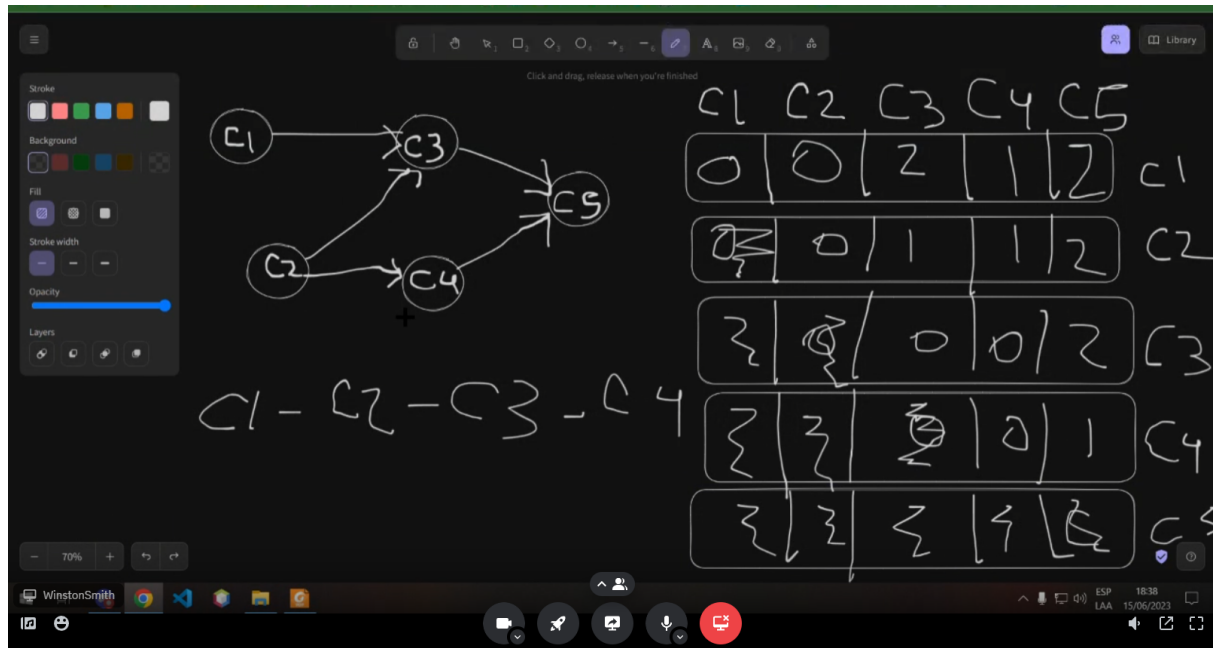


## Sort topologico metodo 1:

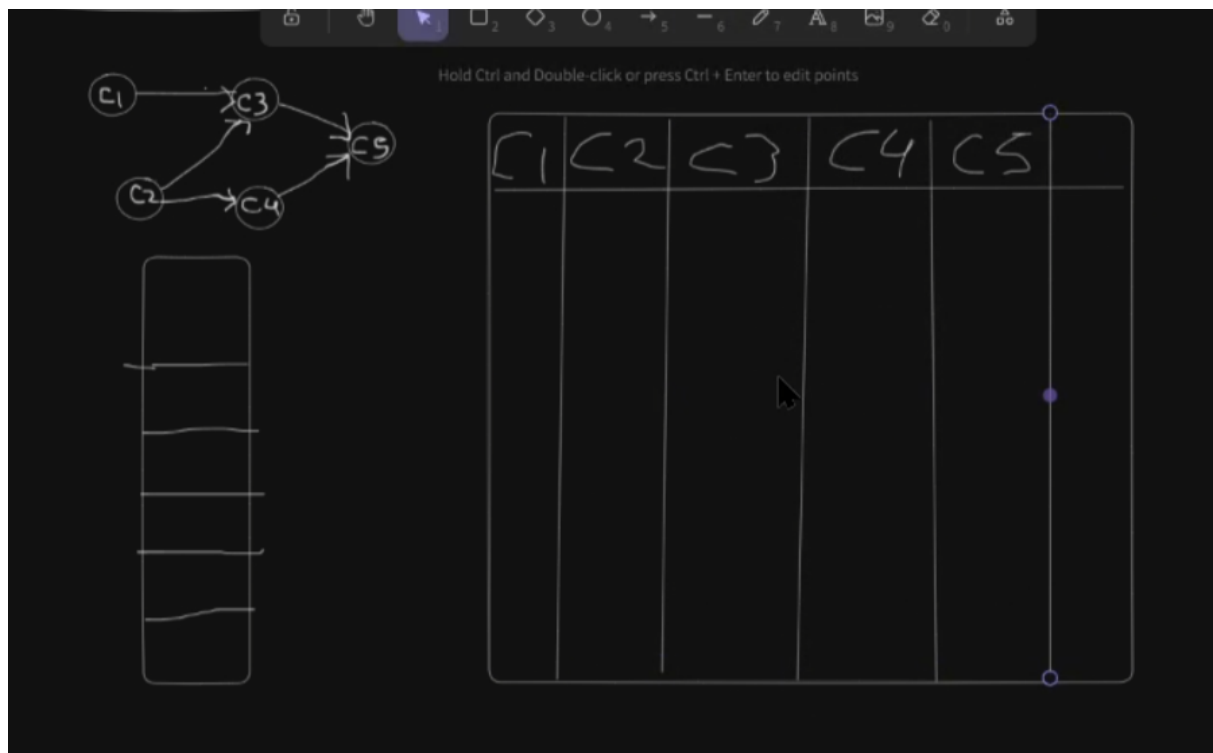
Agarras el grafo y haces un array con los grados de entrada.

Una vez tienes el array tomas los que tengan grado 0 y bajas en uno el grado de entrada a los que tiene de hijos. Repetir y armas tu cadena de correlativos.

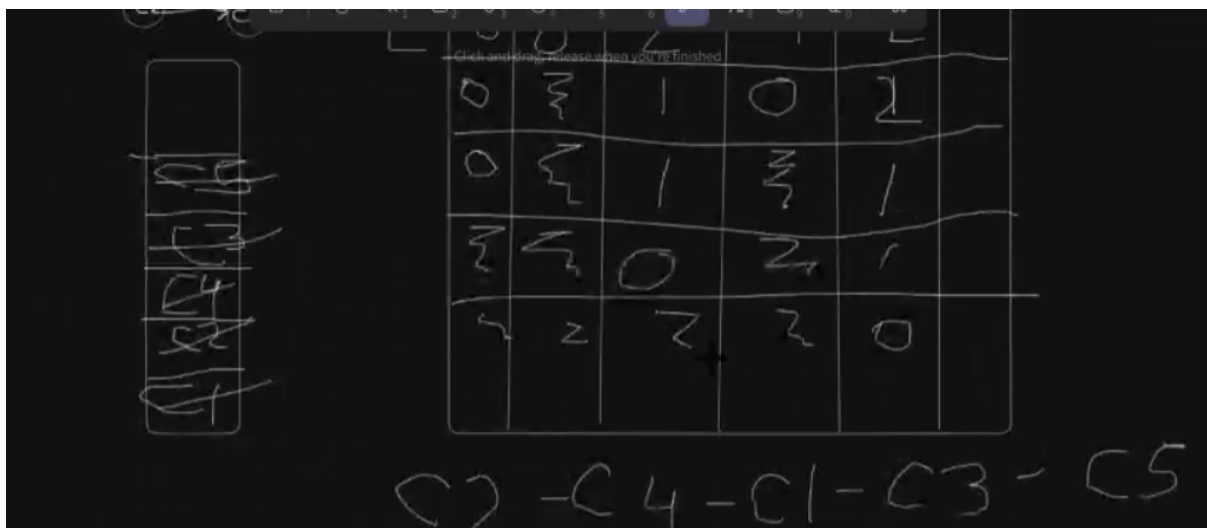
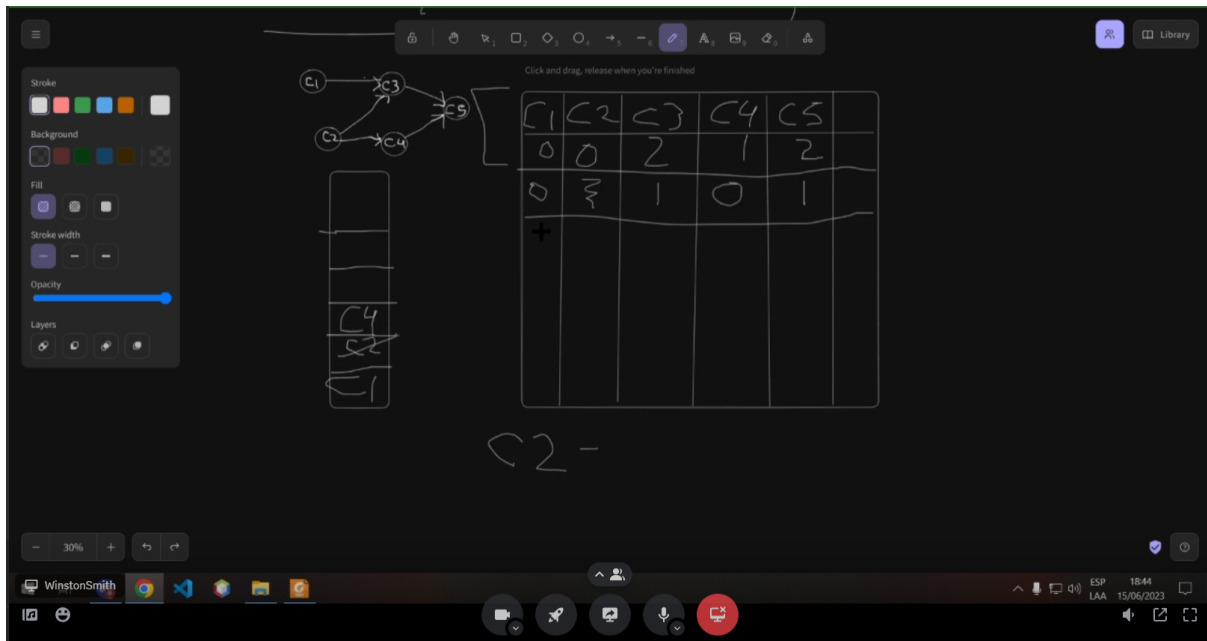


## Sort topologico metodo 2:

Este es con una pila.



Hacemos como el otro, apilamos los de orden de entrada 0 (c1, c2), desapilas c2, y bajas el grado de los siguientes y marcas como usado c2.

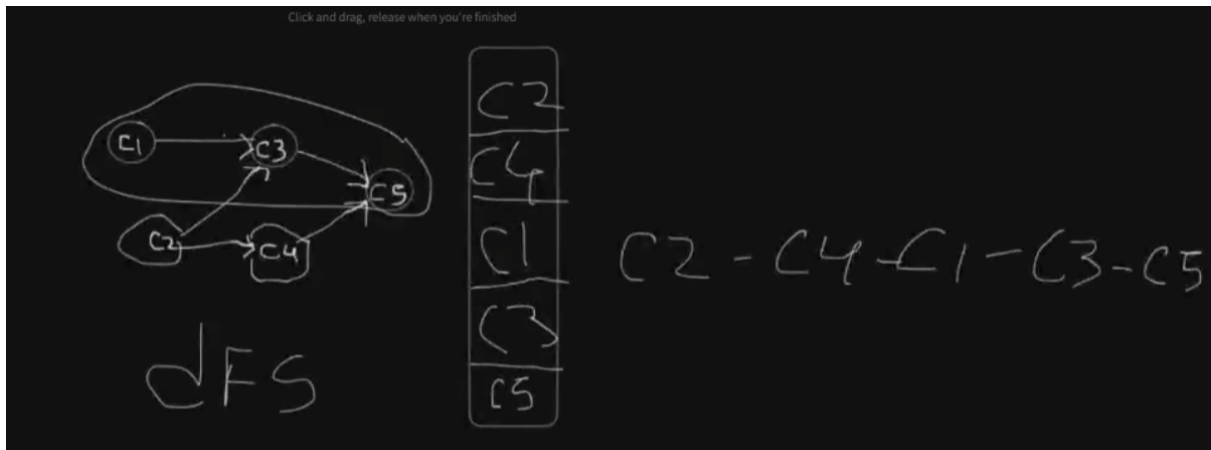


### Sort topologico metodo 3:

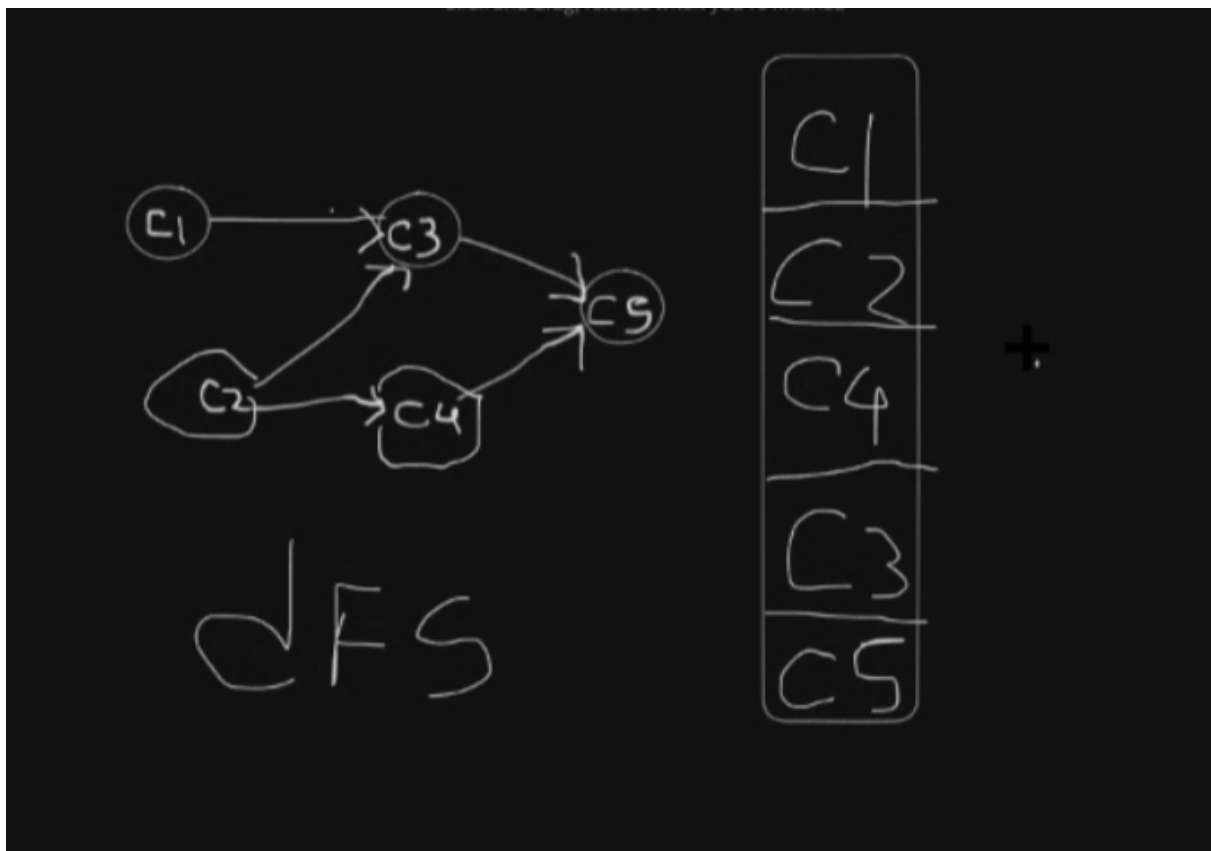
Es un dfs, tiene 2 formas de hacerlo, una es medio rara así que vamos con la piola.

Usamos una pila pero sin tabla, seleccionamos cualquiera de grado 0 (o te dicen de cual iniciar), arrancamos en c1, llamas a su adyacente (c3), llamas al adyacente del siguiente y llegamos a c5 y lo apilamos, volvemos y y ahí apilamos c3, y después apilamos c1. Así funciona.

Despues el orden es ir desapilandolo.



Arrancando de c2:

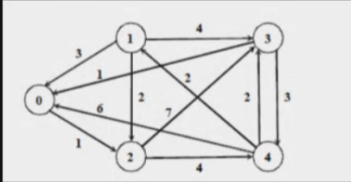


### Dijkstra:

Camino más corto desde un punto de salida a todo el resto.

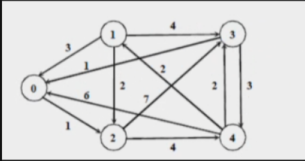
No le calienta los grados de entrada, ya que son los caminos más cortos:

Arrancamos del 3:



Iter	Ver	distancia	anterior	Vis
	0	$\infty$		
	1	$\infty$		
	2	$\infty$		
1	3	0		
	4	$\infty$		

Marcamos todas las aristas que inciden, para ir al 4 tenemos peso 3 y entonces actualizamos, lo mismo con el 0.



Iter	Ver	distancia	anterior	Vis
	0	<del><math>\infty</math></del> 1	3	
	1	$\infty$		
	2	$\infty$		
1	3	0		1
	4	<del><math>\infty</math></del> 3	3	

Ahora partimos del que tiene menor valor en la tabla (0, tiene peso 1).

Iter	Ver	distancia	anterior	Vis
2	0	<del>1</del>	3	↓
	1	$\infty$	+	
	2	<del>2</del>	0	
1	3	0		↓
	4	<del>3</del>	3	

Ahora que estamos en el 2, vamos acumulando los pesos como vemos (peso de ir a 0 + peso de ir al 2).

Estando en el 0, nos damos cuenta que podemos ir al 4, como la suma de los pesos es 6, y el peso anterior era 3, no lo actualizamos.

Nos metemos en el 4 y ahí nos podemos meter al 1.

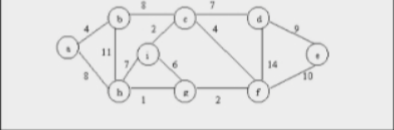
Iter	Ver	distancia	anterior	Vis
2	0	<del>1</del>	3	↓
5	1	<del>5</del>	4	↓
3	2	<del>2</del>	0	↓
1	3	0		↓
4	4	<del>3</del>	3	↓

Obviamente cuando hay pesos negativos usas una cola y no hay que marcar los conocidos.

## PRIM:

Se basa en tocar todos los vértices, pero con el menor costo posible. Por ejemplo, prácticamente sería como intentar distribuir internet, tienes que hacer un camino que se asegura de llegar a todas con un costo mínimo.

Similar al dijkstra pero sin ir acumulando los pesos. Arrancas desde el que sea y tomamos la arista con menor peso. Vas al que sea y en la tablita vas poniendo el peso y de cual viniste. Vas al siguiente con menor valor en la tabla; si el siguiente va a otro que ya visitaste solo lo actualizas si tu peso es menor que el otro.



Iter	Letido	Distancia	Anterior	Vis
1	A	0	—	1
2	B	4	A	1
3	C	8	B	1
4	D	7	C	1
5	E	<del>9</del>	KE	1
6	F	4	C	1
7	G	<del>2</del>	KE	1
8	H	<del>1</del>	KE	1
9	I	2	C	1

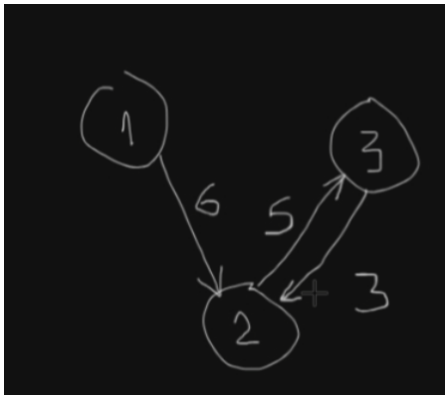
Y después puede que te pidan hacer el árbol. Básicamente es seguir las iteraciones y fijar el anterior.

### Floyd:

sirve para hallar la mínima distancia de un nodo a otro y saber por cual pasar.

V	1	2	3
1	0		
2		0	
3			0

V	1	2	3
1	-		
2		-	
3			-



Arrancamos con esa tabla, lo del medio siempre es 0 porque no hay otra forma de mejorarlo. (Si fuera dirigido tendrías espejado la tabla por la diagonal mayor).  
(EFFECTIVAMENTE IR A SI MISMO NO TIENE RECORRIDO).  
Completamos los datos con lo que tenemos en el grafo.

DIST

V	1	2	3
1	0	6	
2	$\infty$	0	5
3	$\infty$	3	0

REC

V	1	2	3
1	-	2	3
2	1	-	3
3	1	2	-

(Ponemos la distancia para ir de cual a cual, y después en recorridos ponemos lo que está en el índice de la columna).

Ahora lo que tenemos que hacer es evaluar las posiciones cruzadas. Es decir evaluamos lo del 1,2 y el 2,1 y si la sumatoria de eso da menos de lo que está en en 2,2 entonces reemplazamos, en este caso al haber infinitos no reemplazamos.

DIST

V	1	2	3
1	0	6	$\infty$
2	$\infty$	0	5
3	$\infty$	3	0

REC

V	1	2	3
1	-	2	3
2	1	-	3
3	1	2	-

Ahora evaluamos esta cruz:

DIST

V	1	2	3
1	0	6	∞
2	∞	0	5
3	∞	3	0

REC

V	1	2	3
1	-	2	3
2	1	-	3
3	1	2	-

Grosor del trazo

Opacidad

Capas

DIST

	2	3
2	6	11
3	0	5
3	3	0

REC

V	1	2	3
1	-	2	<del>3</del> 2
2	1	-	3
3	1	2	-

$[6+5 < \infty \Rightarrow (1,3)=11] \parallel (1,3)=2$

Iteración 3:

DIST

V	1	2	3
1	0	6	11
2	∞	0	5
3	∞	3	0

REC

V	1	2	3
1	-	2	2
2	1	-	3
3	1	2	-

No podemos cambiar nada.

Hay que dibujar una iteración por cada vértice.