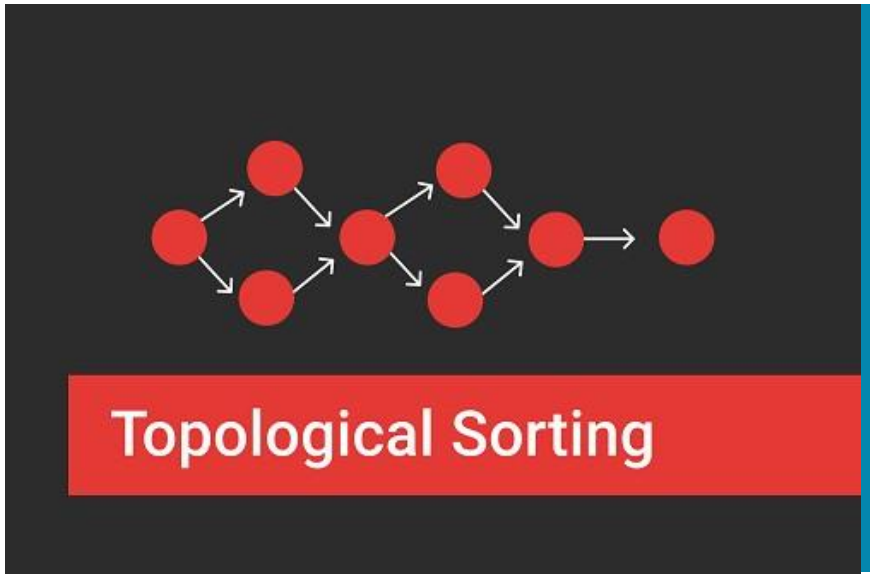


Complejidad Algorítmica

Unidad 1: Comportamiento asintótico, métodos de búsquedas y grafos

Módulo 6: Grafos – Ordenamiento Topológico

Complejidad Algorítmica

Semana 6 / Sesión 1

MÓDULO 6: Grafos – Ordenamiento Topológico

Contenido

1. Definición de Orden Topológico
2. Usos de la ordenación topológica
3. Componentes Fuertemente Conexos (SCC)
4. Algoritmo de Kosaraju

Preguntas

1. Definición de Orden Topológico

¿Qué es el orden topológico en un grafo?

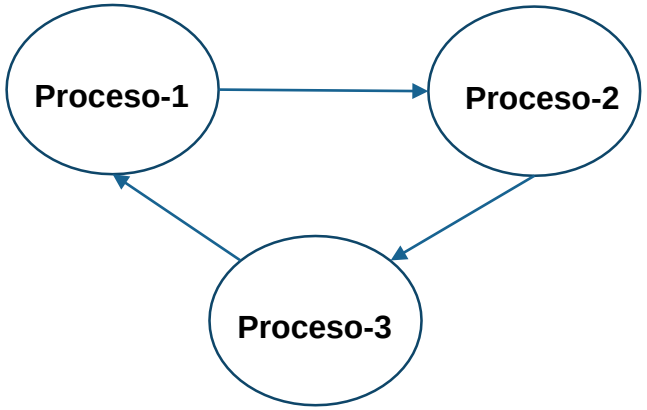
- **Orden o clasificación topológica** es un ordenamiento lineal que toma un gráfico dirigido y devuelve una matriz de los nodos donde cada nodo aparece antes que todos los nodos a los que apunta.
- El primer vértice en la clasificación topológica siempre es un vértice con grado de entrada 0 (es decir, un vértice sin bordes entrantes).
- Esta ordenación **solo se aplica sobre grafos dirigidos acíclicos** (DAG).

¿Y por que esta clasificación no se puede aplicar sobre grafos dirigidos cíclicos?

1. Definición de Orden Topológico

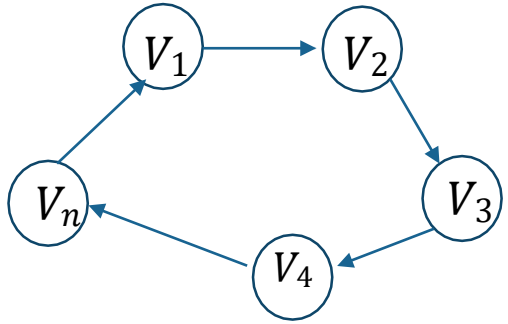
EJEMPLO: GRAFO DIRIGIDO CICLICO

- En la siguiente imagen, el grafo no tiene un vértice con cero requisitos previos (se trata de un grafo dirigido cíclico).



- Vemos que en el tramo (proceso-1 → proceso-2), el proceso-2 puede iniciarse solo cuando el proceso-1 ya ha terminado. Podemos decir que el proceso-2 depende del proceso-1, el proceso-3 del proceso-2 y el proceso-1 del proceso-3.

Para probarlo, supongamos que hay un ciclo hecho de los vértices:



V_1, V_2, V_3, V_4, V_n

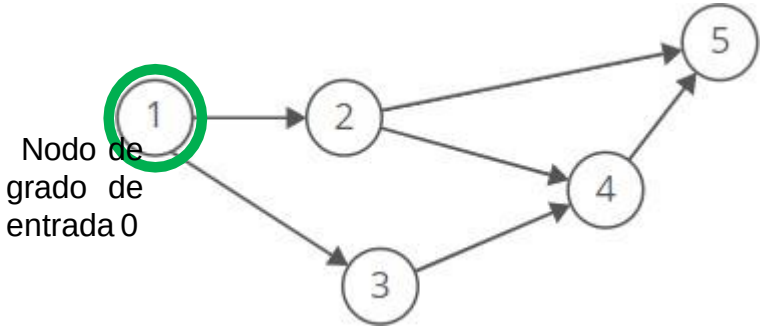
Observamos que:

- Existe un borde dirigido entre V_i y V_{i+1} (1 <= i < n) y entre V_ny V_1
- Si intentamos hacer una ordenación topológica, entonces $V_n$debe venir antes que V_1debido al borde dirigido de V_na V_1, pero $V_n$no debiera tener ningún borde entrante (y no se cumple)
- Por lo tanto, la clasificación topológica se puede lograr solo para gráficos dirigidos y acíclicos.
- Para tener una ordenación topológica, el grafo no debe contener ningún ciclo.

1. Definición de Orden Topológico

EJEMPLO

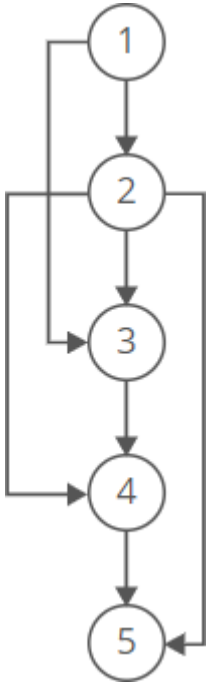
- Tenemos el siguiente **grafo dirigido acíclico** de grado 5



Observamos que

- Dado que el nodo 1 apunta a los nodos 2 y 3, el nodo 1 aparece antes que ellos en el ordenamiento.
- Y, dado que los nodos 2 y 3 apuntan al nodo 4, aparecen antes que él en el ordenamiento.

Ordenación Topológica del grafo

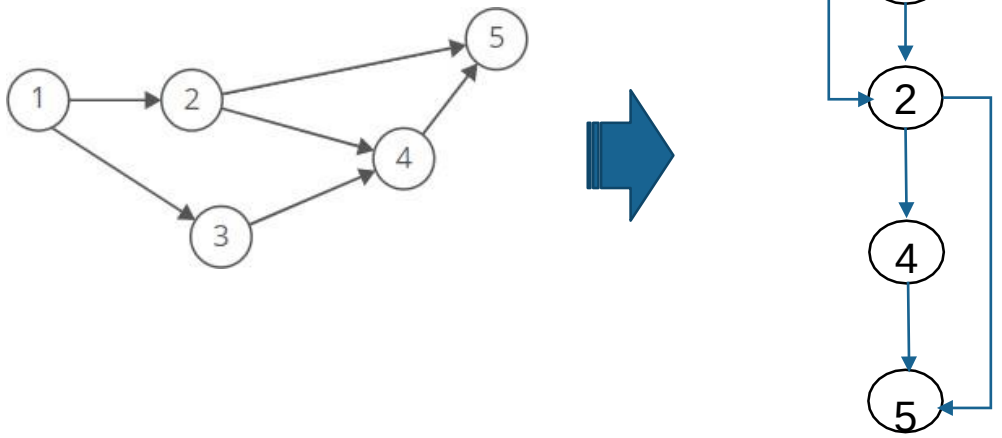


Entonces [1, 2, 3, 4, 5] sería una ordenación topológica del gráfico.

1. Definición de Orden Topológico

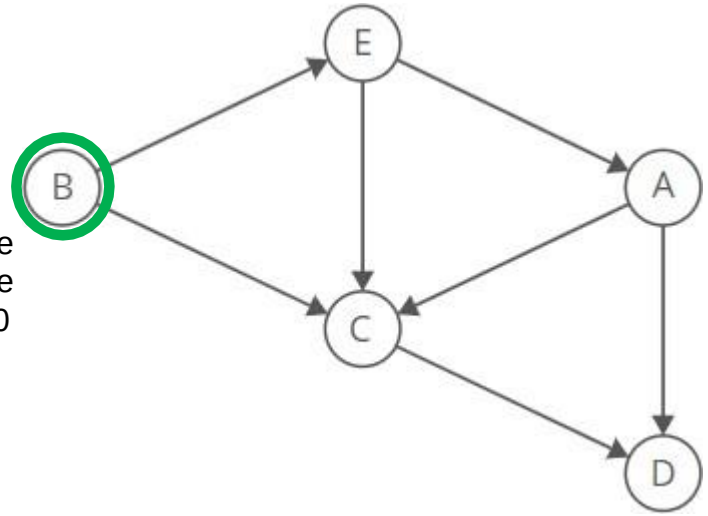
¿Puede un grafo tener <u>más</u> de un ordenamiento topológico válido?

Claro que **SI**, del grafo anterior, la ordenación topológica [1, 3, 2, 4, 5] también funciona.



1. Definición de Orden Topológico

¿Cómo podemos producir un ordenamiento topológico para este grafo dirigido?



Nodo de grado de entrada 0

Pasos:

1. Ubicar el primer nodo del ordenamiento topológico. Este nodo no puede tener bordes dirigidos entrantes (tiene un grado de entrada 0).

   Identificamos al nodo B con un grado de entrada cero y lo agregaremos al ordenamiento topológico.

B				

Ordenamiento topológico

1. Definición de Orden Topológico

Pasos (continuación):

2. Una vez que se agrega un nodo al ordenamiento topológico, podemos sacar el nodo y sus bordes salientes del gráfico.

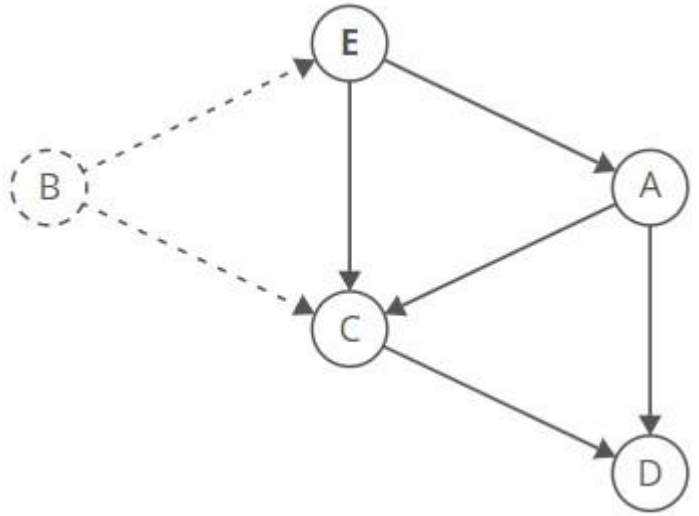
B				

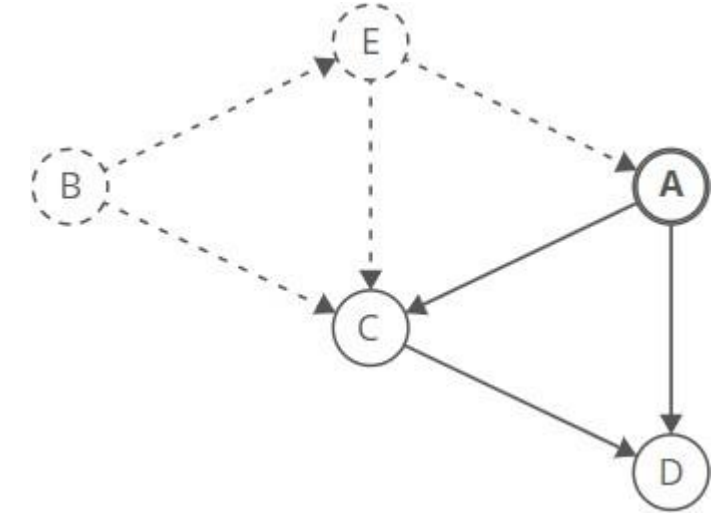
Ordenamiento topológico

3. Repetir el enfoque anterior: buscar cualquier nodo con un grado de entrada de cero y agregarlo a la ordenación.

B	E			

Ordenamiento topológico





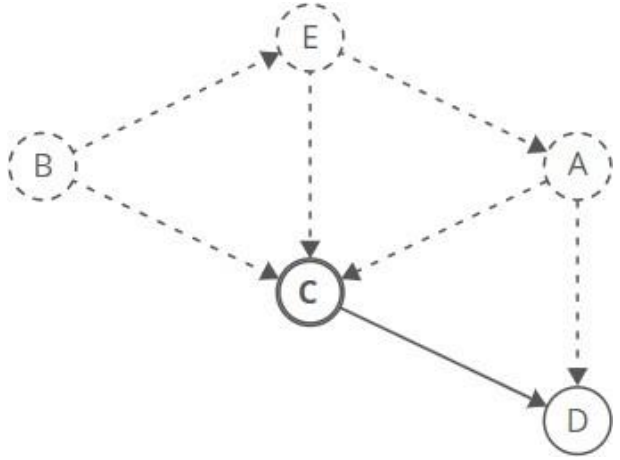
1. Definición de Orden Topológico

Pasos (continuación):

4. Repetir

B	E	A		

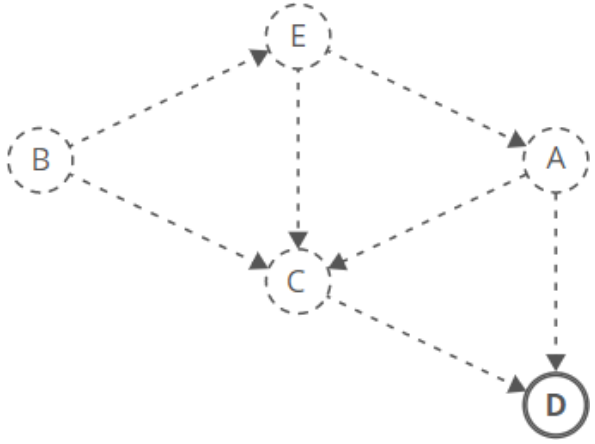
Ordenamiento topológico



5. Repetir

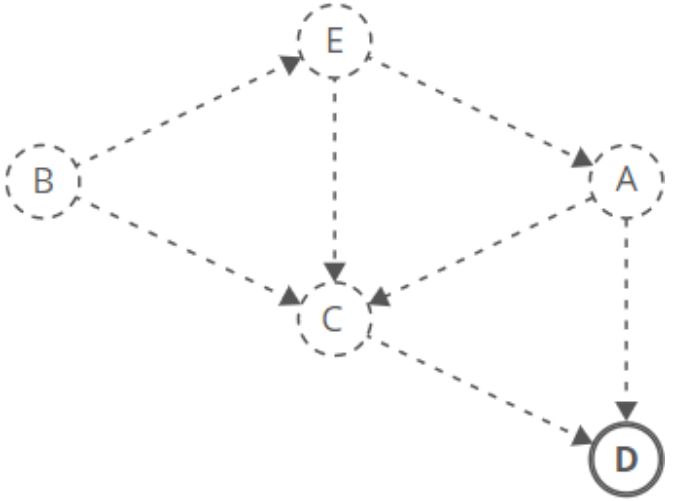
B	E	A	C	

Ordenamiento topológico



1. Definición de Orden Topológico

Pasos (continuación):



6. Repetir hasta que ya no queden mas nodos por evaluar.

B	E	A	C	D

Ordenamiento topológico

Nota: esta no es la única forma de producir un ordenamiento topológico.

1. Definición de Orden Topológico

ALGORITMO

Usaremos la estrategia que describimos anteriormente:

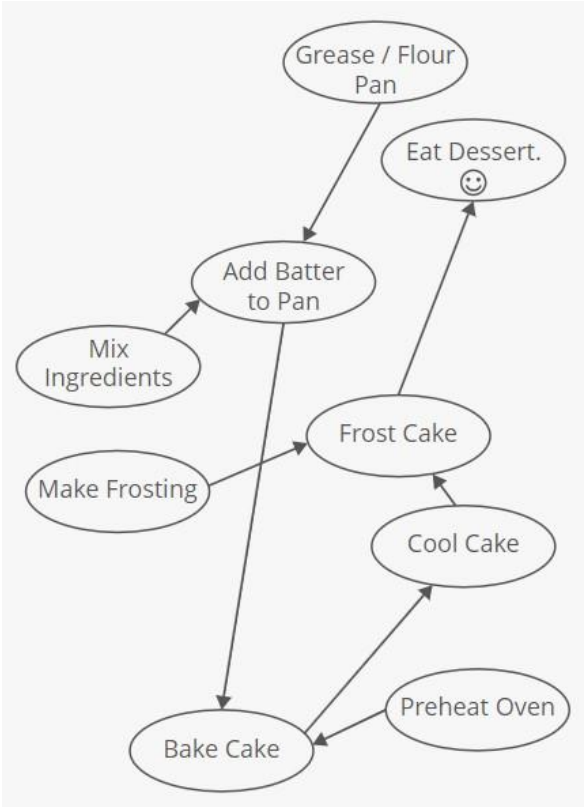
1. Identifique un nodo sin bordes entrantes.
2. Agregue ese nodo a la lista de orden topológico.
3. Eliminarlo de la gráfica.
4. Repetir.

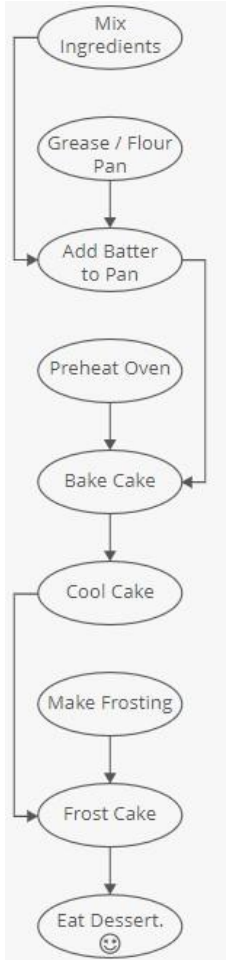
Seguiremos recorriendo hasta que no haya más nodos con grado cero. Esto podría suceder por dos razones:

- **No quedan nodos.** Los hemos sacado a todos del gráfico y los hemos agregado al ordenamiento topológico.
- **Quedan algunos nodos, pero todos tienen bordes entrantes.** Esto significa que el gráfico tiene un ciclo y no existe un ordenamiento topológico.

2. Usos de la ordenación topológica

- El uso más común para la ordenación topológica es ordenar los pasos de un proceso donde algunos de los pasos dependen unos de otros.





- Representamos las dependencias usando un gráfico dirigido.
- Usamos la ordenación topológica para proporcionar un ordenamiento válido para abordar todos los pasos.

2. Usos de la ordenación topológica

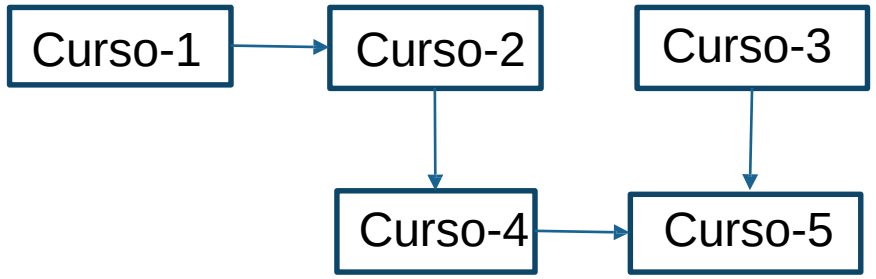
- La creación de componentes en un gran proyecto de software.
- La activación de componentes de hardware en el momento del arranque. (Por ejemplo, la placa base debe inicializar el disco duro antes de que el BIOS intente cargar el gestor de arranque desde el disco).
- Encontrar el ciclo en un gráfico
- Ordenación de oraciones.
- Análisis de ruta crítica.
- Problema de horario del curso.
- Otras aplicaciones como flujos de trabajo de fabricación, serialización de datos y gramática libre de contexto.

2. Usos de la ordenación topológica

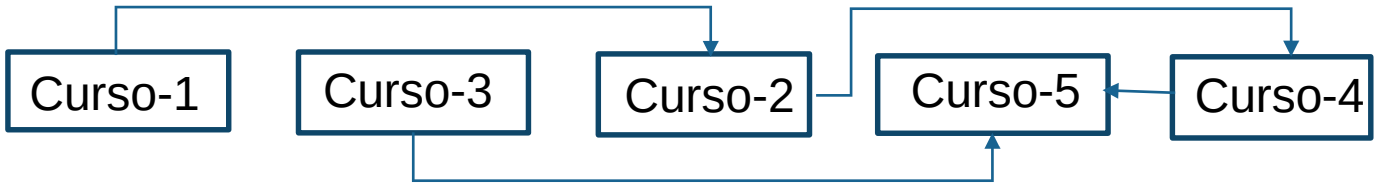
- **Problema de horario del curso**

Existen cursos en una malla curricular y pueden estos tener algunos cursos como requisitos previos.
Uno puede terminar los cursos en algún orden.

Requisitos previos del curso



Una posible ordenación topológica



INVESTIGAR: El **algoritmo de Kahn** permite encontrar una ordenación topológica en un grafo dirigido acíclico (DAG)

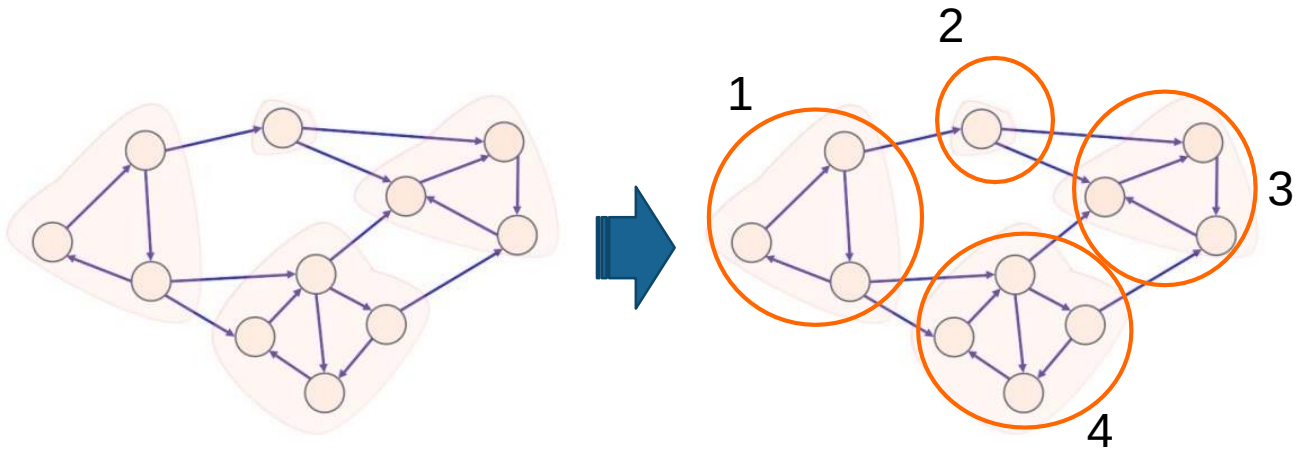
3. Componentes Fuertemente Conexos (SCC)

¿Qué son los Componentes Fuertemente Conexos?

- Llamado por sus siglas en ingles **Strongly Connected Components (SCC).**
- Un grafo dirigido es fuertemente conexo si existe un camino entre todos los pares de vértices.

$$\forall(u, v) \in C : \exists \text{un camino de u a v} \wedge \text{un camino de v a u}$$

- Un componente fuertemente conectado (SCC) de un **grafo dirigido** es un subgrafo máximo fuertemente conectado. Por ejemplo, hay 4 SCC en el siguiente grafo:

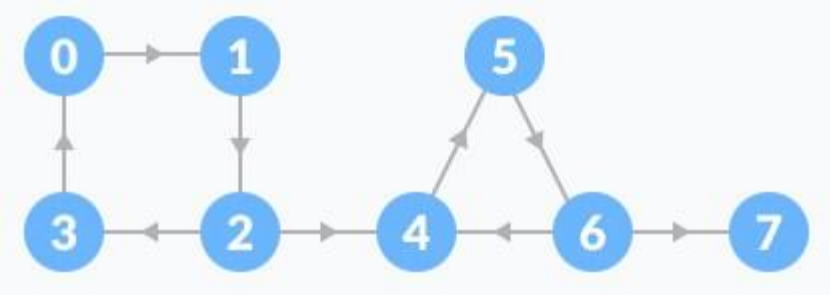


- **NO OLVIDAR:** Es aplicable sólo en un gráfico dirigido.

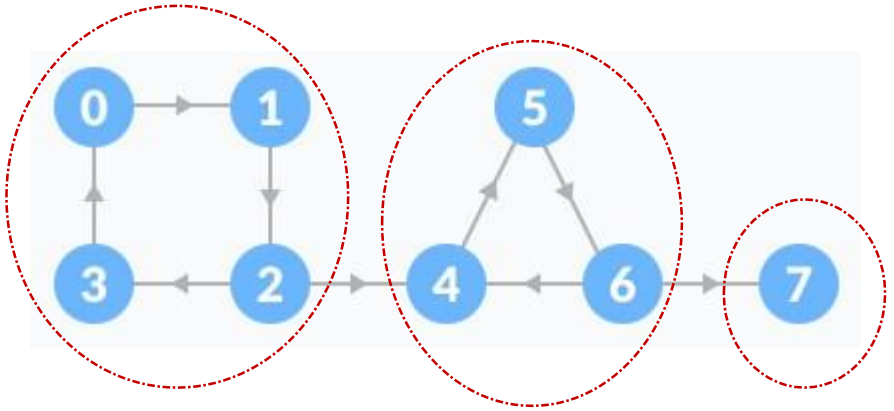
3. Componentes Fuertemente Conexos (SCC)

EJEMPLO

- Tenemos el siguiente grafo inicial:



- Los componentes fuertemente conectados (SCC) del grafo anterior son:



- Se puede observar que en el primer componente fuertemente conectado, cada vértice puede llegar al otro vértice a través del camino dirigido.

- Estos componentes se pueden encontrar utilizando el **Algoritmo de Kosaraju**.

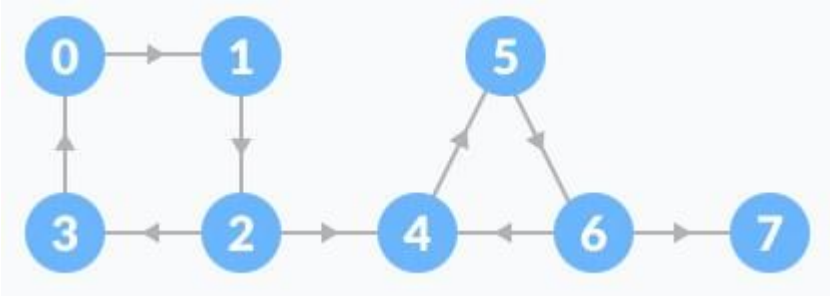
4. Algoritmo de Kosaraju

- El **algoritmo de Kosaraju** se basa en el algoritmo de **búsqueda primero en profundidad** (Deep First Search - DFS), implementado dos veces.

Consta de ejecutar tres (3) pasos:

1. Realizar una búsqueda primero en profundidad (DFS) en todo el grafo.
2. Invertir el grafo original.
3. Realizar otra búsqueda primero en profundidad (DFS) pero ahora sobre el grafo invertido.

Ejecutemos el **algoritmo Kosaraju** sobre el grafo:



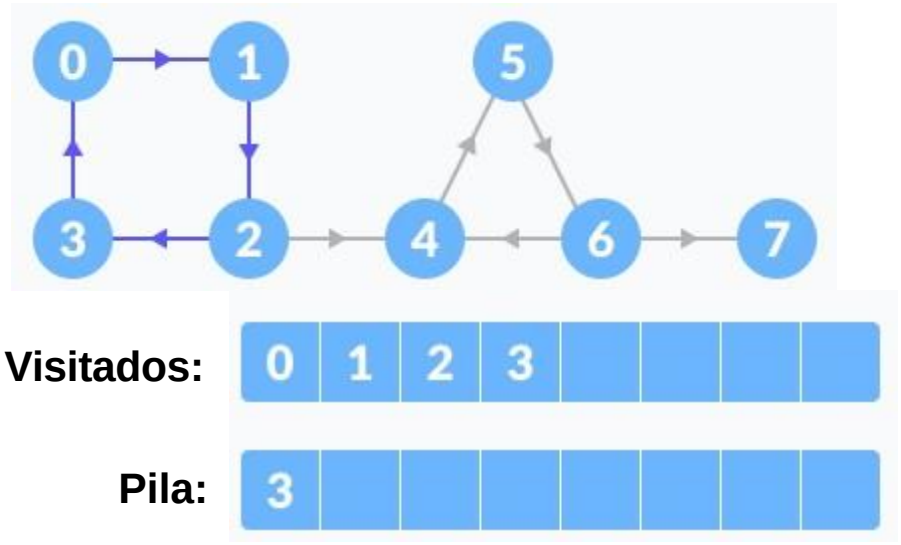
Resultado: debemos encontrar sus componentes fuertemente conectados (SCC)

4. Algoritmo de Kosaraju

Algoritmo de Kosaraju

PASO #1: Realizar una búsqueda primero en profundidad (DFS) en todo el grafo.

a) Comenzamos desde el vértice-0, visitemos todos sus vértices secundarios y los marcamos como visitados.
b) Si un vértice conduce a un vértice ya visitado, colocamos este vértice a la pila.



Por ejemplo:

- A partir del vértice-0, vamos al vértice-1, al vértice-2 y luego al vértice-3.
- El vértice-3 conduce al vértice-0 ya visitado, así que insertamos el vértice de origen (es decir, el vértice-3) en la pila.

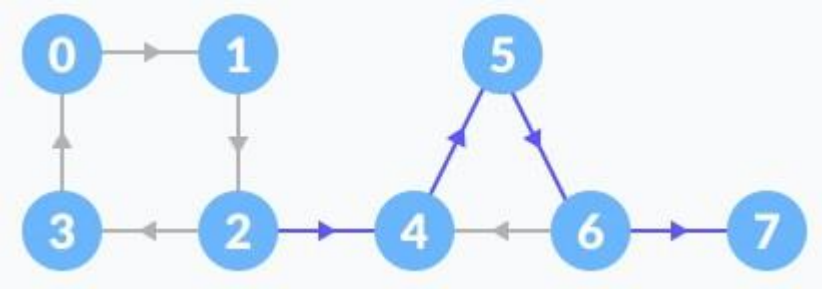
4. Algoritmo de Kosaraju

Algoritmo de Kosaraju

PASO #1: Continuación

c) Vamos al vértice anterior (vértice-2) y visitamos sus vértices secundarios, es decir, vértice-4, vértice-5, vértice-6 y vértice-7 secuencialmente.

d) Dado que no hay adónde ir desde el vértice-7, lo colocamos en la pila.



Visitados:	0	1	2	3	4	5	6	7
Pila:	3	7						

e) Vamos al vértice anterior (vértice-6) y visitamos sus vértices secundarios.

f) Pero se visitan todos sus vértices secundarios, así que los colocamos en la pila.

Visitados:	0	1	2	3	4	5	6	7
Pila:	3	7	6					

De manera similar, se crea una pila final.

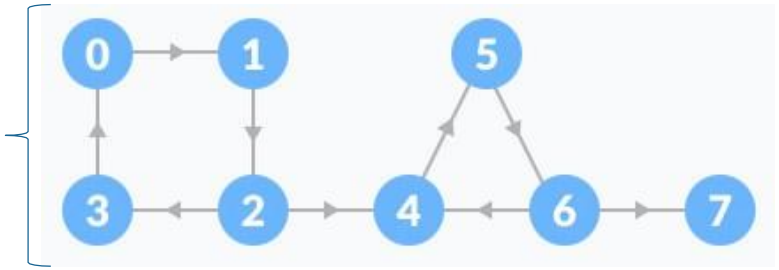
Visitados:	0	1	2	3	4	5	6	7
Pila final:	3	7	6	5	4	2	1	0

4. Algoritmo de Kosaraju

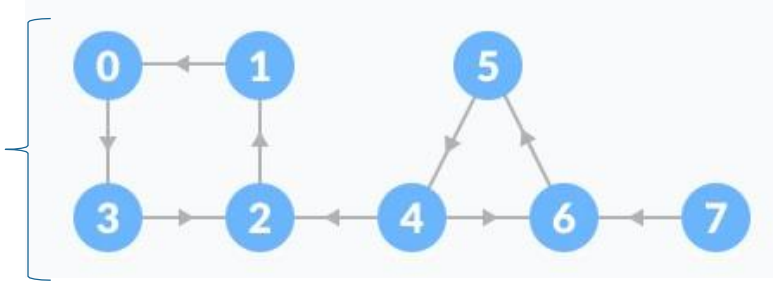
Algoritmo de Kosaraju

PASO #2: Invertir el grafo original.

Grafo Original



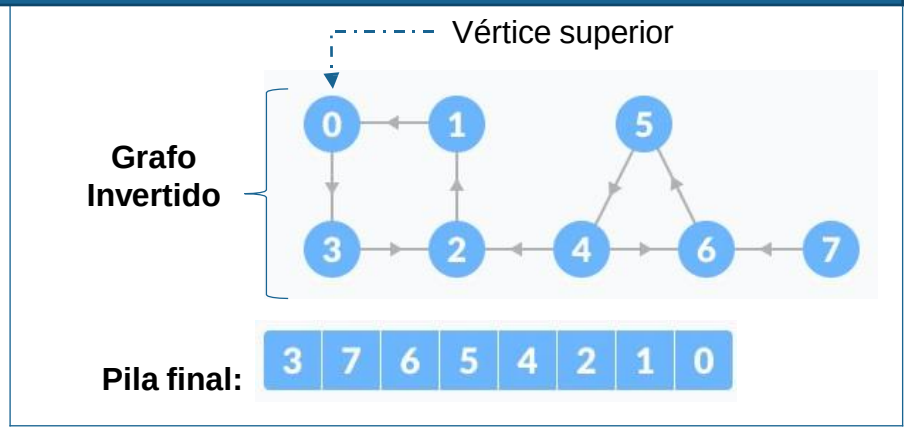
Grafo Invertido



4. Algoritmo de Kosaraju

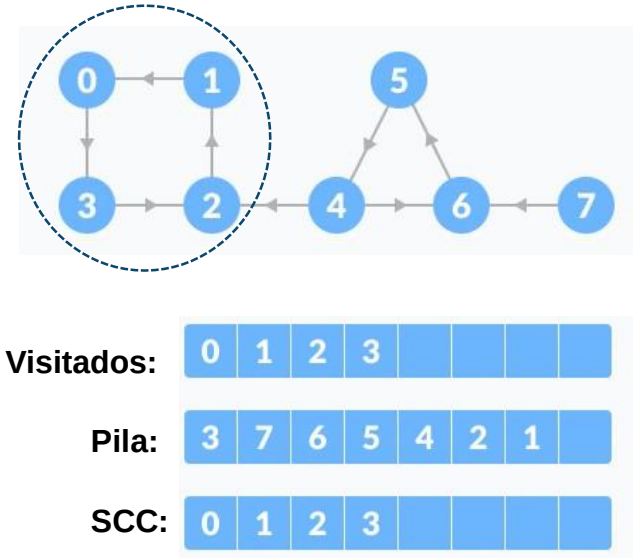
Algoritmo de Kosaraju

PASO #3: Realizar otra búsqueda primero en profundidad (DFS) pero ahora sobre el grafo invertido.



a) Comenzamos desde el vértice superior de la pila. Atravesamos todos sus vértices secundarios.

b) Una vez que se alcanza el vértice ya visitado, se forma un componente fuertemente conectado.



Por ejemplo:

- Extraer el vértice-0 de la pila final del paso #2
- A partir del vértice-0, recorrer sus vértices secundarios (vértice-0, vértice-1, vértice-2, vértice-3 en secuencia) y marcarlos como visitados.
- El hijo del vértice-3 ya está visitado, por lo que estos vértices visitados forman un componente fuertemente conectado.

4. Algoritmo de Kosaraju

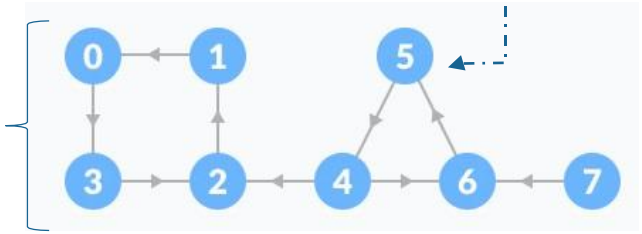
Algoritmo de Kosaraju

PASO #3: Continuación

c) Ir a la pila y extraer el vértice superior si ya se visitó. De lo contrario, elegir el vértice superior de la pila y recorrer sus vértices secundarios como se muestra en el paso anterior.

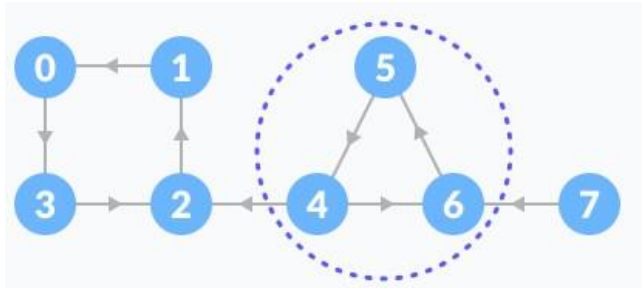
Vértice superior

Grafo Invertido



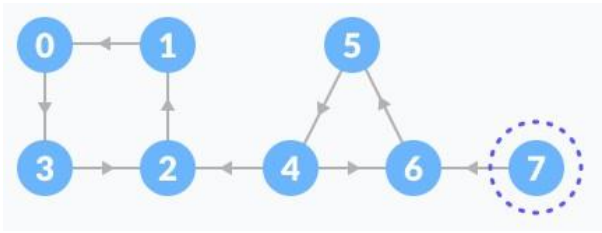
Pila final del paso #2:

3	7	6	5	4	2	1	0



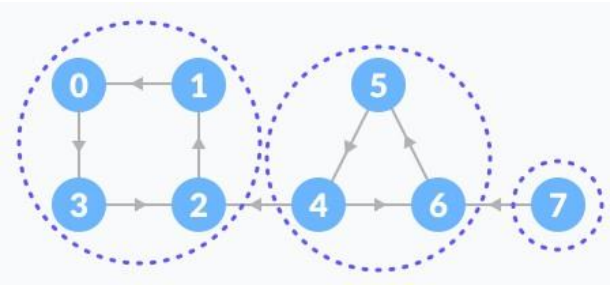
Visitados:	0	1	2	3	4	5	6	
Pila:	3	7	6	5				
SCC:	4	5	6					

d) Componente fuertemente conectado



Visitados:	0	1	2	3	4	5	6	
Pila:								
SCC:	7							

Todos los componentes fuertemente conectados



PREGUNTAS

Dudas y opiniones