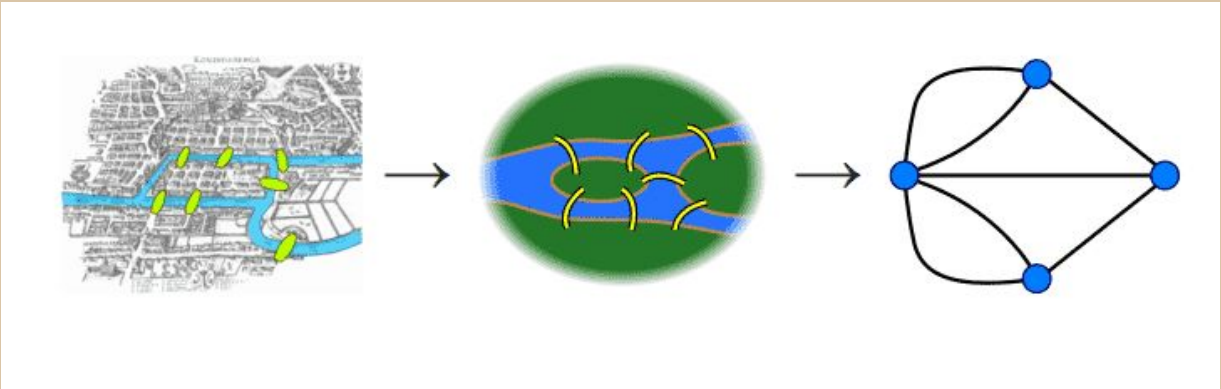
Gráficas

Gráficas

Una gráfica **G = (V, E)** es un conjunto **V** de vértices o nodos y un conjunto **E ⊆ V x V** de aristas.

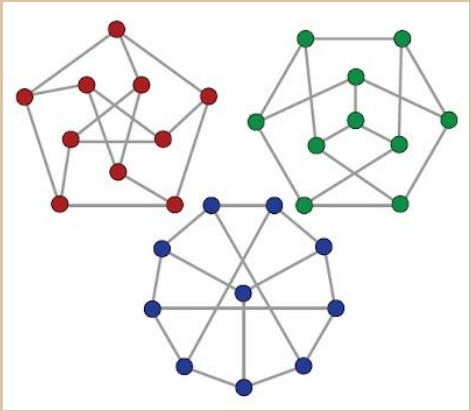


Puentes de Königsberg

Gráficas no dirigidas

Sea G = (V, E) una gráfica, es no dirigida si se cumple que:

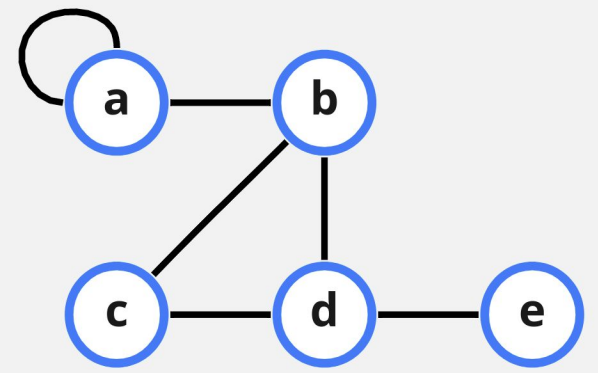
$$(u, v) \in E \Leftrightarrow (v, u) \in E$$



ED
ESTRUCTURAS DE DATOS

Implementaciones de gráficas no dirigidas

- **Lista de aristas**: Se tienen todas las aristas de una gráfica en una lista. Dado que es no dirigida, no es necesario tener las parejas simétricas.

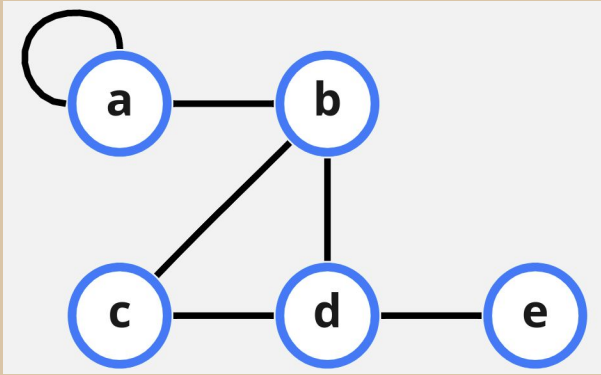


(a, a), (a, b), (b, c), (b, d), (c, d), (d, e)

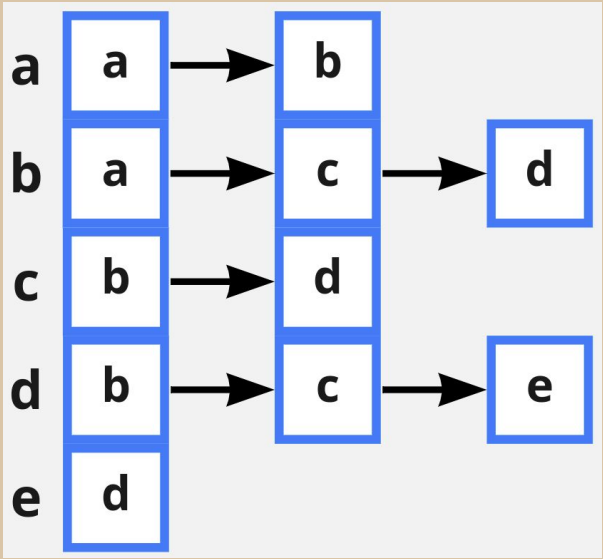
- **Matriz de adyacencias**: Se usa una matriz en la que por cada vértice de la gráfica se pone una columna y un renglón, de tal forma que si existe la arista (**i, j**), entonces la entrada M[**i, j**] debe representar la existencia de la arista.

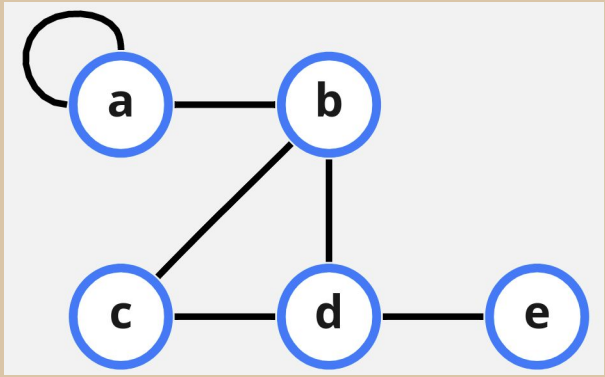
	a	b	c	d	e
a	1	1	0	0	0
b	1	0	1	1	0
c	0	1	0	1	0
d	0	1	1	0	1
e	0	0	0	1	0

	a	b	c	d	e
a	1	1	0	0	0
b		0	1	1	0
c			0	1	0
d				0	1
e					0

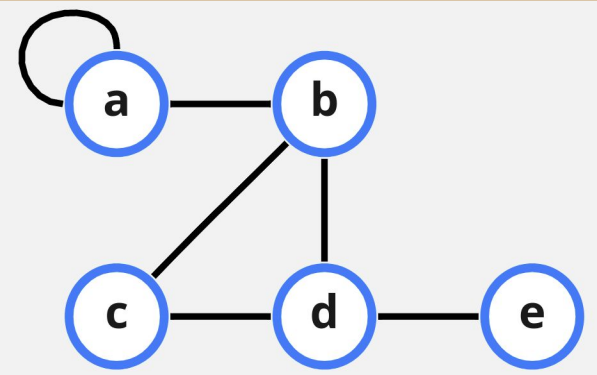


- **Listas de adyacencias**: Por cada vértice se tiene una lista de los nodos con los que está conectado por medio de una arista.





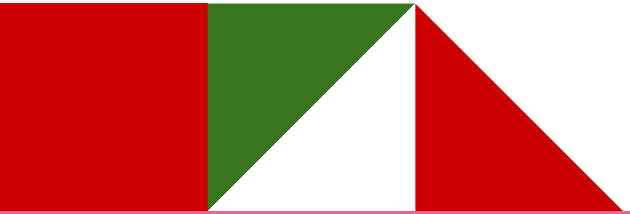
- **Hash de adyacencias**: Se tiene una tabla de dispersión H, en la que las llaves son parejas de vértices (u, v), de tal forma que si existe una arista para ellos, entonces el valor guardado en H[(u, v)] debe representar la presencia de la arista.



H[(a, a)] = true	H[(a, b)] = true
H[(b, c)] = true	H[(b, d)] = true
H[(c, d)] = true	H[(d, e)] = true

Ejercicio

Analiza y compara las distintas formas de implementar una gráfica no dirigida.



Recorrido BFS en gráficas

Para hacer un recorrido BFS, necesitamos definir un vértice **s** como inicio del recorrido. Necesitaremos además una cola de vértices **q**.

```
1  Agregamos a s a q.
2  Mientras q no esté vacía:
3     Sacamos al siguiente vértice v de q.
4     Si v no ha sido visitado:
5        Marcamos a v como visitado.
6        Procesamos a v.
7        Metemos a los vecinos no visitados de v a q.
```

ED
ESTRUCTURAS DE DATOS

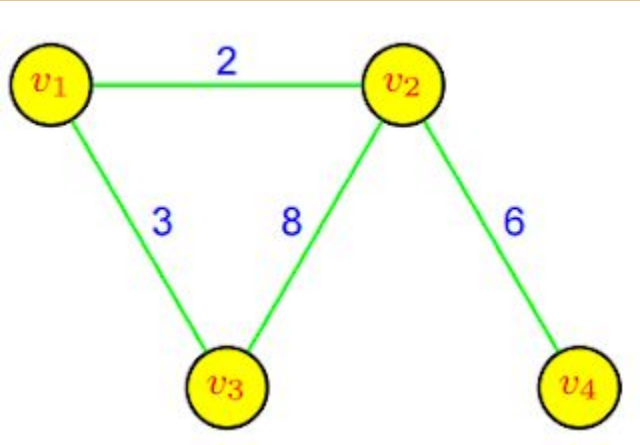
Recorrido DFS en gráficas

Para hacer un recorrido DFS, necesitamos definir un vértice **s** como inicio del recorrido y aplicarle la siguiente función:

```
1  dfs( v ):
2     Marcamos a v como visitado.
3     Procesamos a v.
4     Por cada vecino w de v:
5        Si w no ha sido visitado, aplicamos dfs( w ).
```

Gráfica con pesos

Sea **G** = (**V**, **E**) una gráfica, tiene pesos si tiene una función *w:* **E** → $\mathbb{R}^+$ donde al número *w(* **e** *)* se le conoce como el peso de la arista **e**.



ED
ESTRUCTURAS DE DATOS

Caminos y trayectorias

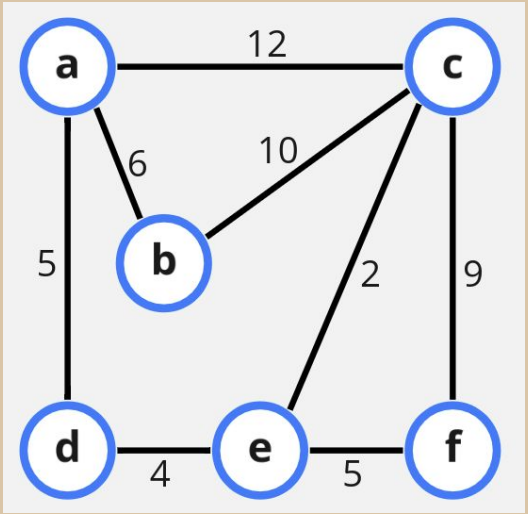
Un camino en una gráfica $G = (V, E)$ es una secuencia de vértices $v_1, v_2, \dots , v_k$ tal que $v_i \in V$ para $1 \leq i \leq k$ y $(v_{i-1}, v_i) \in E$ para $1 < i \leq k$.

Una trayectoria en una gráfica $G = (V, E)$ es un camino $T = v_1, v_2, \dots , v_k$ tal que $v_i \neq v_j$ si $i \neq j$.

Si G tiene pesos, entonces el peso de una trayectoria T es la suma del peso de sus aristas.

Ejercicio

Encuentra la trayectoria de peso mínimo que va desde **a** hasta **c**.



Algoritmo de Dijkstra - 1956

Para encontrar la trayectoria de peso mínimo entre dos vértices **s** y **t** se puede usar el siguiente algoritmo que necesita de una cola de prioridad **q** donde el criterio de prioridad de un nodo **v** será la distancia a **s** denotada como d(**v**).

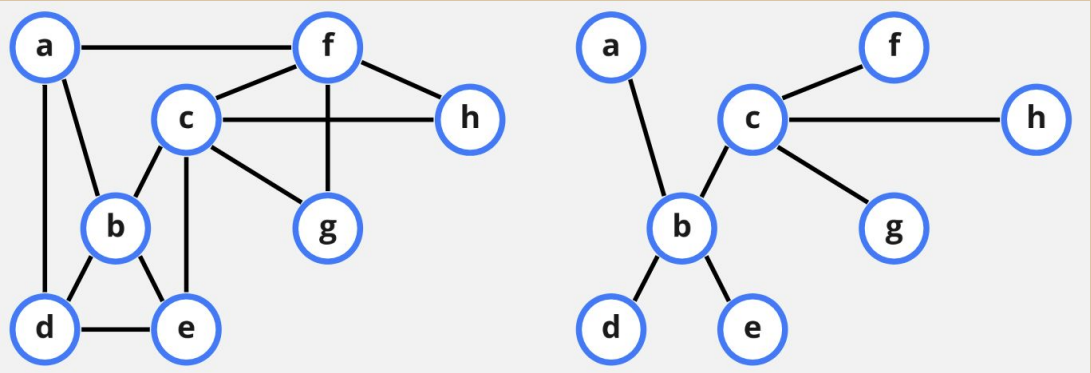
1 Se incializa d(**s**) = 0 y para el resto vértices d(**v**) = ∞.
2 Se meten todos los vértices a **q**.
3 Mientras **q** no esté vacía:
4 Sacamos al siguiente vértice **v** de **q**.
5 Por cada vecino **u** de **v**:
6 Si d(**u**) > d(**v**) + w((**u**, **v**)):
7 Actualizamos d(**u**) = d(**v**) + w((**u**, **v**))

ED
ESTRUCTURAS DE DATOS

Árbol generador

Un árbol es una gráfica conexa y acíclica.

Un árbol generador para una gráfica G = (V, E) es una subgráfica de G que posee la cualidad de ser un árbol y contar con todos los vértices de G.



ED
ESTRUCTURAS DE DATOS

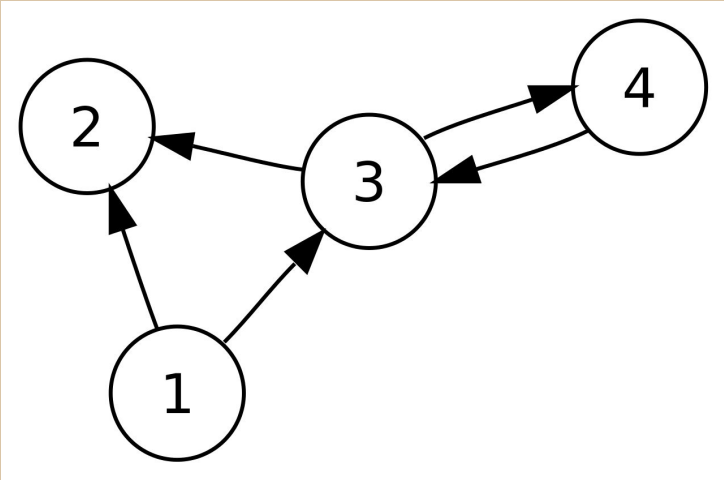
Algoritmo de Kruskal

Para encontrar el árbol generador de peso mínimo se puede usar el siguiente algoritmo:

1 Se crea un conjunto para cada vértice.
2 Se ordenan las aristas por su peso.
3 Por cada arista **e**, de menor a mayor peso:
4 Si **e** conecta vértices de conjuntos distintos C_1 y C_2:
5 Se fusionan C_1 y C_2.
6 Se agrega **e** al árbol generador de peso mínimo.

Gráficas dirigidas

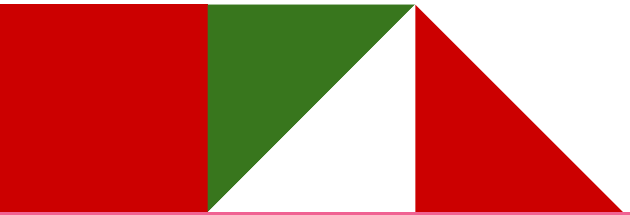
Sea G = (V, E) una gráfica, es dirigida si se cumple que si (u, v) ϵ E, no necesariamente (v, u) ϵ E.



ED
ESTRUCTURAS DE DATOS

Pregunta

¿Qué habría que modificar en las alternativas que vimos de implementación de gráficas no dirigidas para poder implementar gráficas dirigidas y/o con pesos?



Grado y vecindario

Sea **G** = (**V**, **E**) una gráfica sin lazos, es decir, aristas de la forma (**a**, **a**):

El grado de un vértice **v**, denotado como d_G(**v**), es la *cantidad* vértices **u** ϵ **V**, tal que (**v**, **u**) ϵ **E**. (En una gráfica dirigida son solo las aristas que salen de **v**).

El vecindario de un vértice v, denotado como N(**v**), son todos los vértices **u** ϵ **V**, tal que (**u**, **v**) ϵ **E**. (En una gráfica dirigida son solo las aristas que apuntan a **v**).

PageRank simplificado - 1998

En una gráfica dirigida, para saber qué tan relevante es cada nodo v se usa la función $r_i(v)$ y el siguiente algoritmo:

1 Para cada vértice **v** se inicializa $r_0(\mathbf{v}) = 1$
2 Desde i = 1 hasta k:
3 Por cada vértice **v**:
4 Se establece $r_i(\mathbf{v}) = 0$
5 Por cada vértice **w** ϵ N(**v**):
6 $r_i(\mathbf{v}) \mathrel{+}= r_{i-1}(\mathbf{w}) / d_G(\mathbf{w})$

ED
ESTRUCTURAS DE DATOS

Ejercicio

Ejecuta dos iteraciones del algoritmo PageRank en la siguiente gráfica: