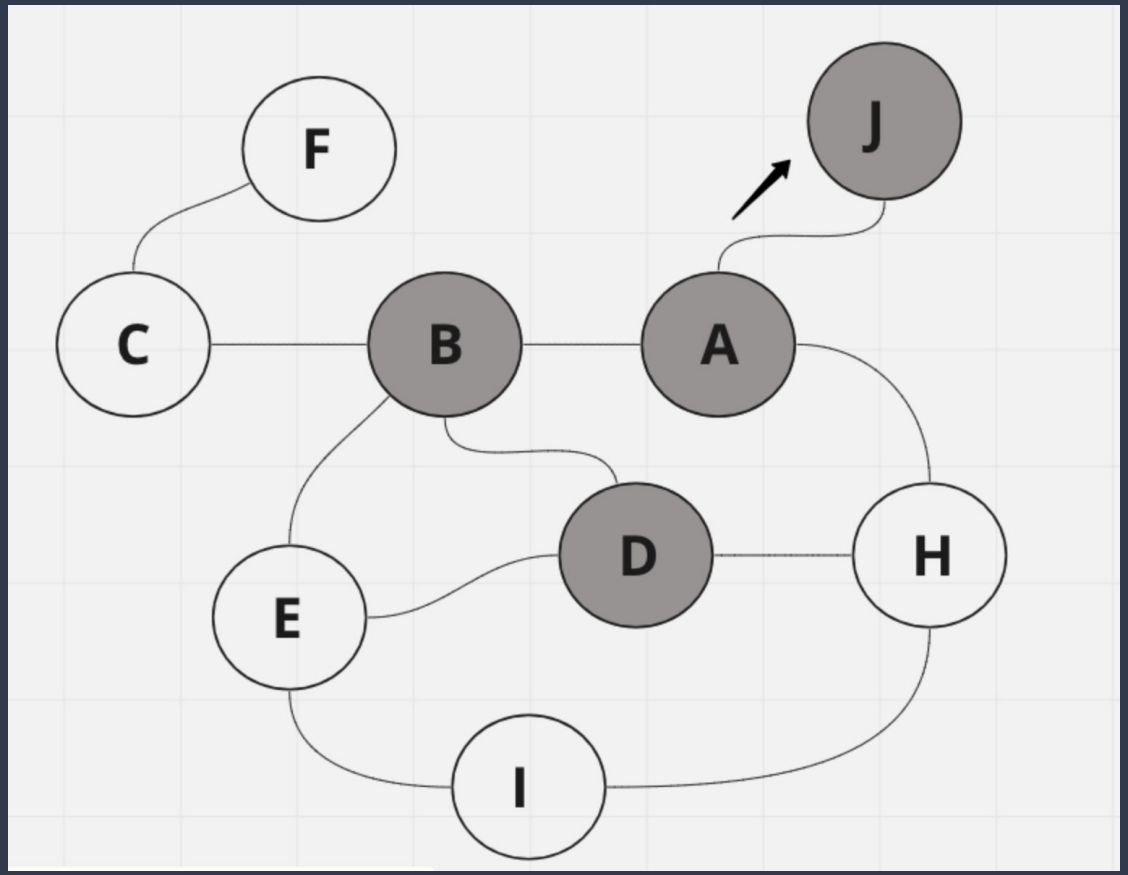
DFS



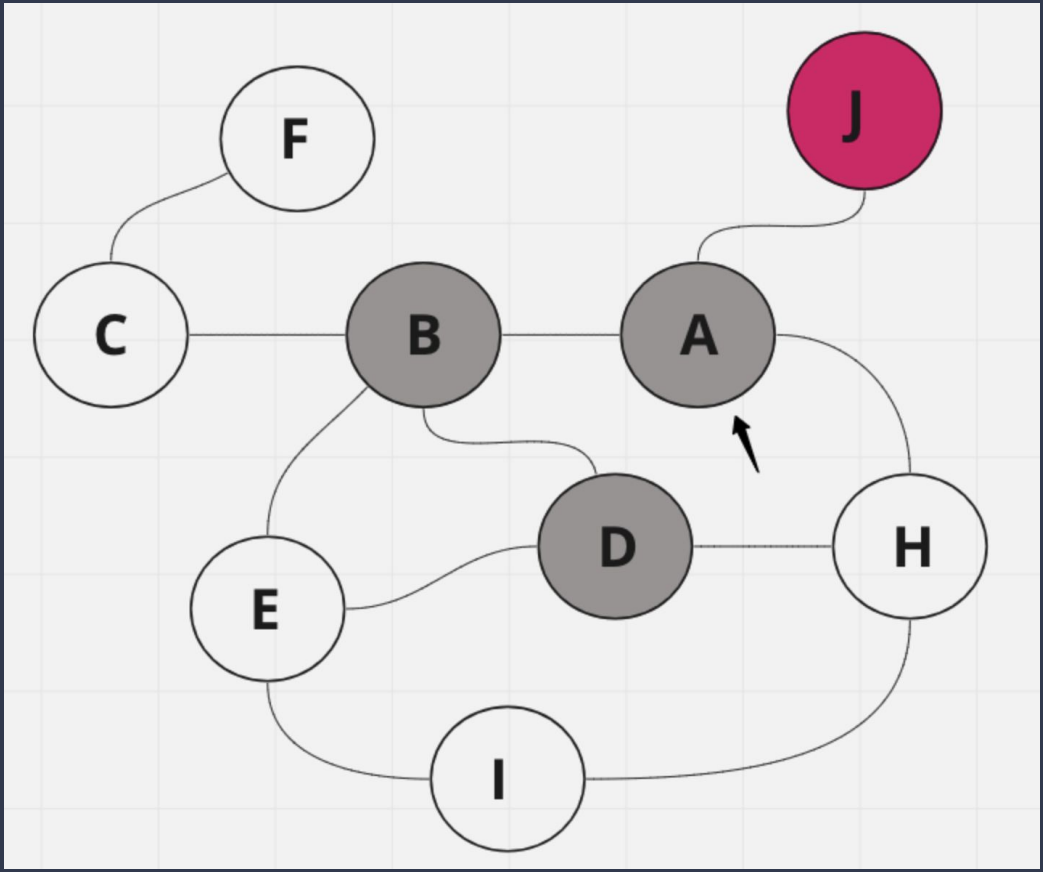
DFS



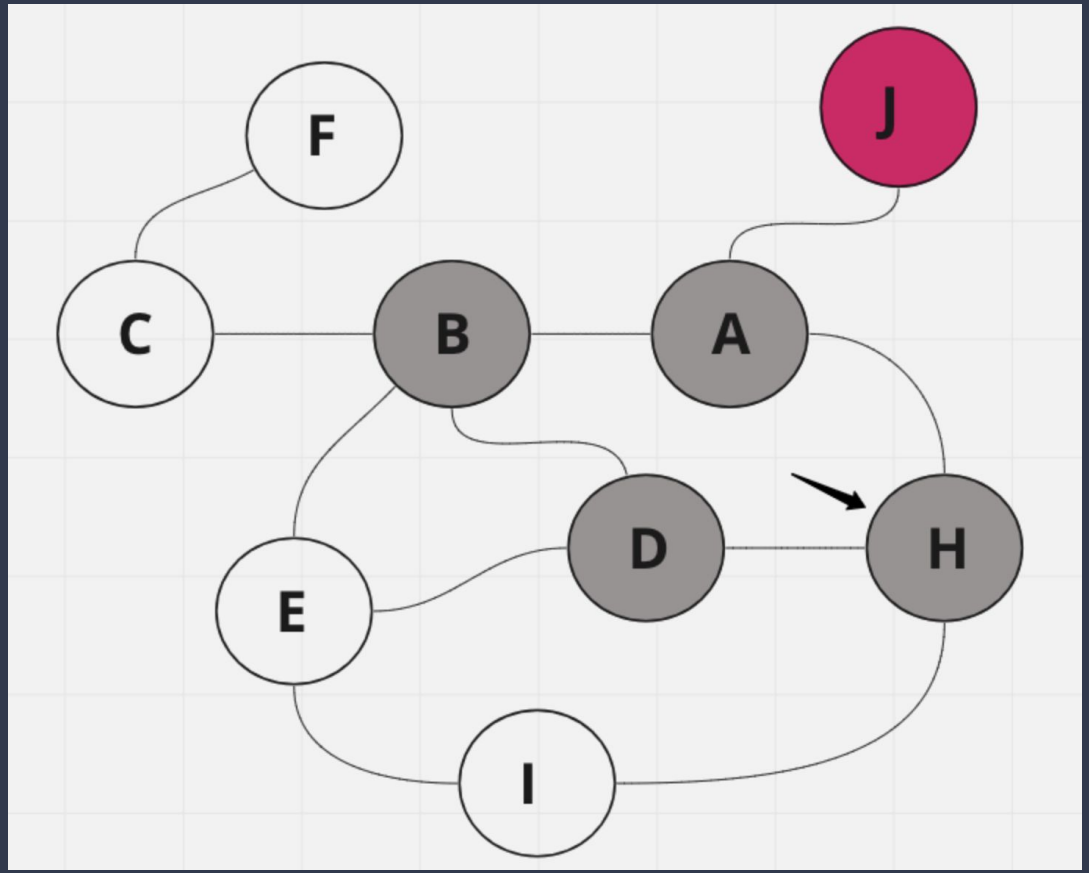
DFS



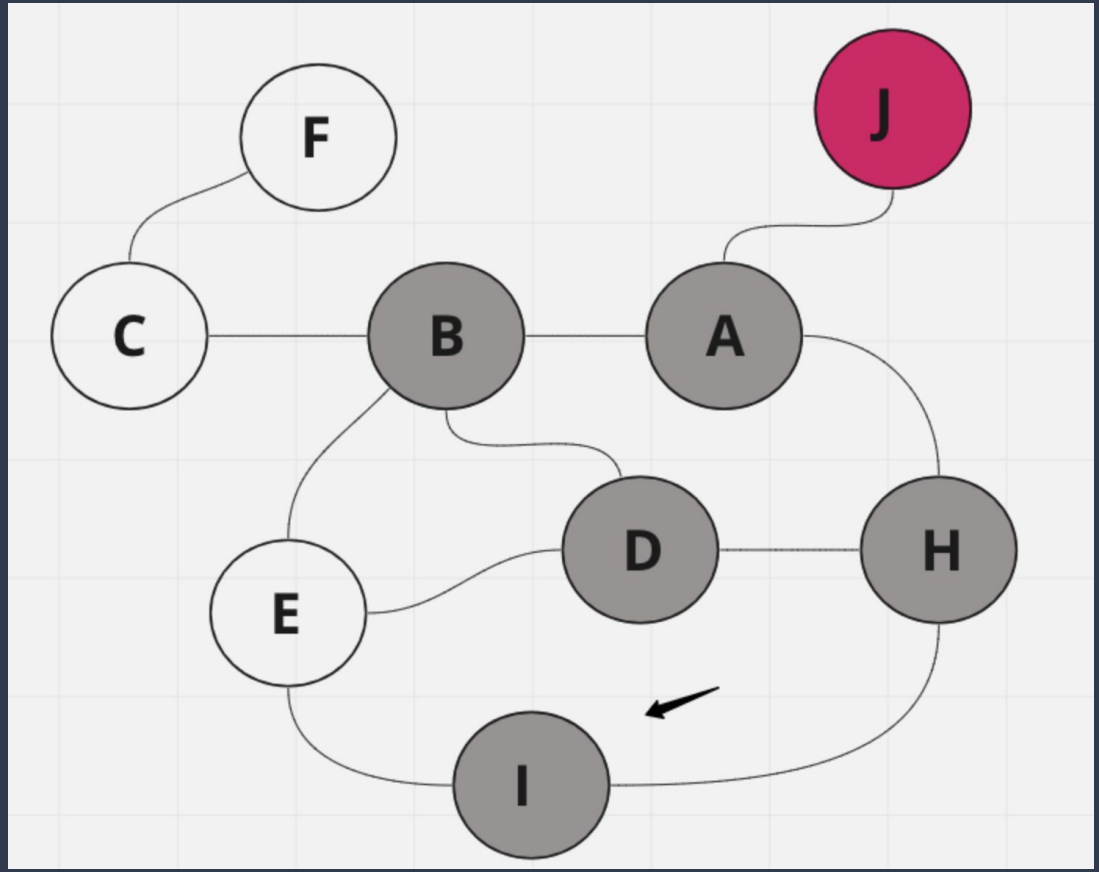
DFS



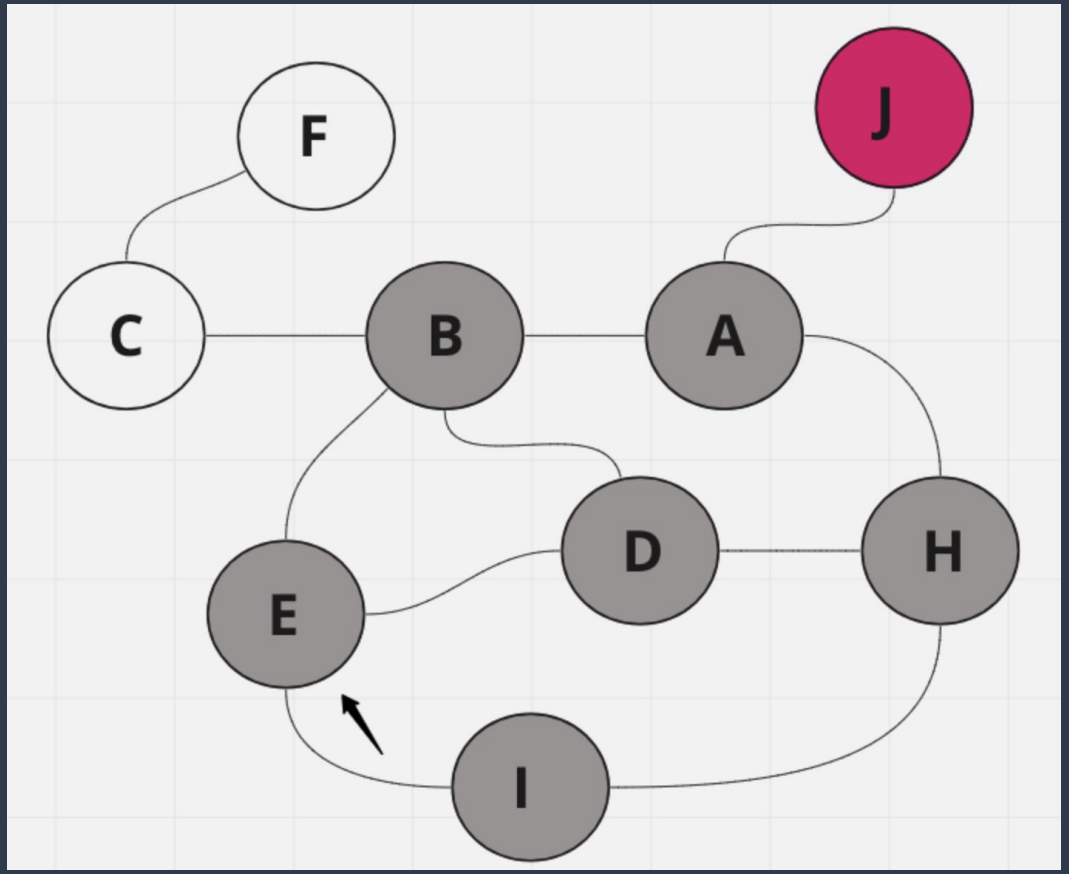
DFS



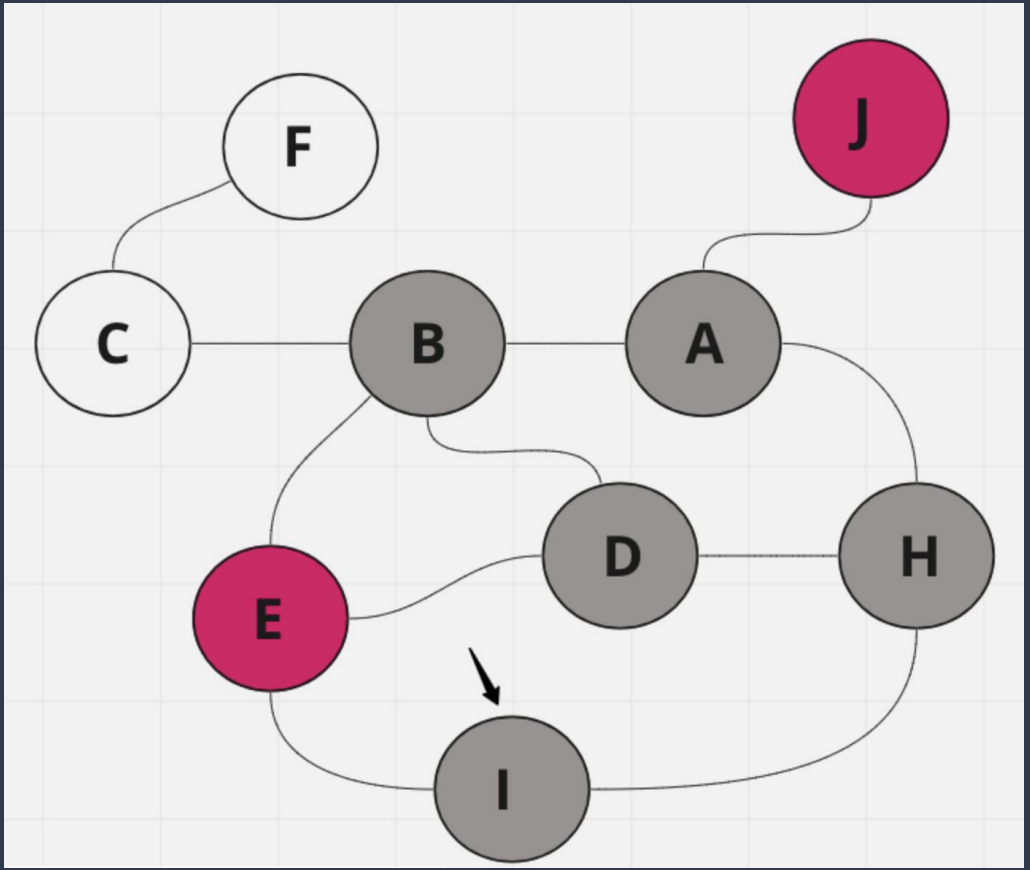
DFS



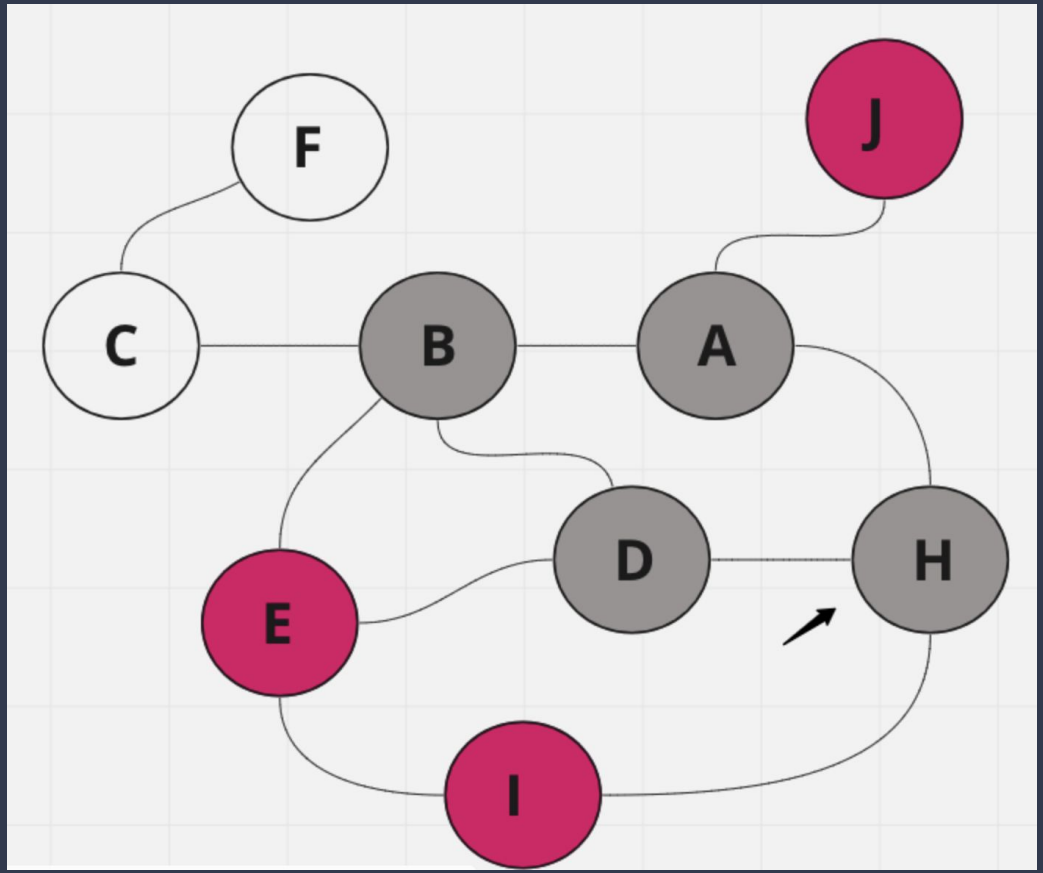
DFS



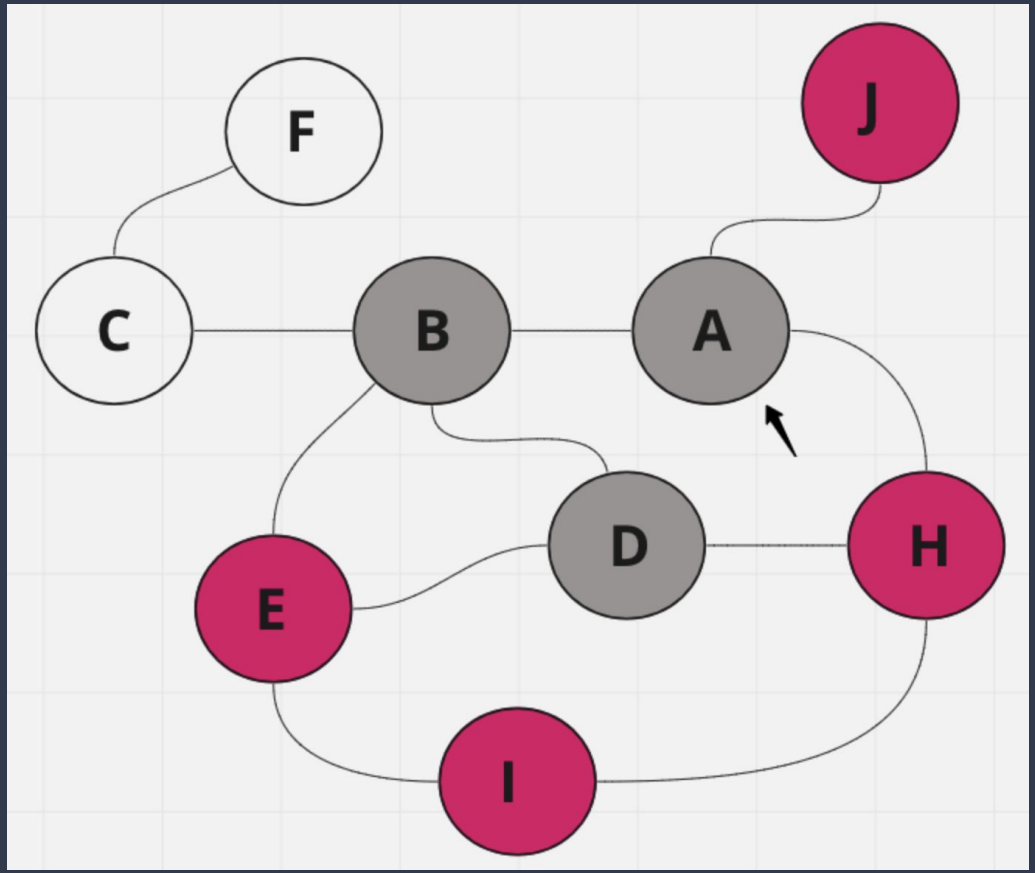
DFS



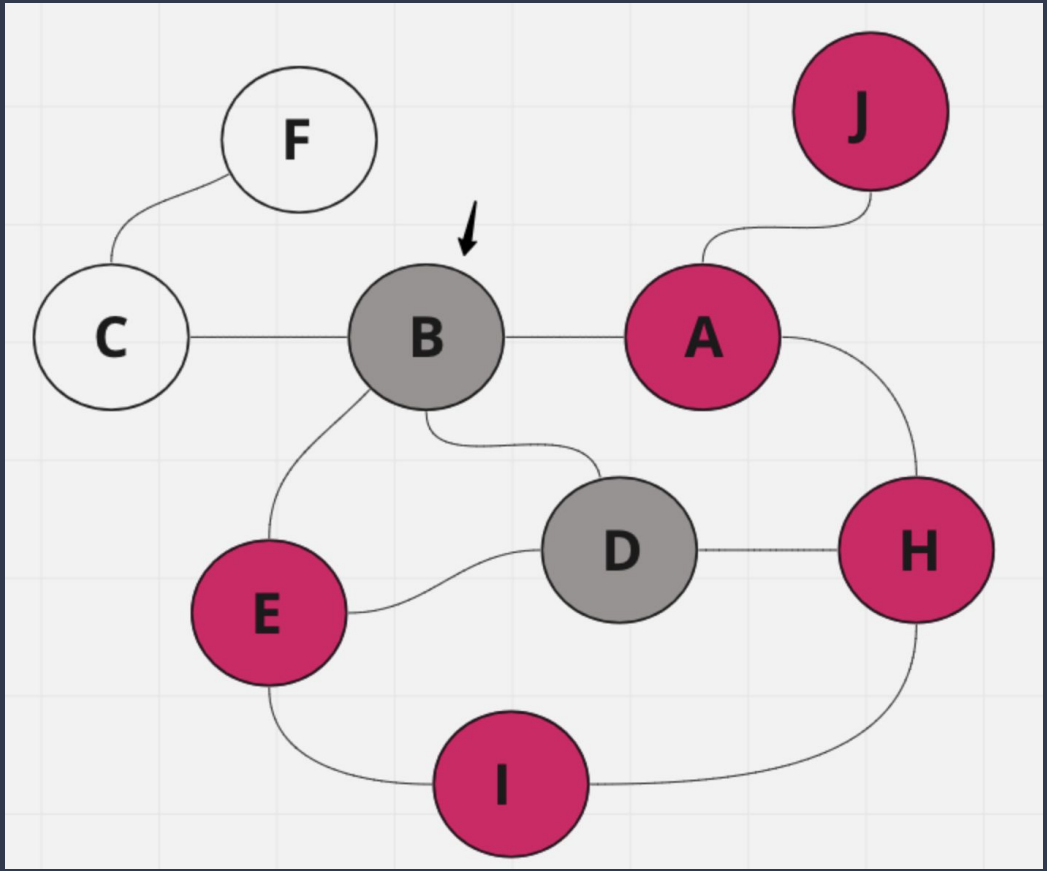
DFS



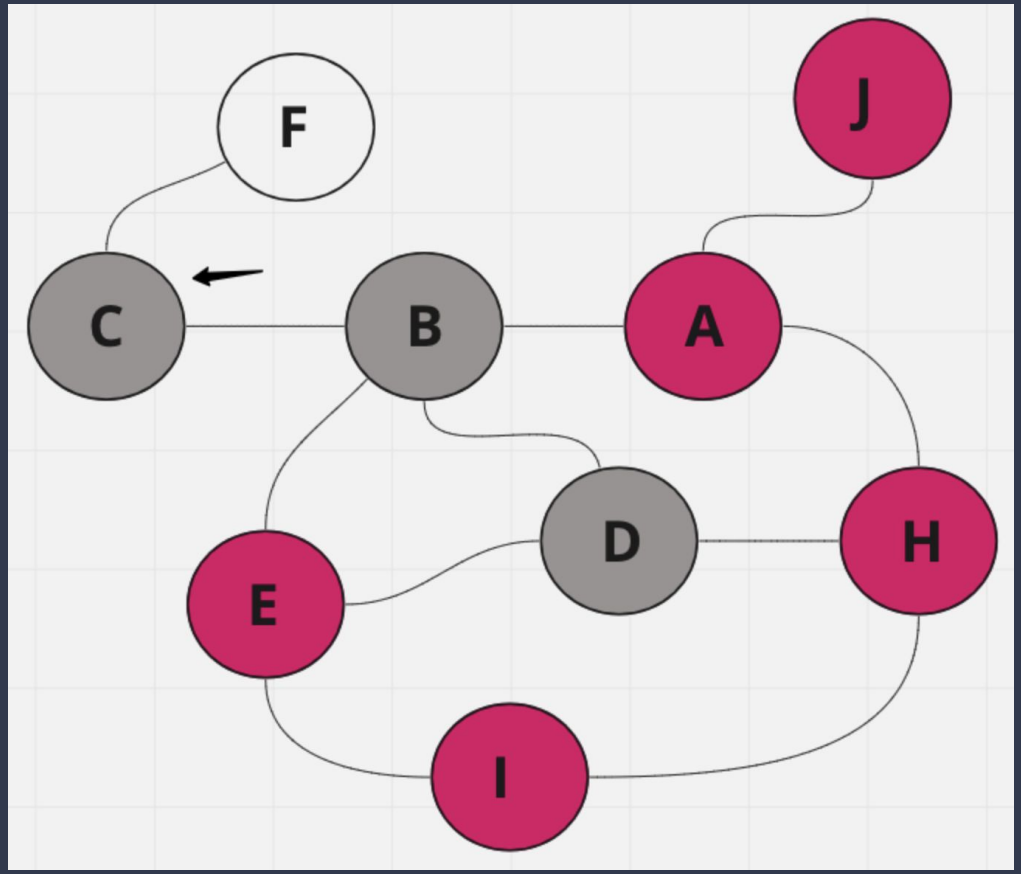
DFS



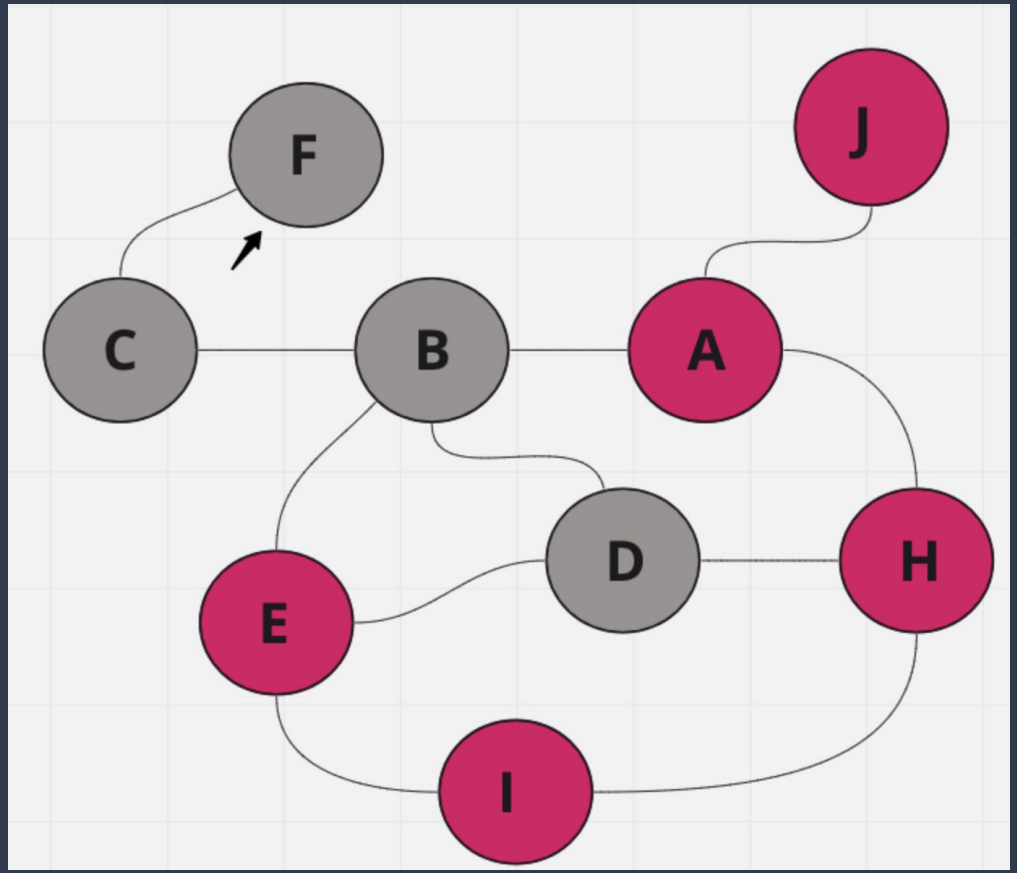
DFS



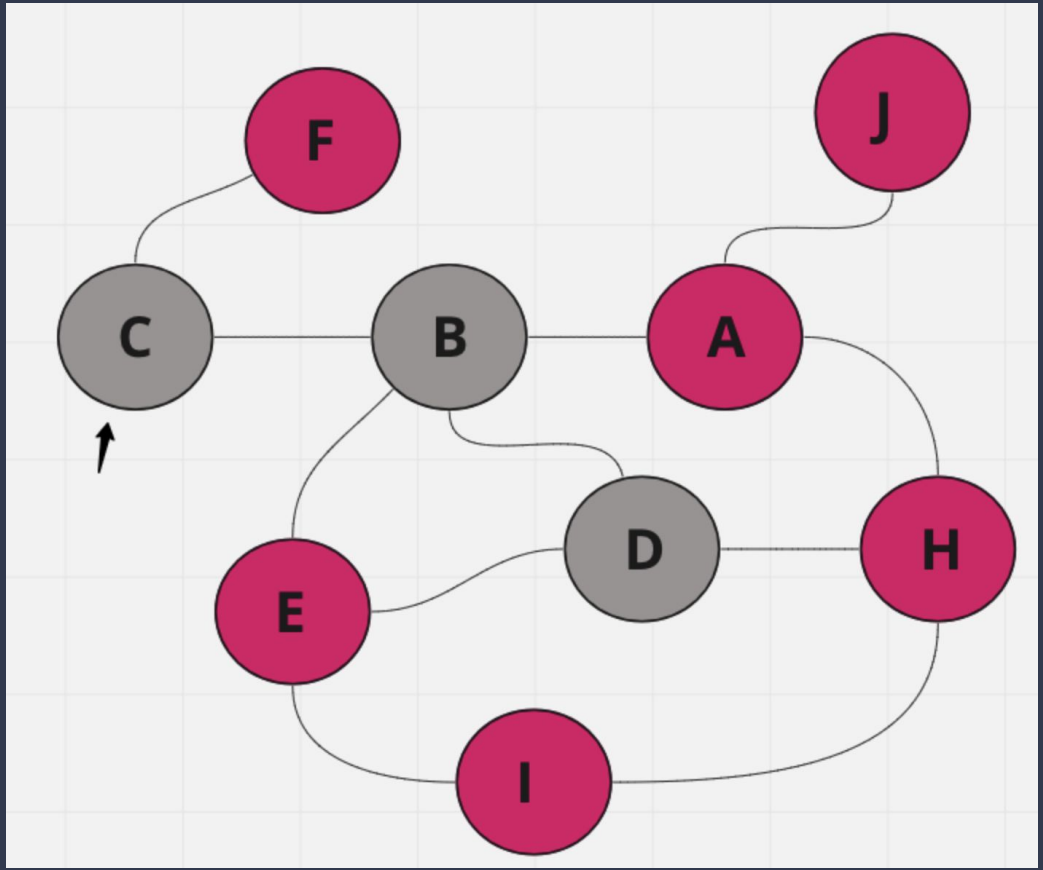
DFS



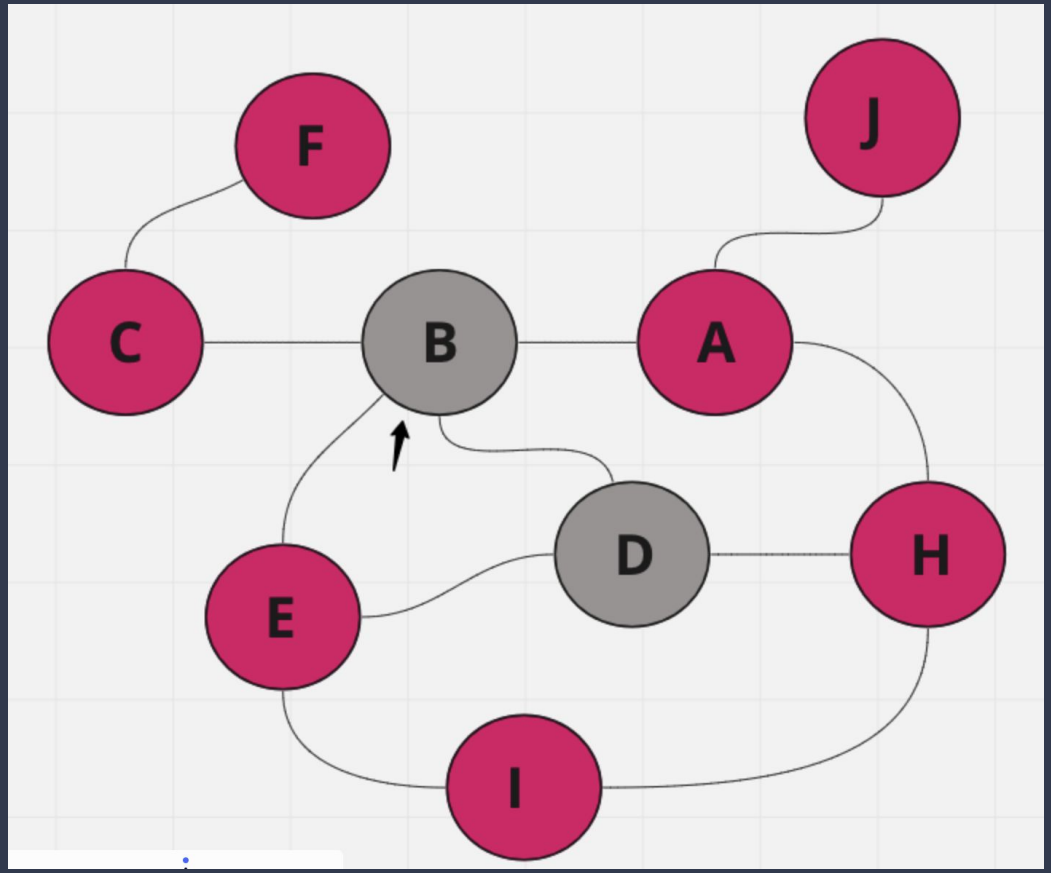
DFS



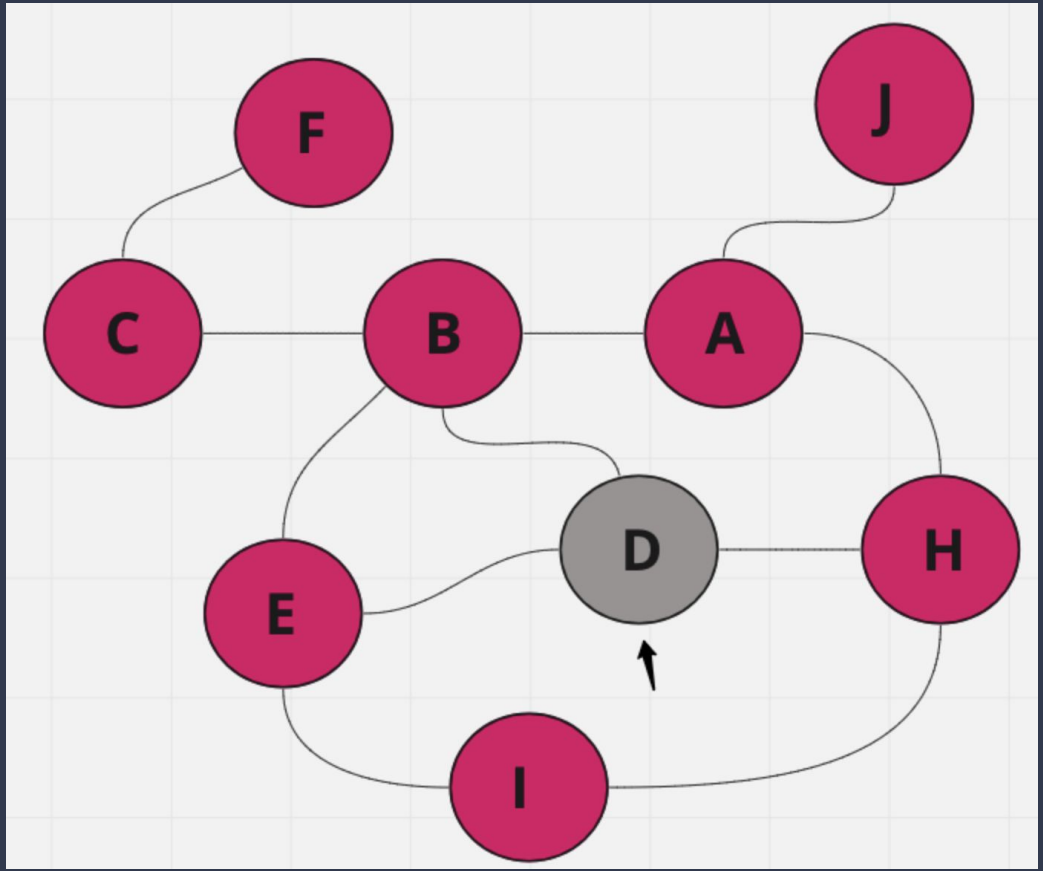
DFS



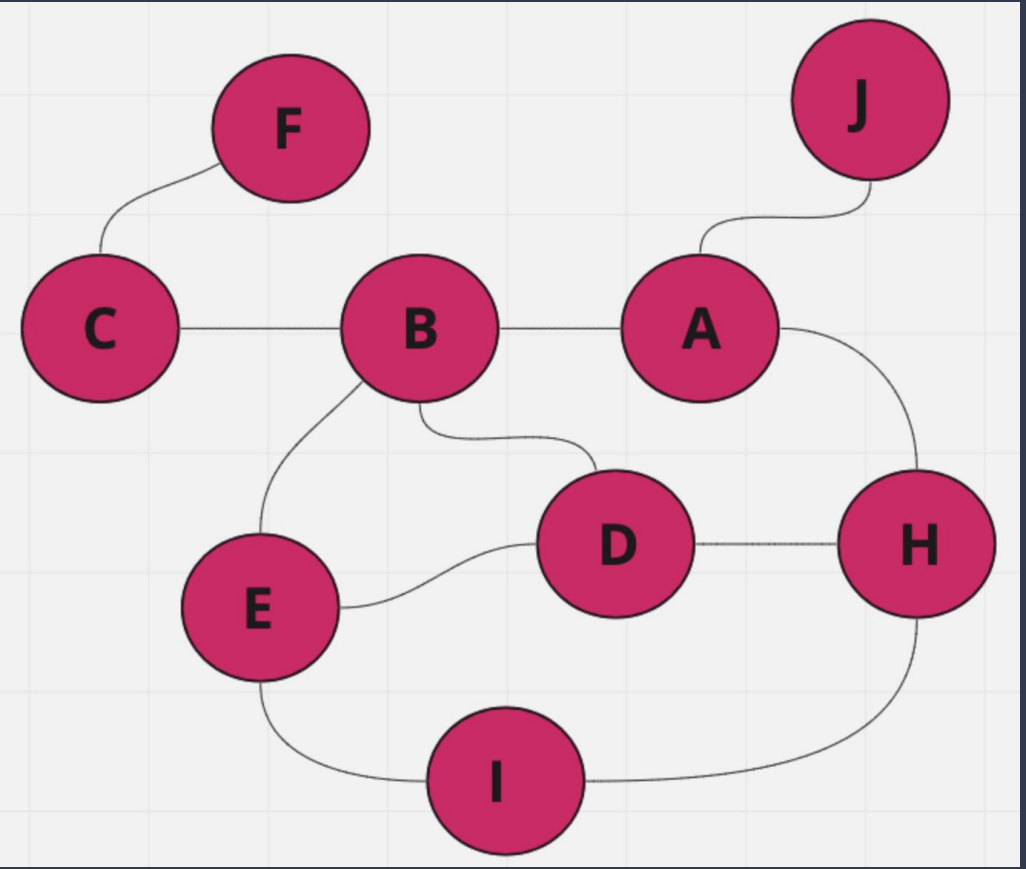
DFS



DFS



DFS



BFS y DFS

- Ambos procesan todos los nodos alcanzables desde un nodo pero en distinto orden.
- Muchos problemas se resuelven con ambos algoritmos
- Otros, solamente con uno:
  - **BFS**: distancia entre dos nodos
  - **DFS**: problemas que tienen que ver con la estructura de un grafo

Algoritmo de Dijkstra

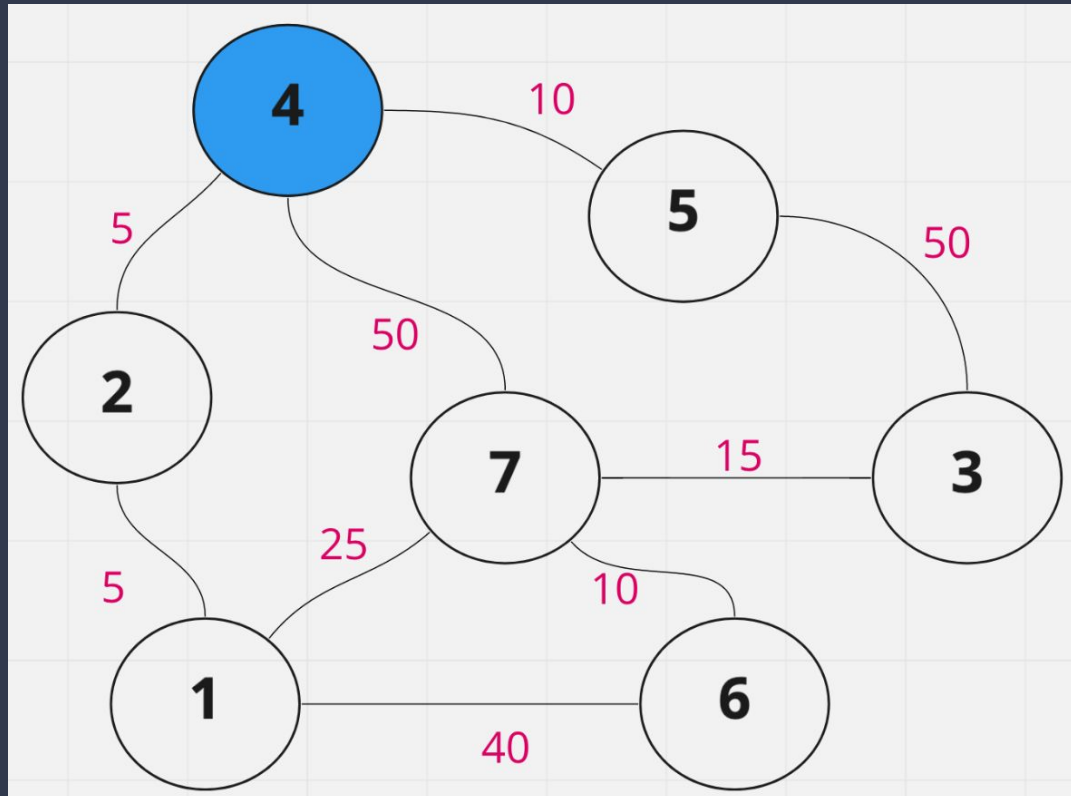
Grafo ponderado

- Las aristas tienen un peso
- **Matriz de adyacencia**: en lugar de guardar 0 o 1, almacenamos el peso
- **Lista de adyacencia:** nuestro vector<vector<int > > pasa a ser vector<vector<pair<int,int> > > donde guardamos destino y peso.

Dijkstra

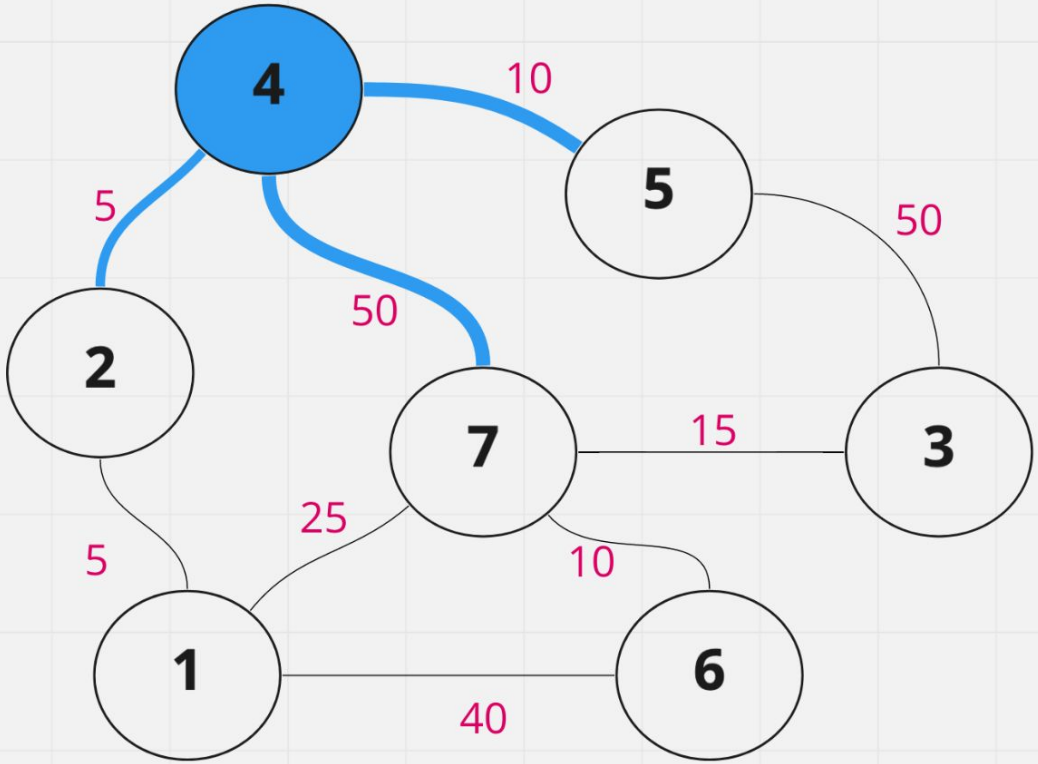
- Determina el camino más corto desde un vértice hacia el resto cuando las aristas tienen peso.
- ¿Cómo funciona?
  - La distancia al nodo origen es 0.
  - Se procesan todos los nodos eligiendo el que tiene menor distancia al origen
  - En cada vecino actualizamos la distancia solamente si la nueva distancia es menor a la que tenía guardada

Dijkstra



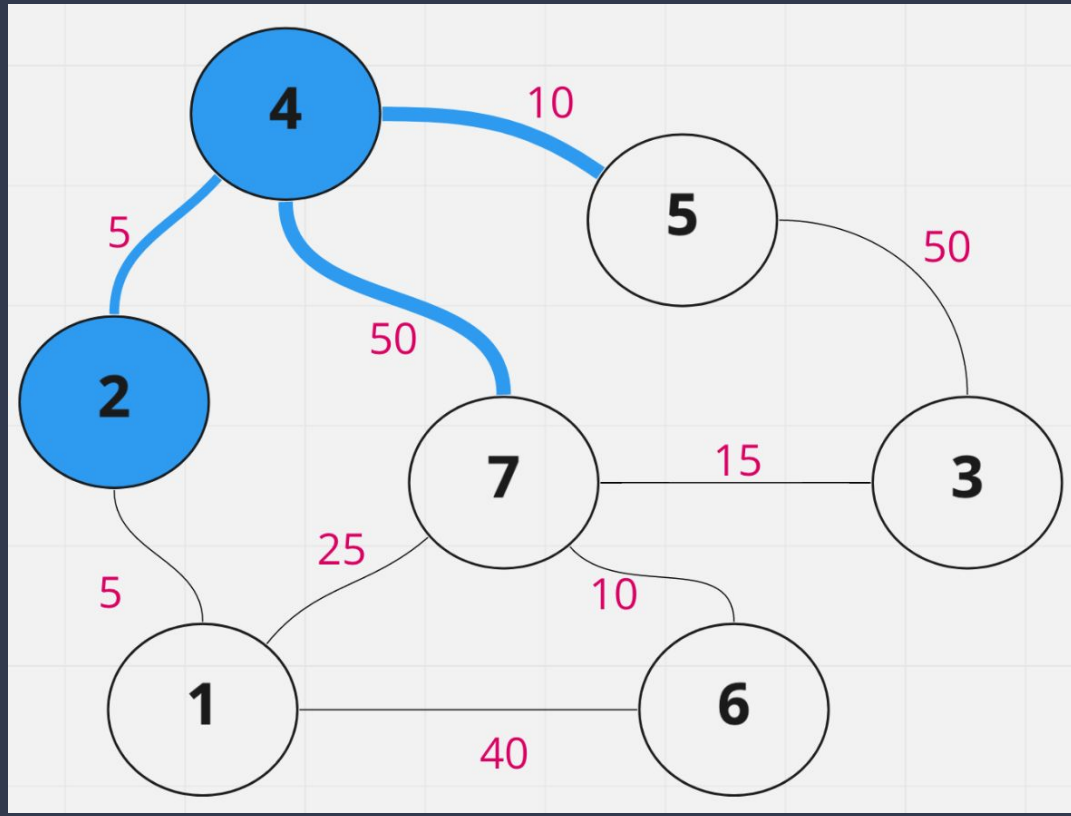
∞	∞	∞	0	∞	∞	∞
1	2	3	4	5	6	7

Dijkstra



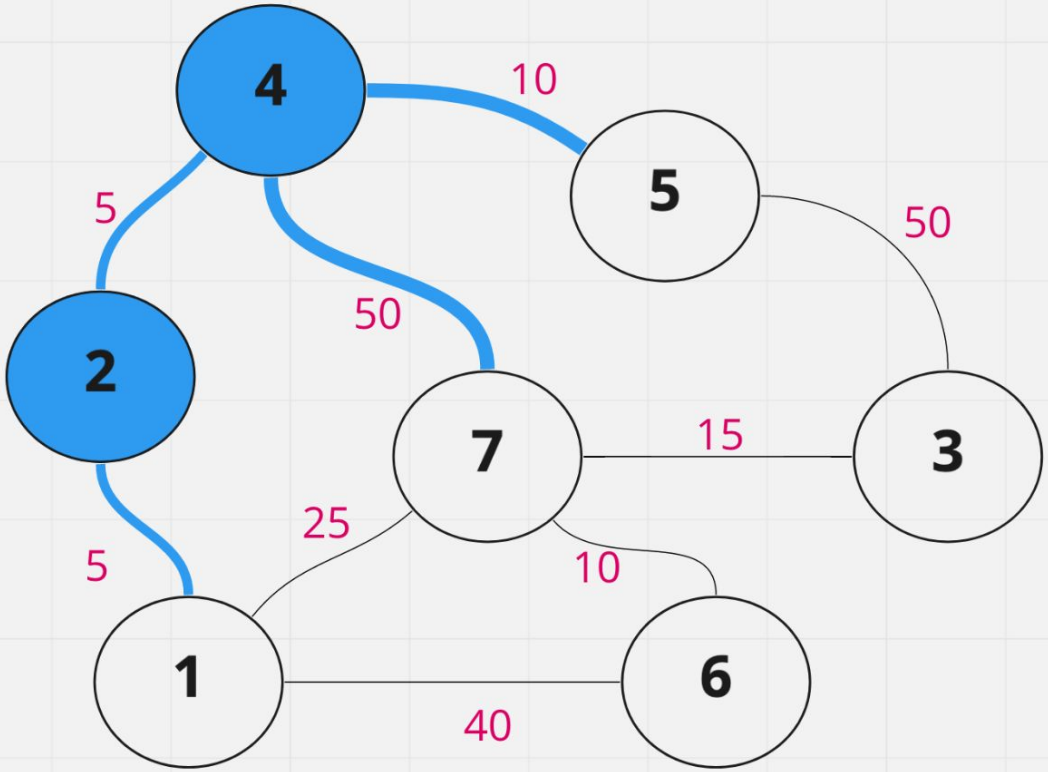
∞	5	∞	0	10	∞	50
1	2	3	4	5	6	7

Dijkstra



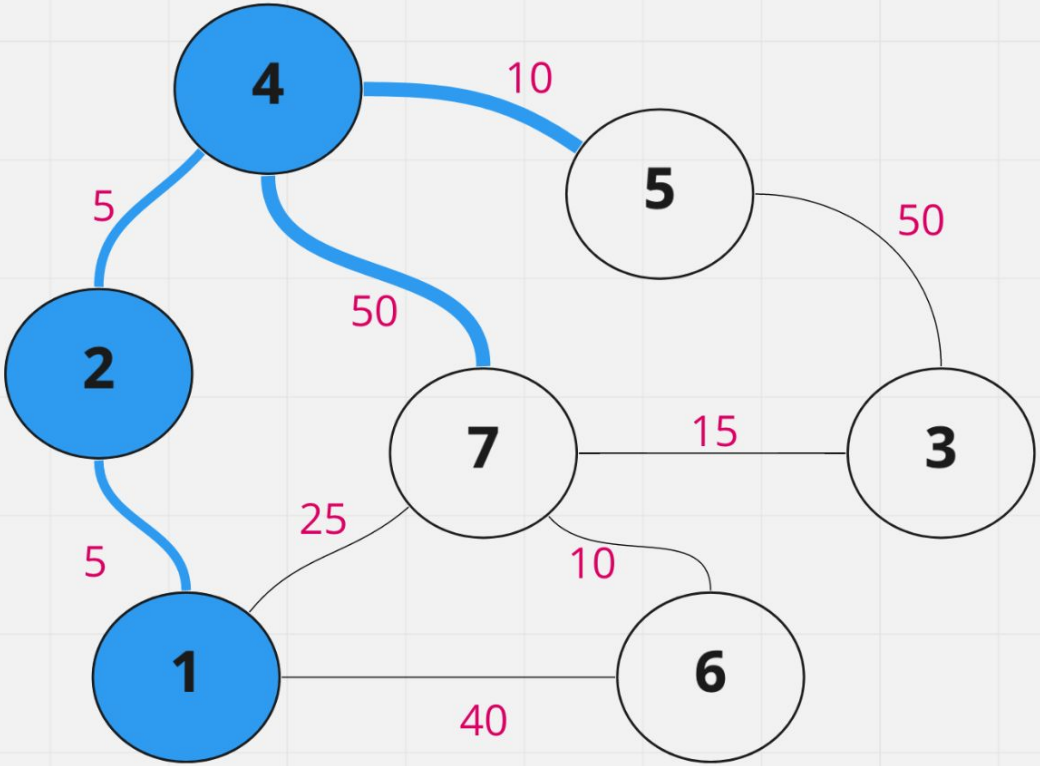
∞	5	∞	0	10	∞	50
1	2	3	4	5	6	7

Dijkstra



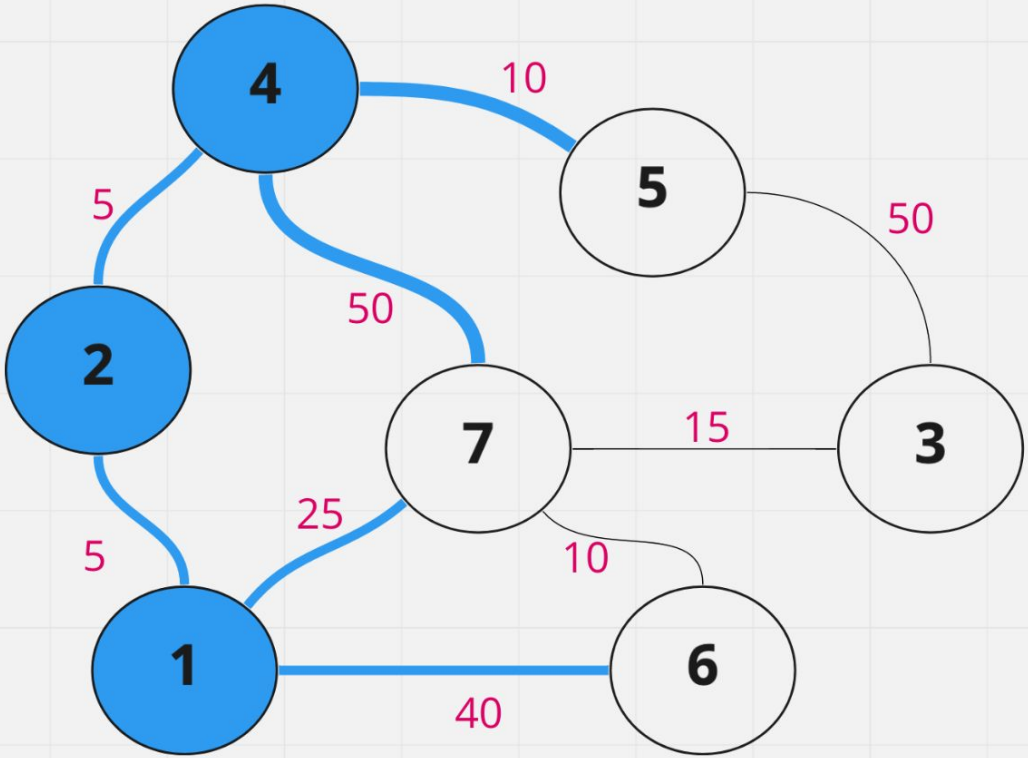
10	5	∞	0	10	∞	50
1	2	3	4	5	6	7

Dijkstra



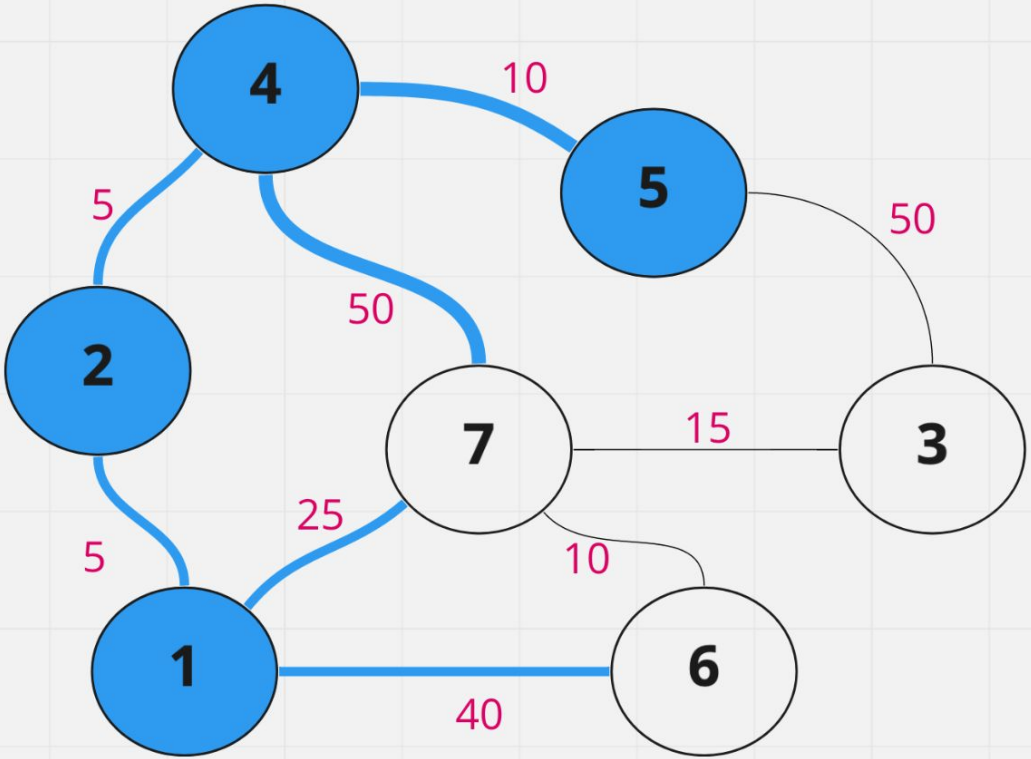
10	5	∞	0	10	∞	50
1	2	3	4	5	6	7

Dijkstra



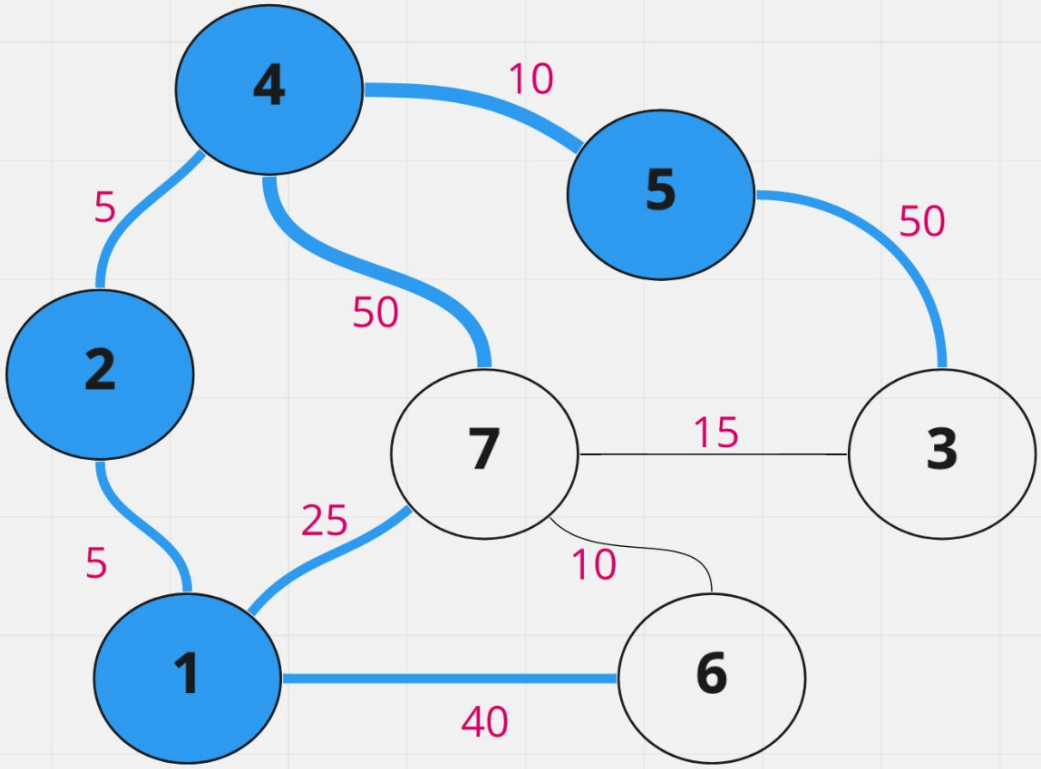
10	5	∞	0	10	50	35
1	2	3	4	5	6	7

Dijkstra



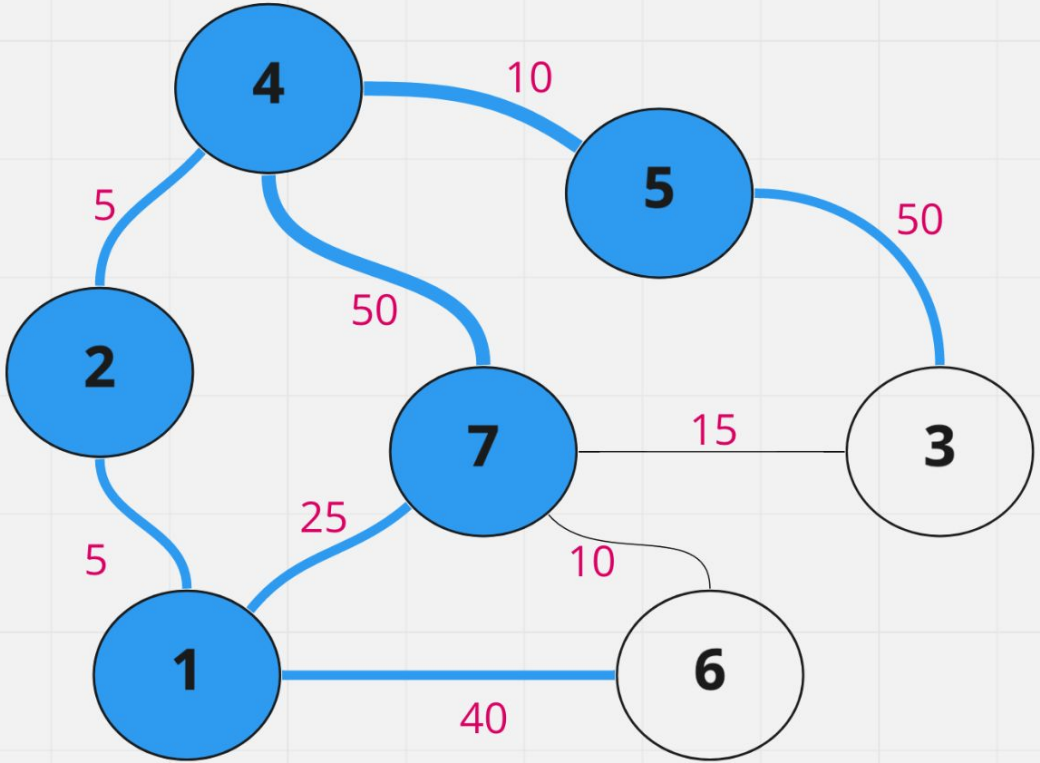
10	5	∞	0	10	50	35
1	2	3	4	5	6	7

Dijkstra



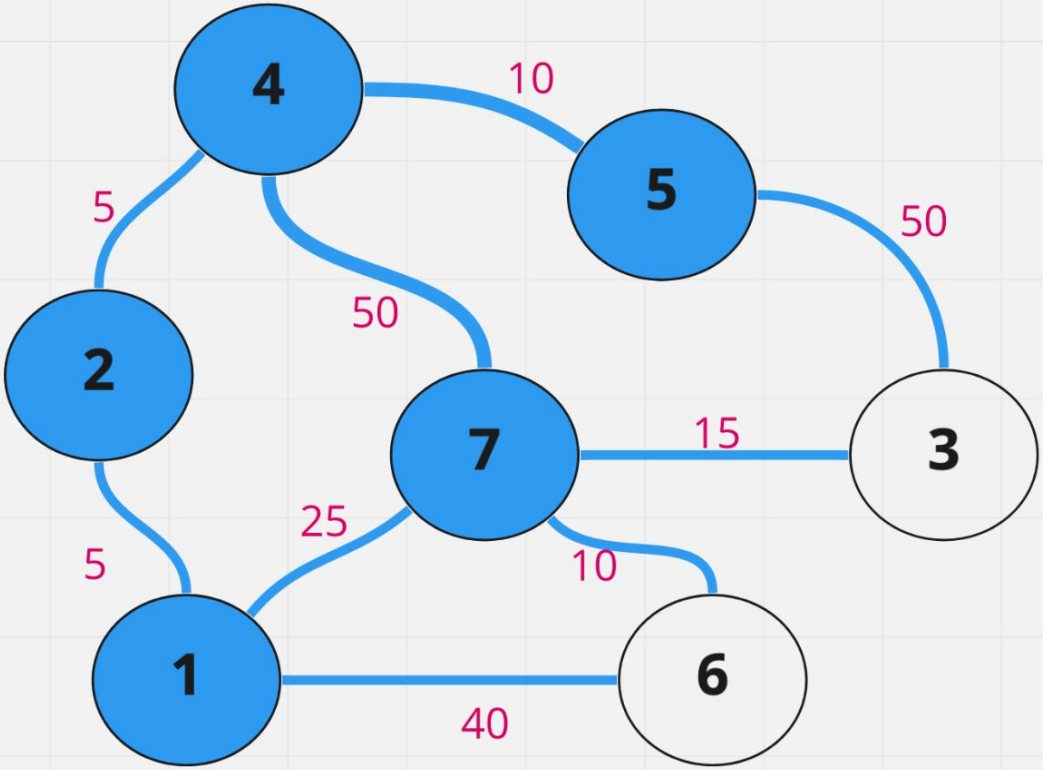
10	5	60	0	10	50	35
1	2	3	4	5	6	7

Dijkstra



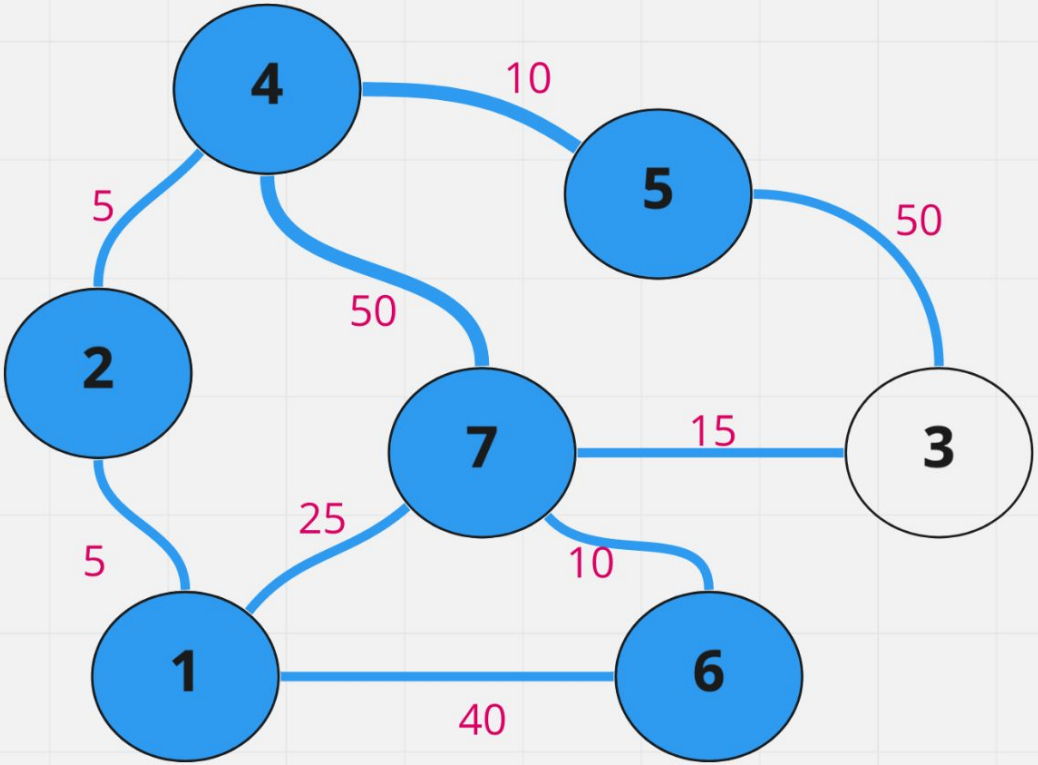
10	5	60	0	10	50	35
1	2	3	4	5	6	7

Dijkstra



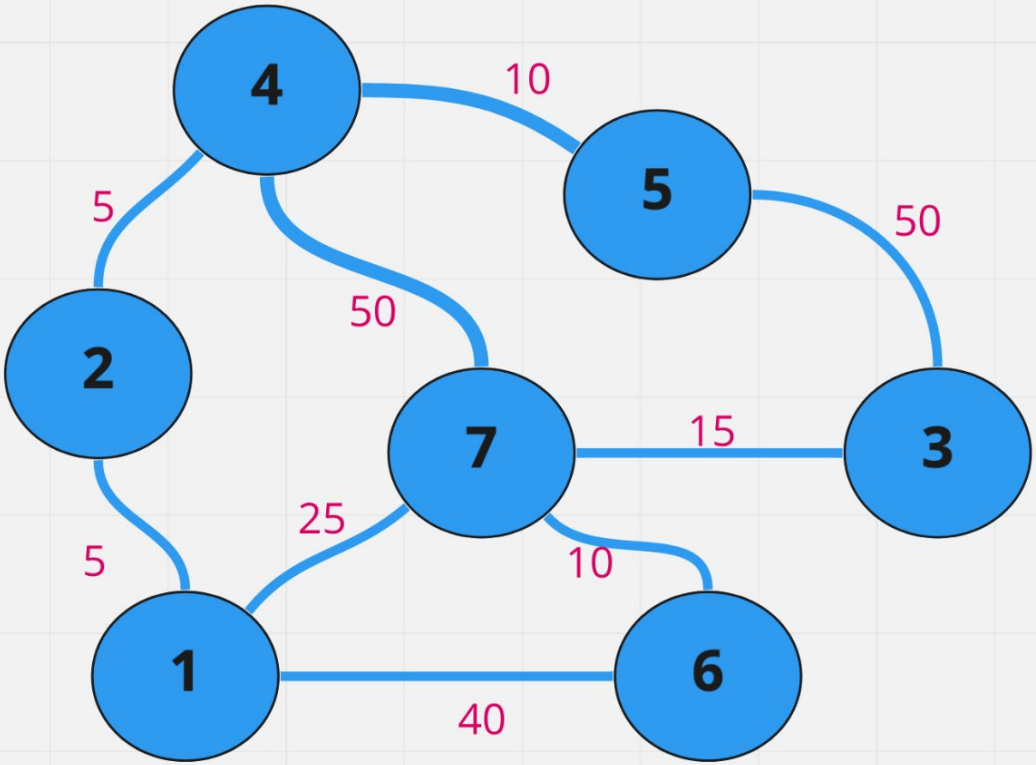
10	5	50	0	10	45	35
1	2	3	4	5	6	7

Dijkstra



10	5	50	0	10	45	35
1	2	3	4	5	6	7

Dijkstra



10	5	50	0	10	45	35
1	2	3	4	5	6	7

¿Dudas?