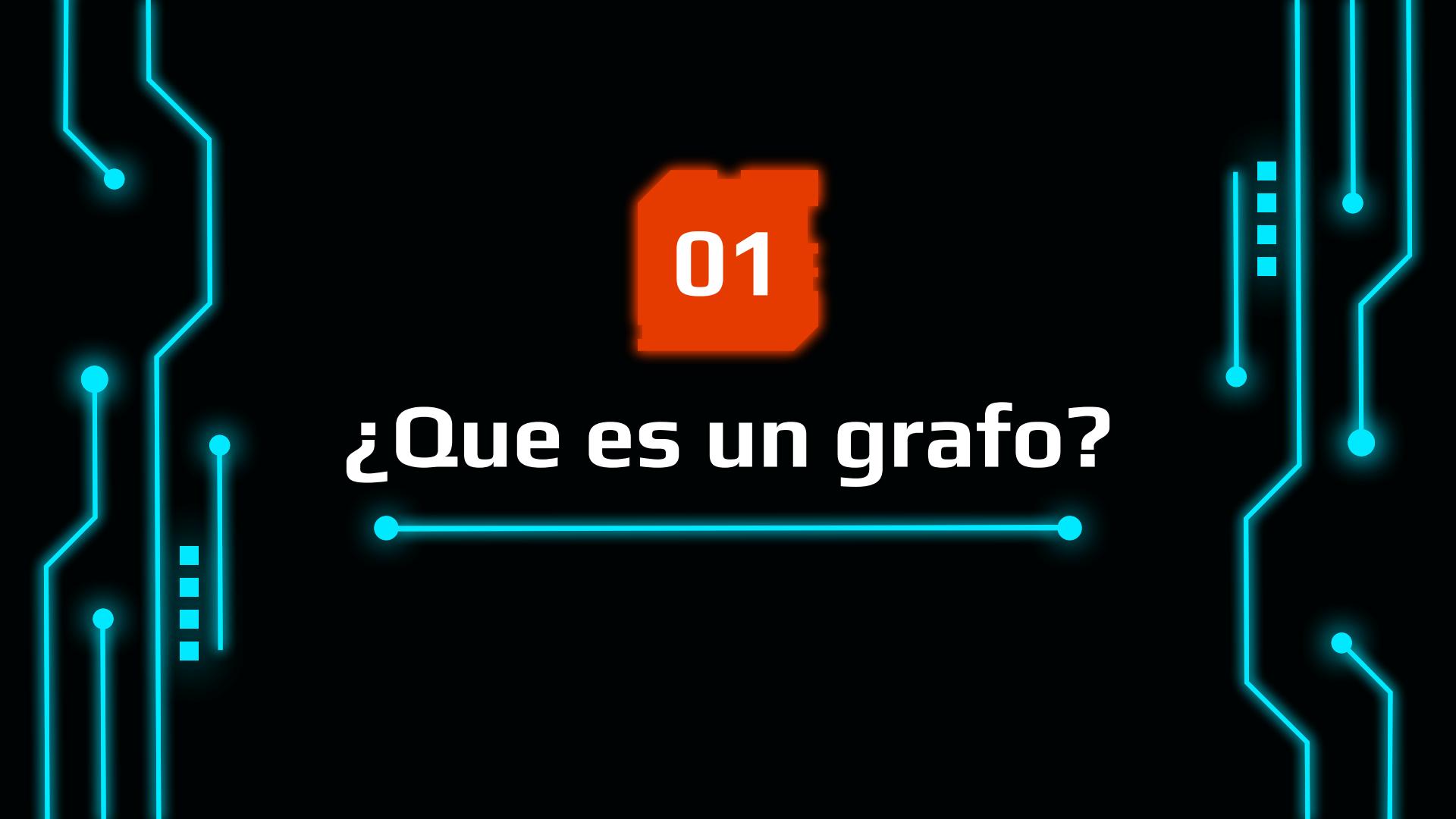


Introducción a la Teoría de Grafos

(Teoría de gráficas si son de la gente extraña de
la UNAM xD)

Developed by Erick Ignacio Santillan Zaragoza
(Santirax para los compas)



01

¿Que es un grafo?

El Profe:



01

Los grafos son estructuras generalizadas que aparecen en muchos problemas de Ciencias de la Computación. Un grafo ($G = (V,E)$) en su forma básica es simplemente un conjunto de vértices (V) y aristas (E ; almacenando información de conectividad entre vértices en V).



—Mi traducción del Competitive Programming 3 :P

Definición



Una *gráfica* (o *gráfica no dirigida*) G consiste en un conjunto V de vértices (o nodos) y un conjunto E de *aristas* (o *arcos*) tal que cada arista $e \in E$ se asocia con un par no ordenado de vértices. Si existe una arista única e asociada con los vértices u y w , se escribe $e = (v, w)$ o $e = (w, v)$. En este contexto, (v, w) denota una arista entre v y w en una gráfica no dirigida y *no* es un par ordenado.

Definición



Una *gráfica dirigida* (o *digráfica*) G consiste en un conjunto V de vértices (o nodos) y un conjunto E de aristas (o arcos) tales que cada arista $e \in E$ está asociada con un par ordenado de vértices. Si hay una arista única e asociada con el par ordenado (v, w) de vértices, se escribe $e = (v, w)$, que denota una arista de v a w .

Definición



Se dice que una arista e en una gráfica (no dirigida o dirigida) que se asocia con el par de vértices v y w es *incidente sobre* v y w , y se dice que v y w son *incidentes sobre* e y son *vértices adyacentes*.

Si G es una gráfica (no dirigida o dirigida) con vértices V y aristas E , se escribe $G = (V, E)$.

A menos que se especifique lo contrario, se supone que los conjuntos E y V son finitos y que V es no vacío.

NOTAS IMPORTANTES

1. A los vértices los vamos a conocer como **NODOS**.
2. El nombre de aristas lo vamos a conservar.
3. El que diga gráficas en lugar de grafos, lo saco del server (solo la gente rara les dice asi).



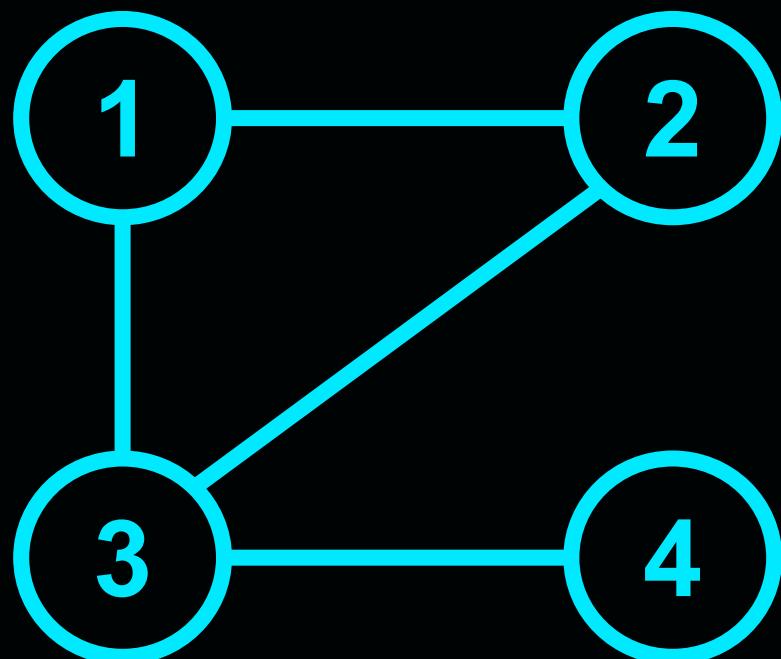


Y ahora unos
ejemplos...

G = (V, E) Donde:

$$V = \{1, 2, 3, 4\}$$
$$E = \{(1, 2), (1, 3), (2, 3), (3, 4)\}$$

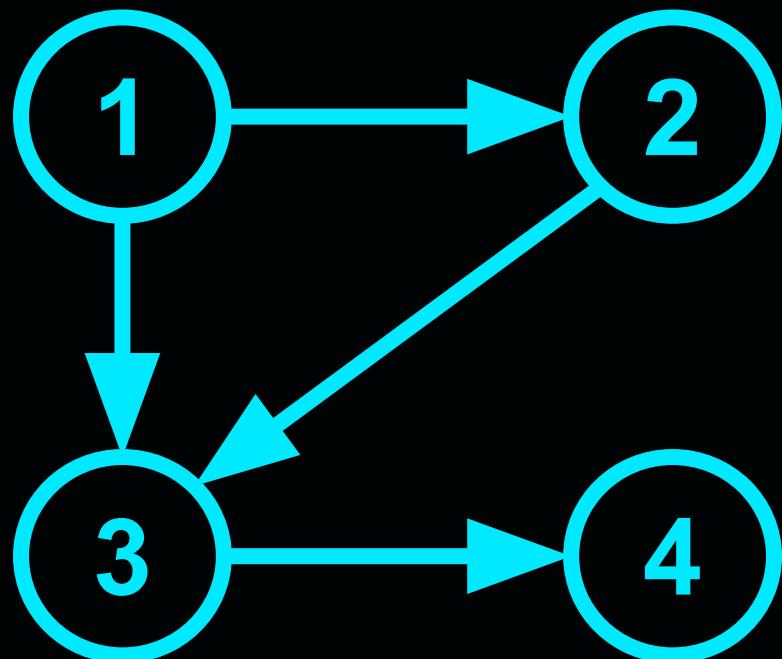
Esto es un grafo no dirigido, es decir, podemos ir de un nodo a otro y de regreso. Por ejemplo podemos ir del nodo 1 al nodo 3 y regresar (ir del nodo 3 al nodo 1).

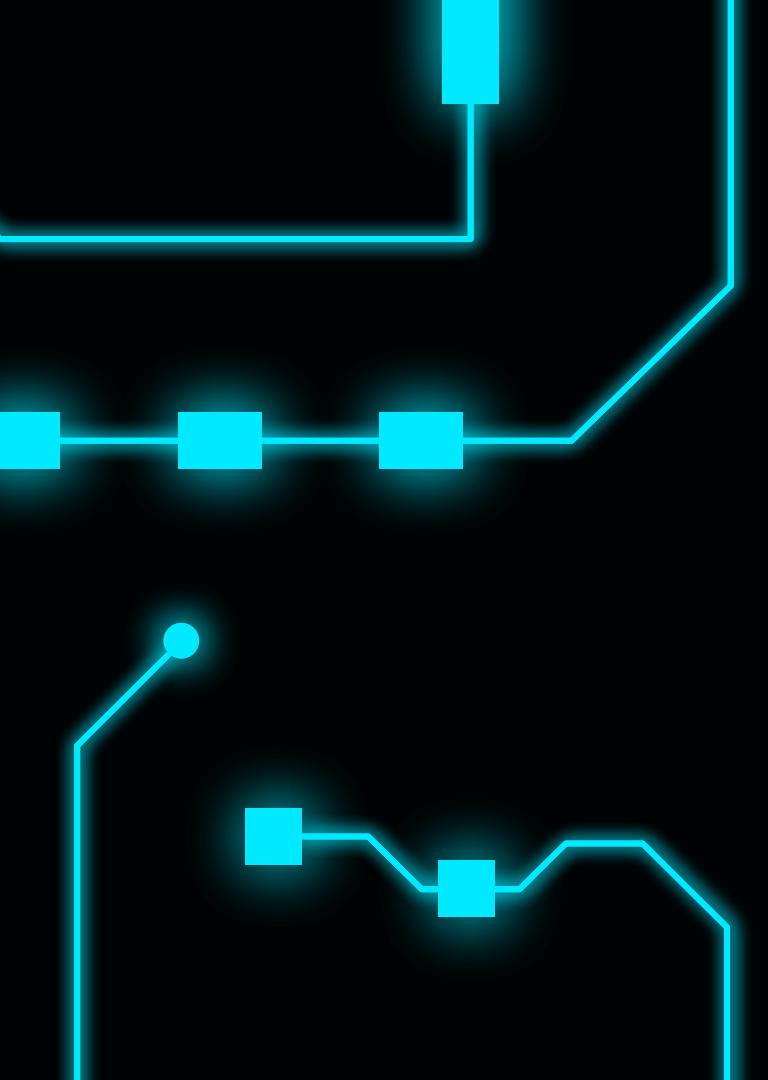


G = (V, E) Donde:

$$V = \{1, 2, 3, 4\}$$
$$E = \{(1, 2), (1, 3), (2, 3), (3, 4)\}$$

Esto es un grafo dirigido, es decir, podemos ir de un nodo a otro pero no de regreso, a menos de que exista la arista para poder regresar. Por ejemplo podemos ir del nodo 1 al nodo 3 pero no de regreso.





WHOA!



Hasta aqui.
Alguna duda?



Todo muy padre pero...

**¿Cómo puedo programar
un grafo?**

Cuando te piden el código...



Mi trabajo aquí ha terminado pero
el suyo acaba de empezar

Las 3 formas básicas de guardar un grafo según el Competitive Programming 3



Matriz de adyacencia

Earth is the third planet from the Sun and the only one that harbors life



Lista de adyacencia

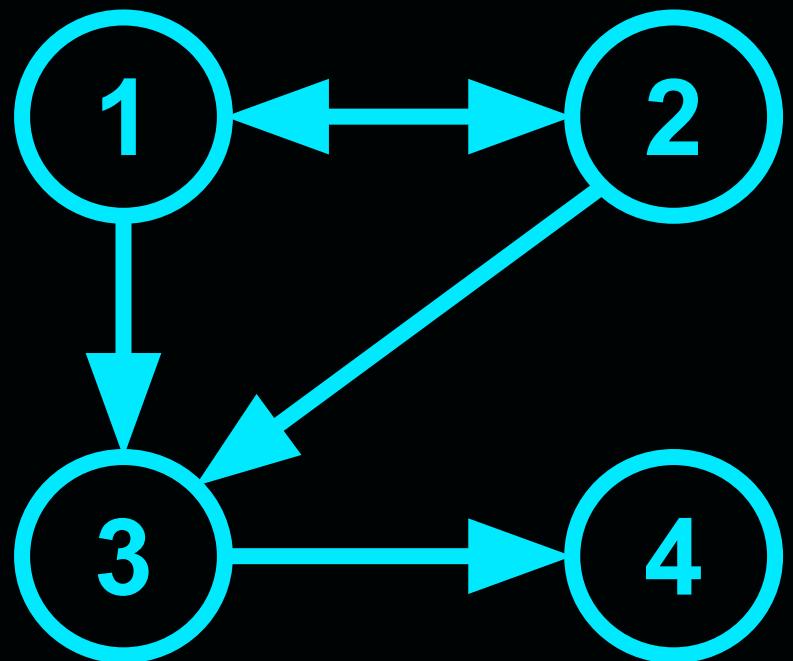
Despite being red, Mars is actually a cold place. It's full of iron oxide dust



Lista de aristas

Mercury is the closest planet to the Sun and the smallest one in the System

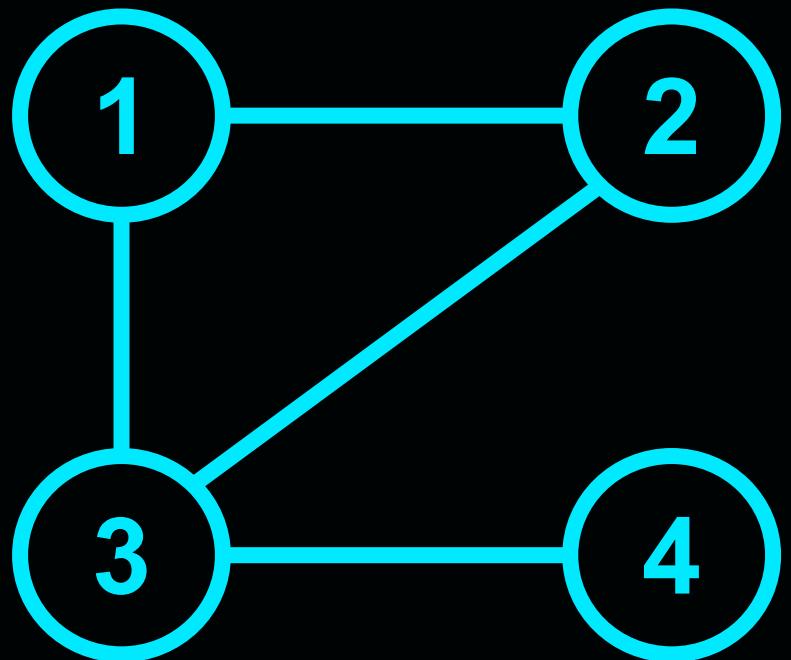
Matriz de adyacencia.



Vamos a hacer una matriz de NxN donde N es el número de nodos

	1	2	3	4
1	0	1	1	0
2	1	0	1	0
3	0	0	0	1
4	0	0	0	0

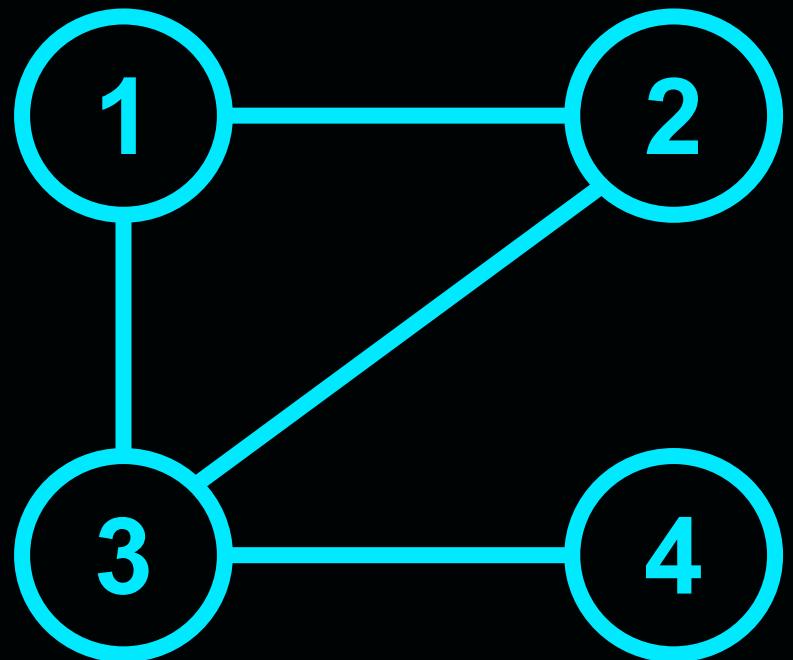
Lista de adyacencia.



Vamos a hacer una lista de size N
donde N es el número de nodos

Nodos	Lista de adyacencia
1	2, 3
2	1, 3
3	1, 2, 4
4	3

Lista de aristas.



Vamos a hacer una lista de size M donde M es el número de aristas.

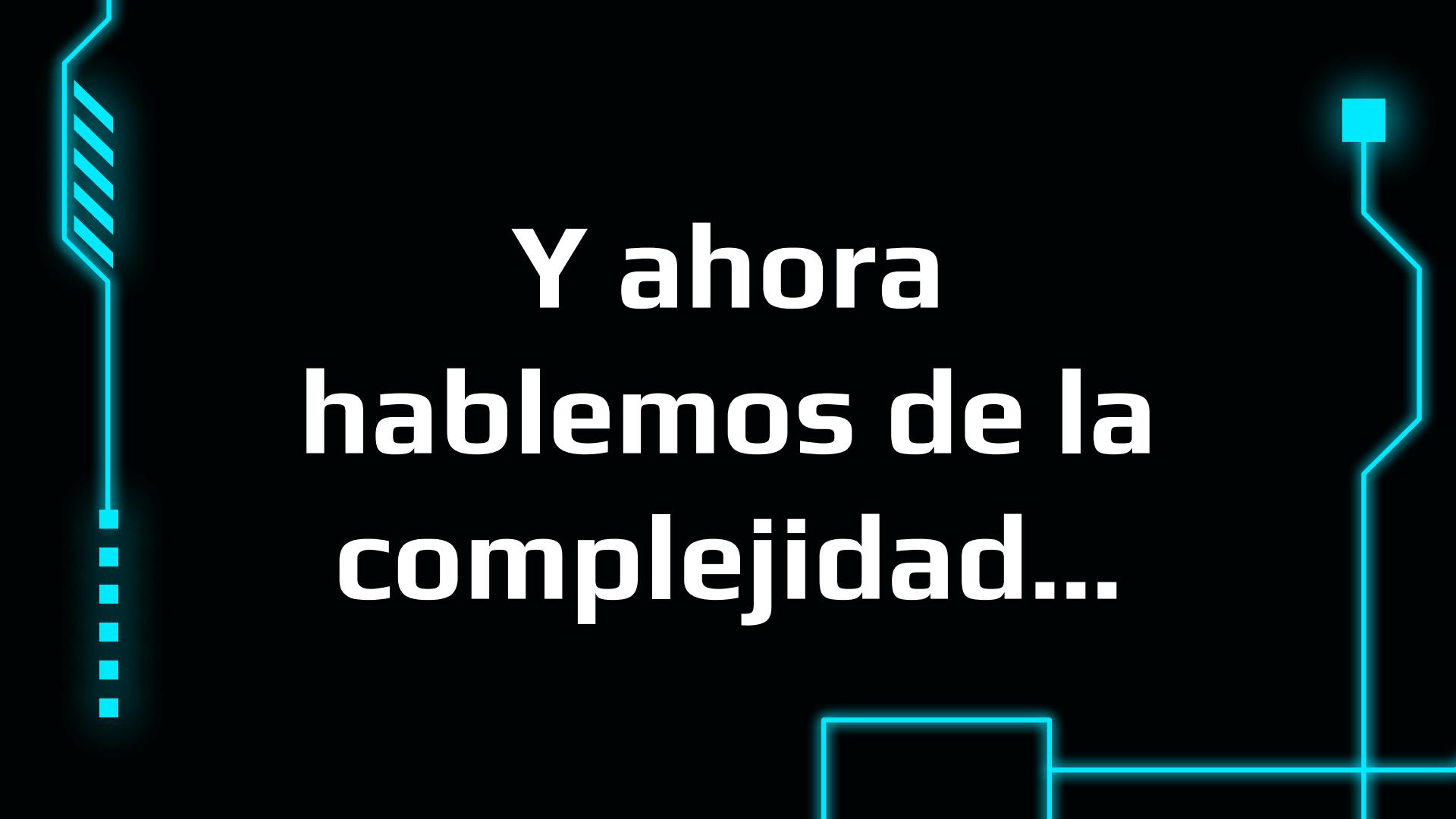
Lista de Aristas

(1,2), (2,1)

(1,3), (3,1)

(2,3), (3,2)

(3,4), (4,3)

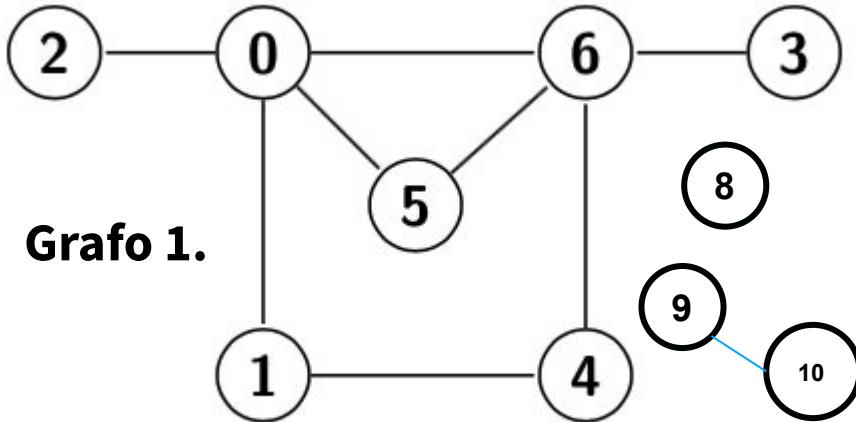


Y ahora
hablemos de la
complejidad...

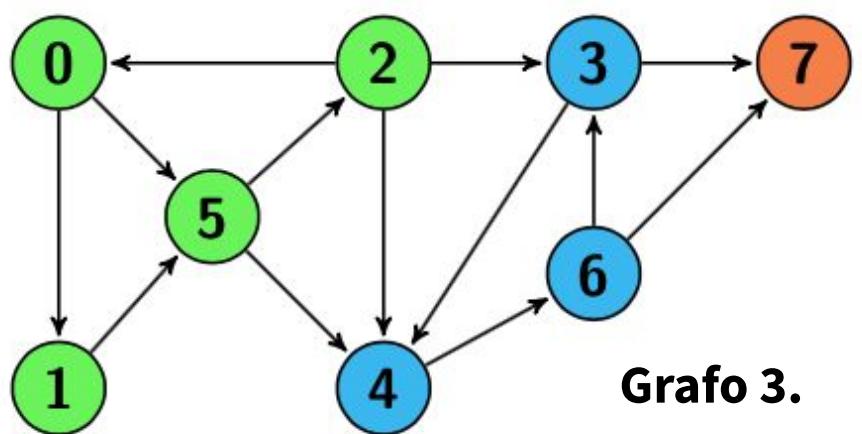
	Tiempo	Espacio/Memoria
Matriz de adyacencia	$O(N^2)$	$O(N^2)$
Lista de adyacencia	$O(N+M)$	$O(N+M)$
Lista de aristas	$O(M)$	$O(M)$

Ahora si banda, ya se
la saben, a
programar...

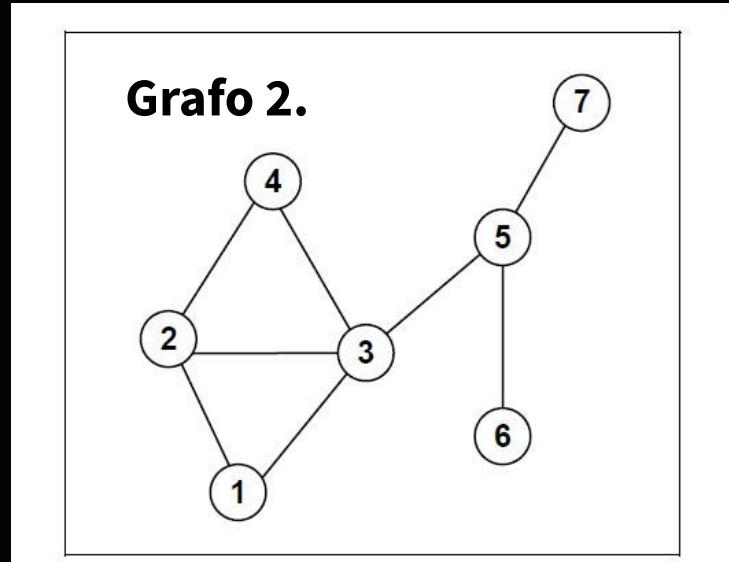




Grafo 1.



Grafo 3.



Grafo 2.



Necesito el código

- El Jeilo verde xD



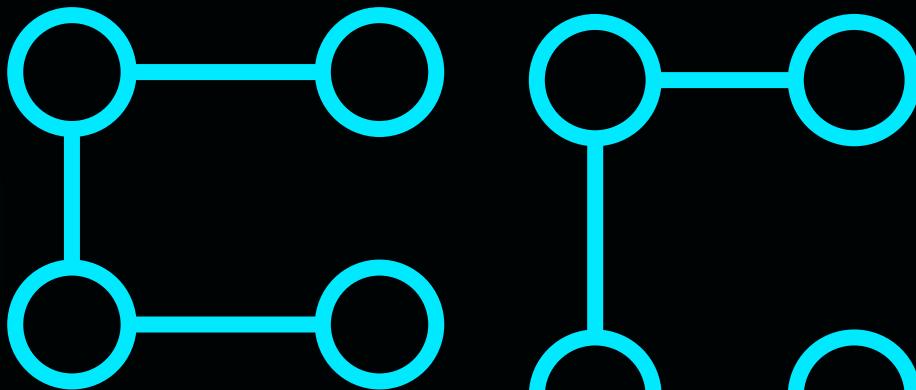
* Yo sin saber porque no me
entienden

* Los de básicos esperando
a que comparta pantalla

Nos pasamos al editor de código.

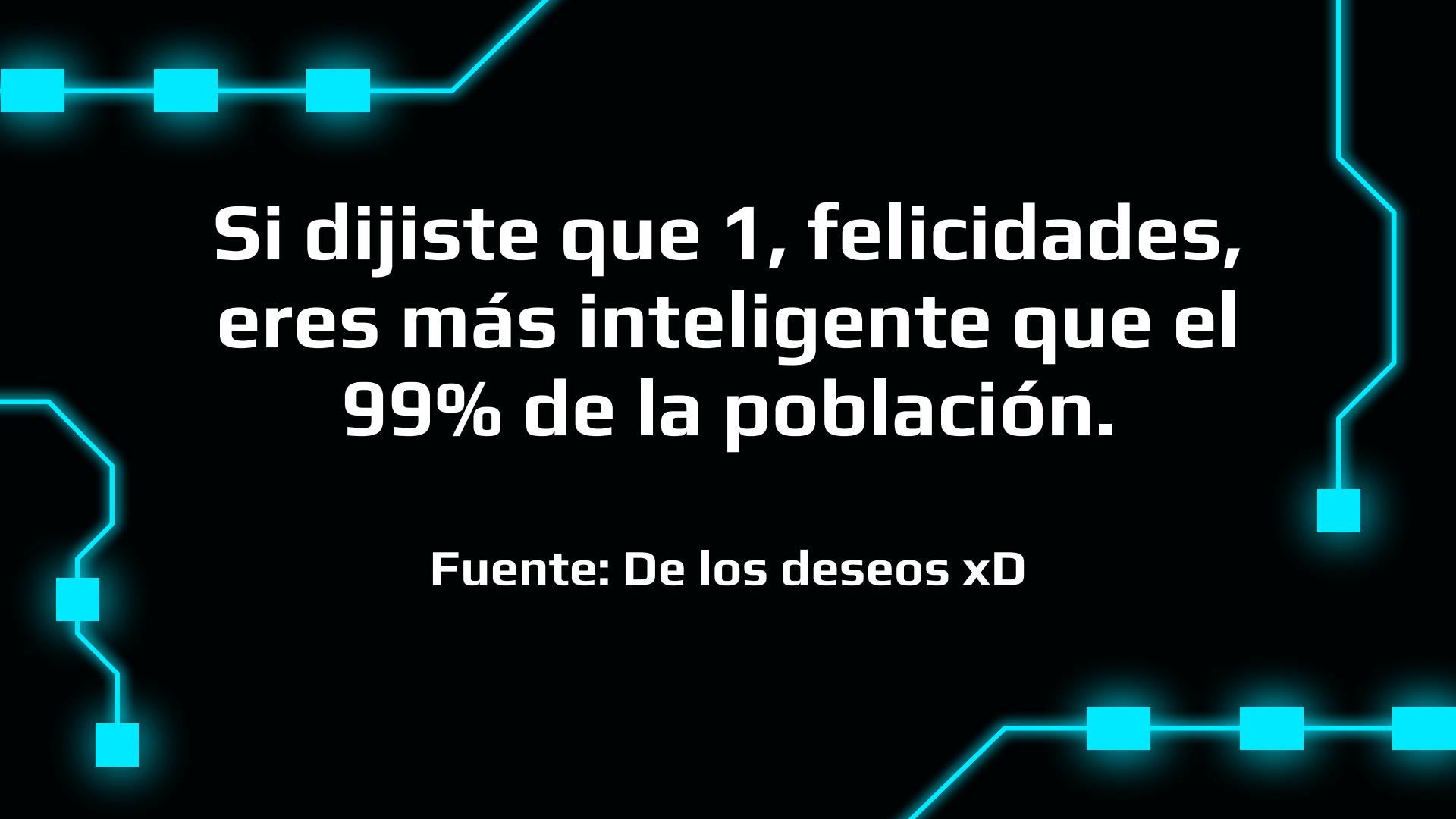
Sigamos con la componentes conexas.

Tenemos lo siguiente:



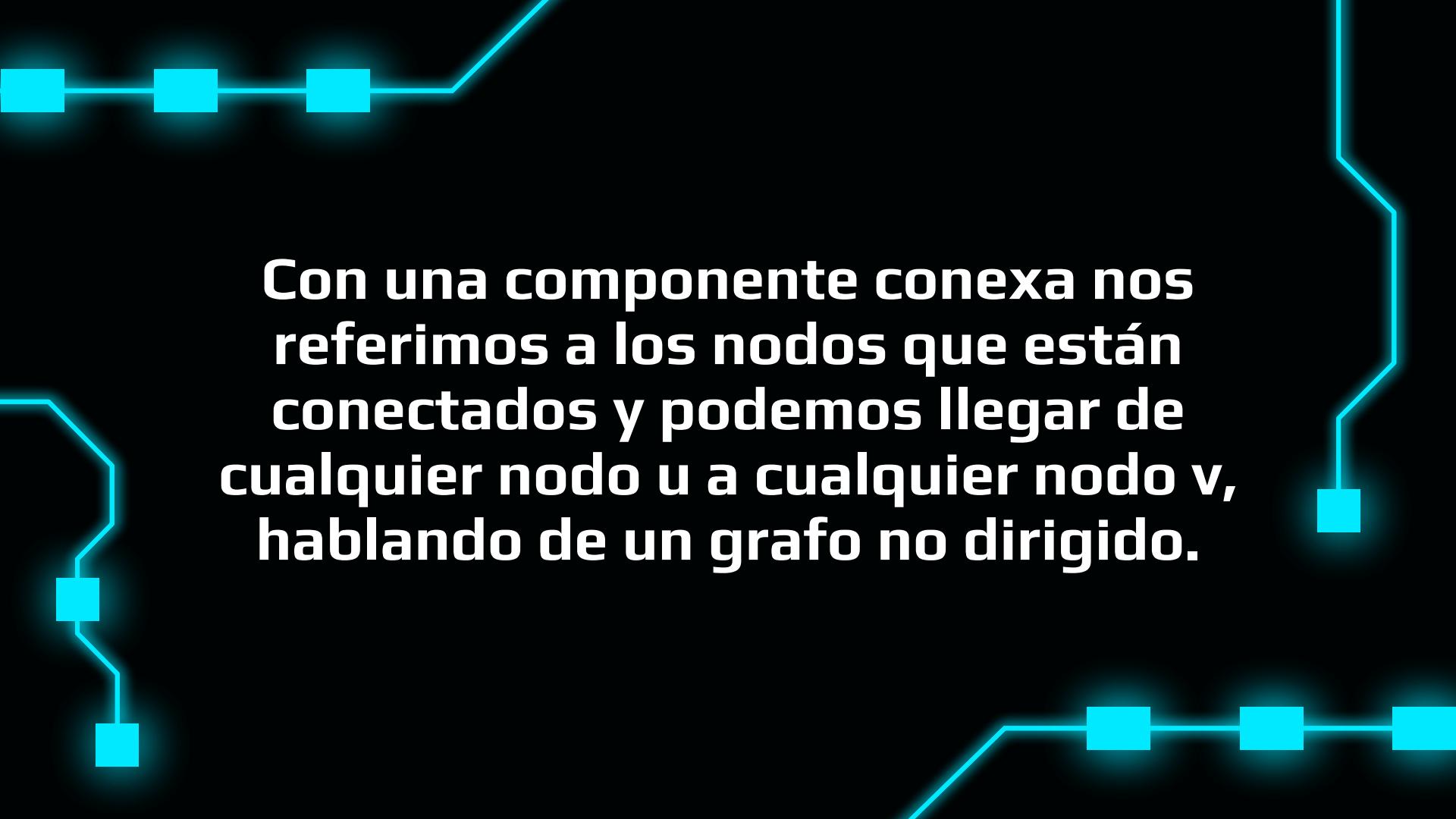
La pregunta es,
¿Cuántos grafos
tenemos?

(El 99% de la población falla este test :O)



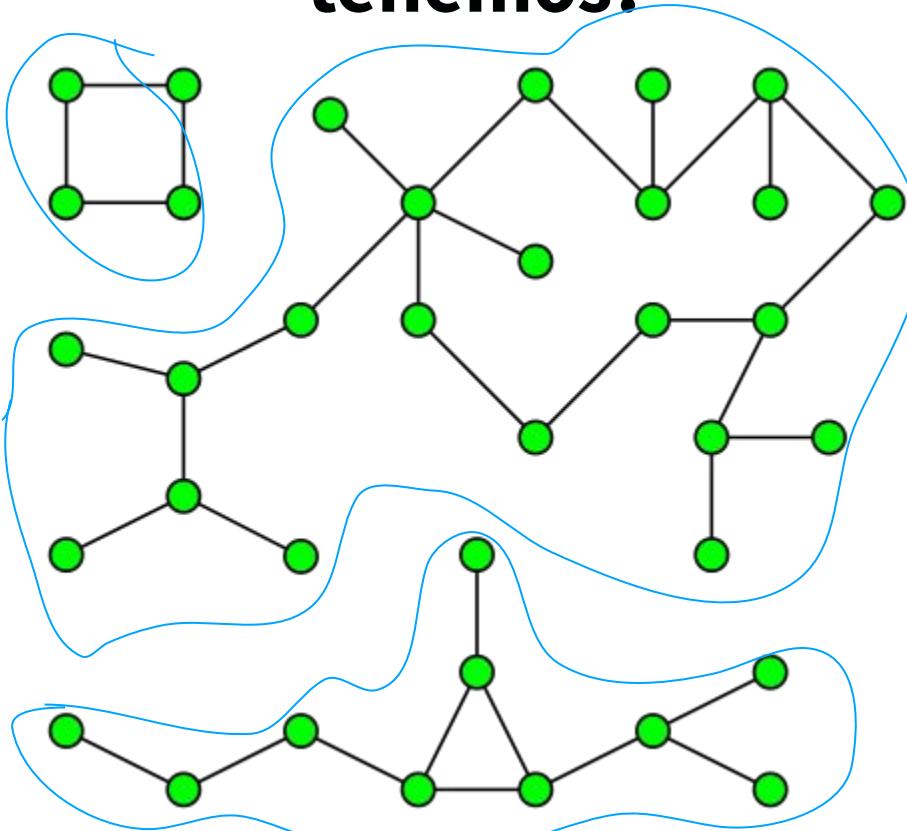
**Si dijiste que 1, felicidades,
eres más inteligente que el
99% de la población.**

Fuente: De los deseos xD



Con una componente conexa nos referimos a los nodos que están conectados y podemos llegar de cualquier nodo u a cualquier nodo v, hablando de un grafo no dirigido.

¿Cuántas componentes tenemos?



¿Cómo recorrer un grafo?



01

DFS

Depth First Search.
Búsqueda por profundidad

02

BFS

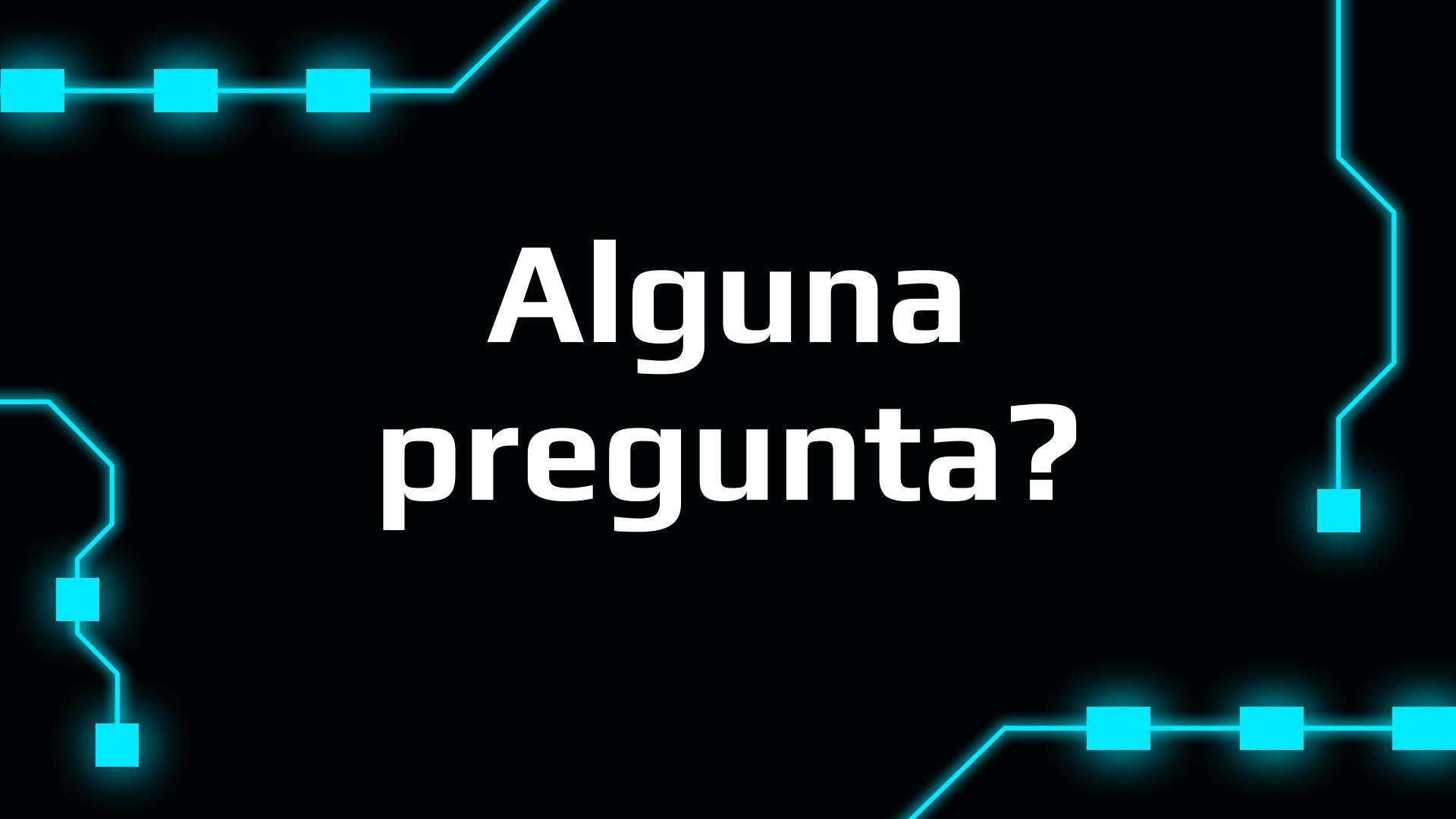
Breadth First Search
Búsqueda en anchura



DFS

Depth First Search

Y para explicar este tema.
Vamos al pizarrón.



Alguna
pregunta?

BFS

Breadth First Search

Y para explicar este tema.
Vamos al pizarrón. Otra vez.

Coming up next...

Un tipo especial de grafo

Los arboles

Clase que esperamos, va a ser
impartida por el Profe Norman

Amonos con problemas...

Problema 1.

1

Problema 2.

2

Problema 3.

3

O la caja
misteriosa...



Dudas, preguntas, traumas, comentarios, observaciones?



Gracias por su atención.

Se aceptan aplausos, elogios, ovaciones,
transferencias, paypal, donaciones,
mercado pago, ...

Es broma pero si quieren
no es broma xD



REFERENCES

- Competitive Programming 3
- Introduction to Algorithms, Cormen