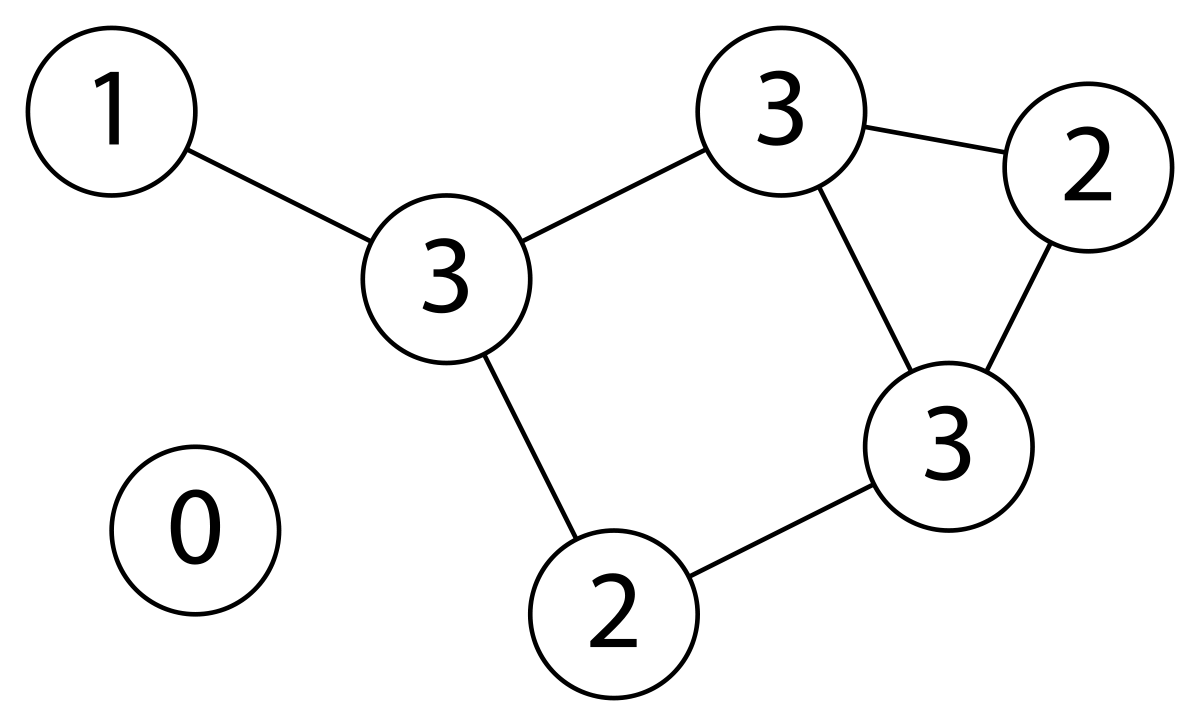
Grafos

Son estructuras de datos generales

Se utilizan en sociología, Ing. eléctrica, industrial, química.



# Grafo G

Agrupa física o conceptualmente entidades.

Este compuesto de:

Vértices, nodos o puntos, que representan a cada una de las entidades.

Aristas, arcos o líneas, que representan las relaciones entre los nodos.

Notacion para grafo : G = {V,A}

V= {1,4,5,7,8} Nodes

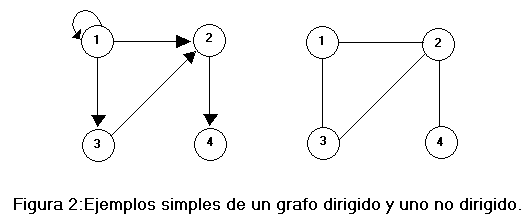
A={(1,4),(4,1),(…),(…)}

## No dirigido

La relación es bidireccional (no tienen flechas)

## Dirigido

La relación es unidireccional (se representan con flechas)



No es posible combinar dir y no dir

## Adyacente

Un nodo es adyacente con otro cuando hay comunicación directa con otro nodo.

Arriba 1 es adyacente a 3

## Peso de arista

Se hace un grafo con peso.

Si se le asigna un peso, todas las aristas necesitan peso.

Representa un costo asociado de trasladarme de un vértice a otro (nodo)

## Grado de un vértice

### No Dirigidos

Para grafos no dirigidos es la cantidad de aristas relacionadas con 5 (fig2: 2 es de grado 3)

### Dirigido

Hay un grado de input y un grado de output

Input: Son la cant aristas que llegan que llegan a ese vértice (fig2: 2 es de grado 2 input)

Output: Son la cant aristas que llegan que salen a ese vértice (fig2: 2 es de grado 1 output)

## Rutas

Es un conjunto de vértices que crean la ruta de un V0 a un Vn. Donde V0 y Vn pueden ser el mismo vertice.

### Ruta simple

Si todos los vértices en la ruta entre V0 y Vn son diferentes (no necesariamente V0 y Vn son distintos), se considera que es una ruta simple.

### Ciclo

Es una ruta simple donde v0 y vn son iguales.

### DAG

(grafo directo Acíclico)

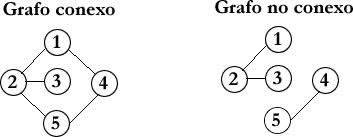
Un grafo donde no se forma ni un solo ciclo.

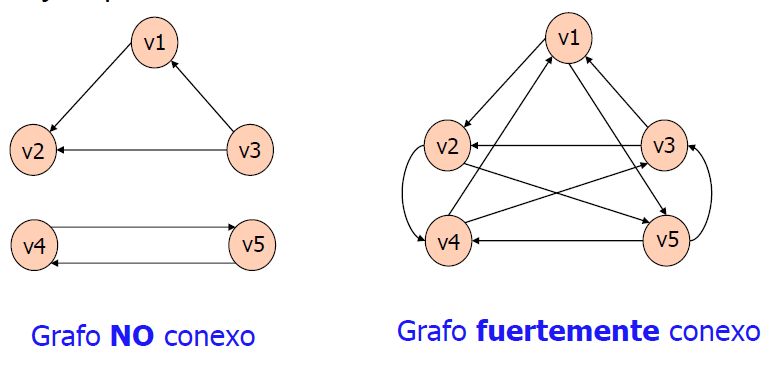
## Grafos Conectados

Siempre existe una ruta (directa o indirecta) ente cualquier par de nodos del grafo.

## Grafos fuertemente conectados

Se conectado y dirigido. (no tiene por qué ser rutas directas )





El grafo puede implementarse mediante:

## Matrices de adyacencia (arreglo bidimensional)

Es mejor para grafos densos (con muchos arcos)

## Listas de adyacencia (Listas Enlazadas)

Es mejor para grafos no tan densos

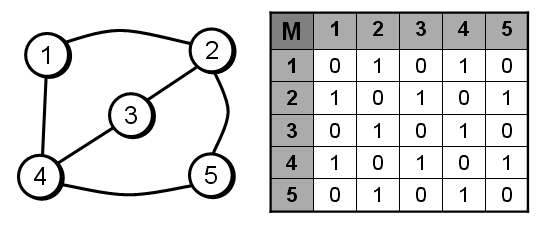
# Matriz de adyacencia

G = {V,A}

A= {(Vi,vj)} 🡪 se puede ir de vi a vj

aij puede ser 0 o 1.

0 si no hay relación. 1 si sí hay.



Son relaciones/comunicaciones directas.

(peso 0 = no hay arista)

Si se hace con pesos, entonces el 0 aplica igual, pero si hay relación se pone el valor del peso.

# Lista de Adyacencia

Cada elemento de la lista representa un nodo y tiene otra lista que representa las relaciones.

# Recorrer un grafo

Significa Visitar (bajo la misma def de árbol), todos los al menos una vez desde un nodo especifico.

## Breadth-first

Utiliza una cola con los vertices marcados

FIFO

Algoritmo:

* Toma un nodo inicial
* Marca el escogido como visitado
* Agrega a la cola los nodos a los que puedo ir desde el último visitado
* Marca el primero como visitado (cola)
* Repita hasta que no pueda agregar más a la cola

## Depth-First

Utiliza un stack/pila

LIFO

Algorirmo:

* Toma un nodo inicial
* Marca el escogido como visitado
* Agrega a la pila los nodos a los que puedo ir desde el último visitado
* Marca el ultimo como visitado (stack)
* Repita hasta que la pila esta vacía

# Shortest-path: Dijkstra

Buscar la mejor ruta: menor cantidad de aristas con la menor suma de peso

De un nodo especifico (no cualquiera) a cualquier otro.

Es un algoritmo ávido borás(toma la mejor decisión en el momento con la información que tiene)

# Algoritmo de Floyd