Ewan TALEC

TP1 - Algoritmos de búsqueda

Según el juego Grados de separación de Kevin Bacon.

Problem

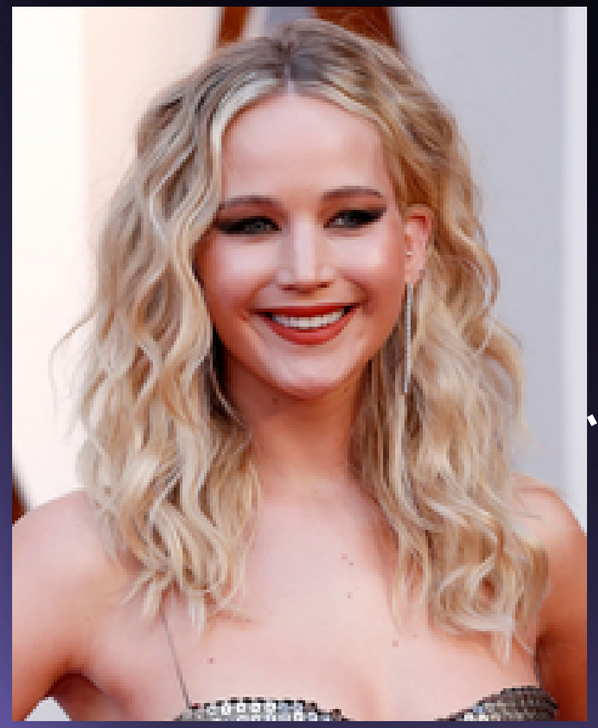
Definir el principio del problema

01

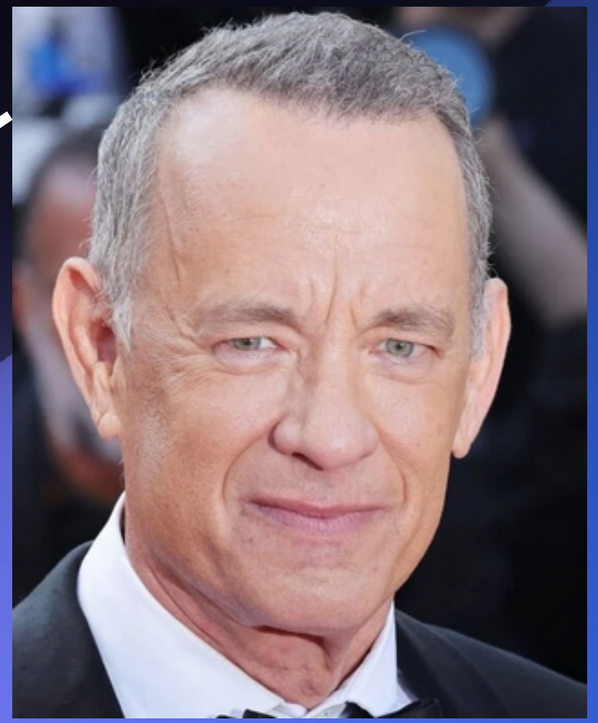
Cual es el objectivo ?

Encontrar el camino más corto entre dos actores a través una secuencia de películas que los conecte.

Jennifer Lawrence



Tom Hanks



Problem

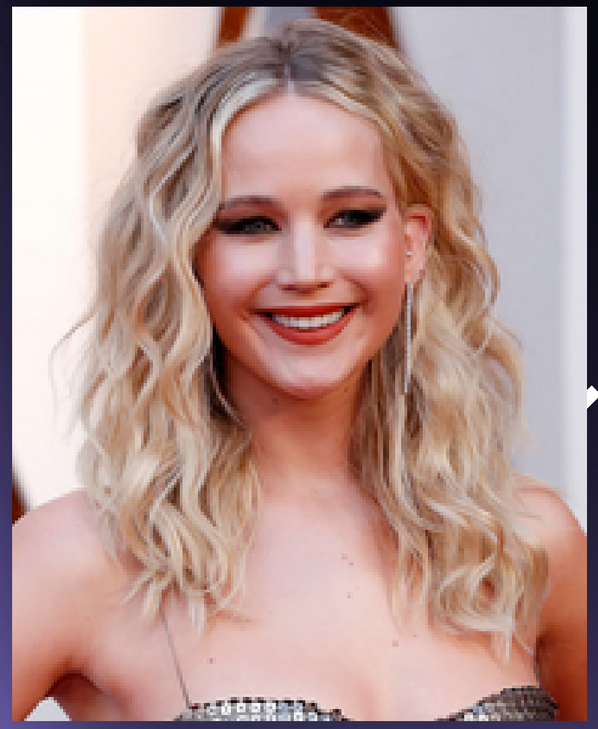
Definir el principio del problema

01

Cual es el objectivo ?

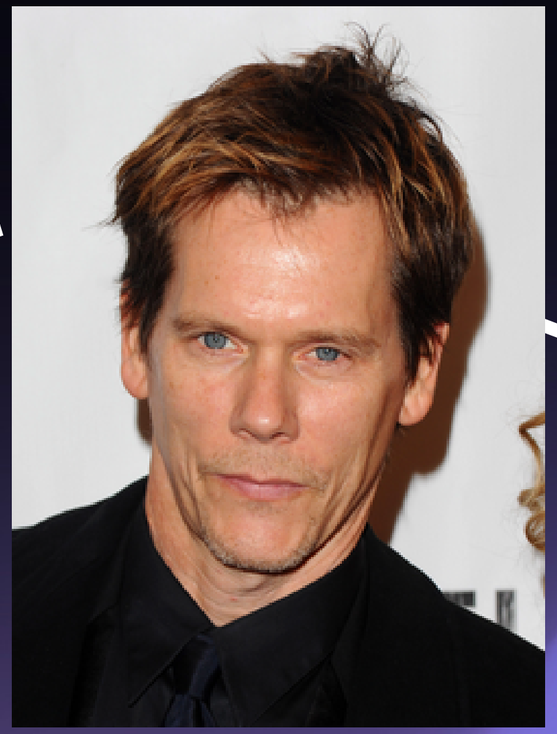
Encontrar el camino más corto entre dos actores a través una secuencia de películas que los conecte.

Jennifer Lawrence



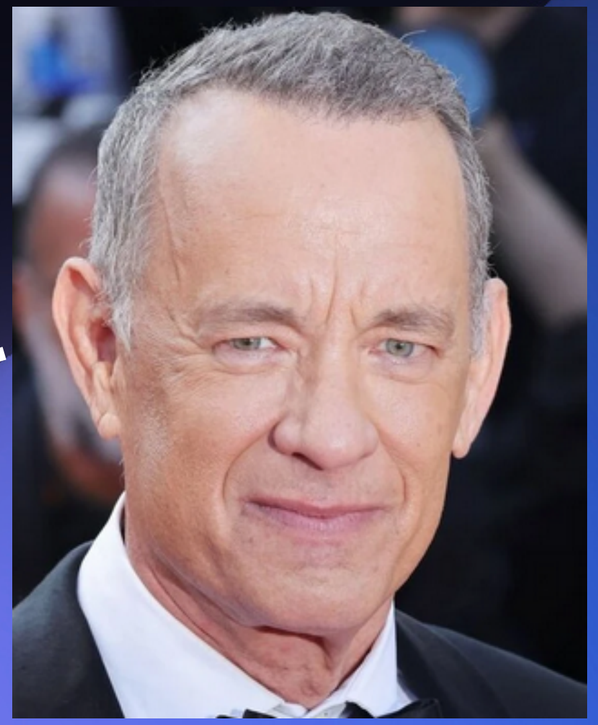
X-Men : First Class

Kevin Bacon



Apollo 13

Tom Hanks



BFS
(Breadth First Search)

DFS
(Depth First Search)

...

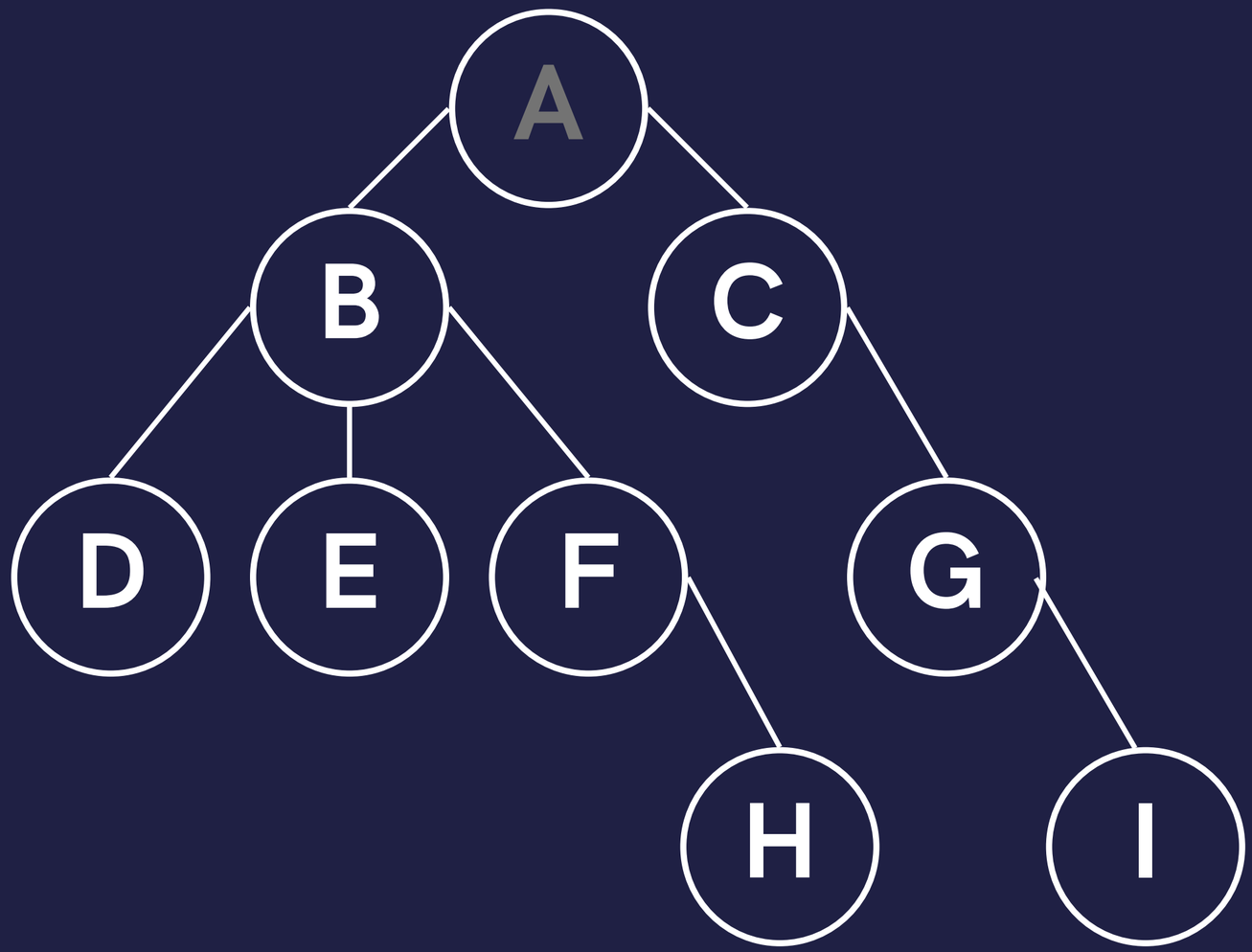
Principo de BFS	Pros/Contras	Principio de DFS
Algoritmo que expande al nodo menos profundo, o sea, a los nodos vecinos antes de pasar a los del siguiente nivel.	DFS : Mas eficiente pero no encuentra el camino más corto BFS : Menos eficiente pero encontra el camino más corto	Algoritmo que expande al nodo más profundo, o sea, explora tanto como sea posible a lo largo de rama.

¿Como funciona el BFS?

Queue : [A]

Negro: Visitado

Gris : A visitar

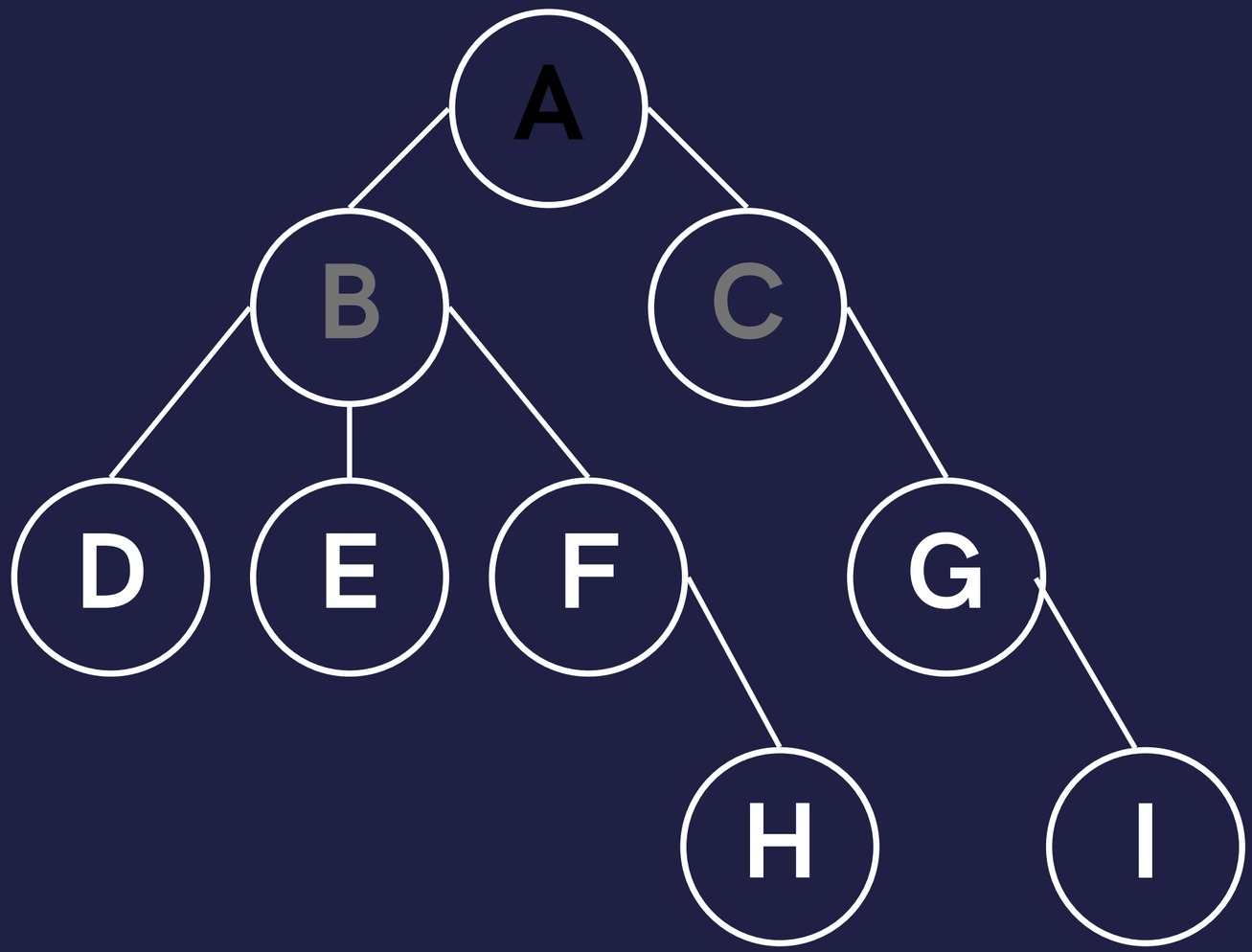


¿Como funciona el BFS?

Queue : [C,B]

Negro: Visitado

Gris : A visitar

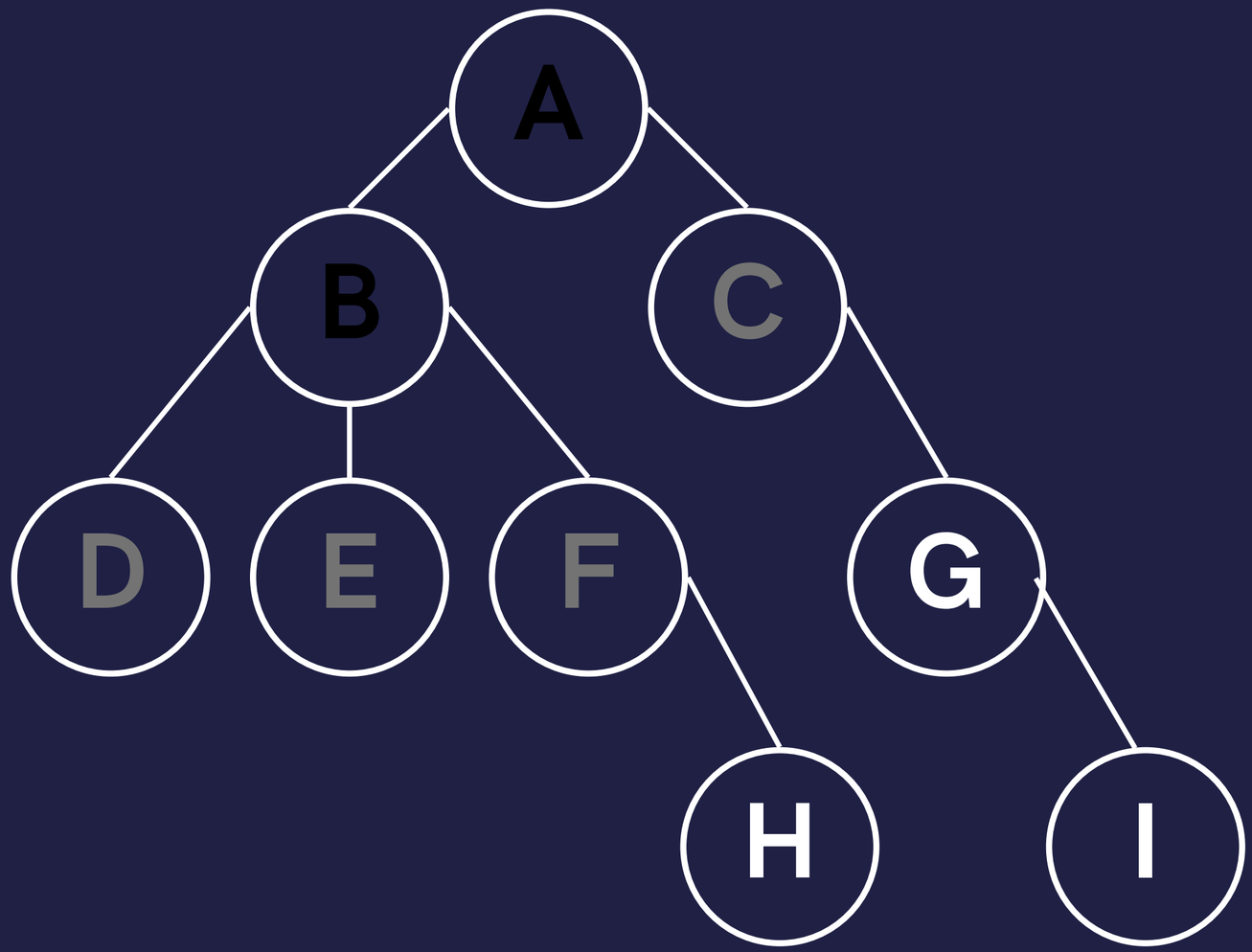


¿Como funciona el BFS?

Queue : [F,E,D,C]

Negro: Visitado

Gris : A visitar

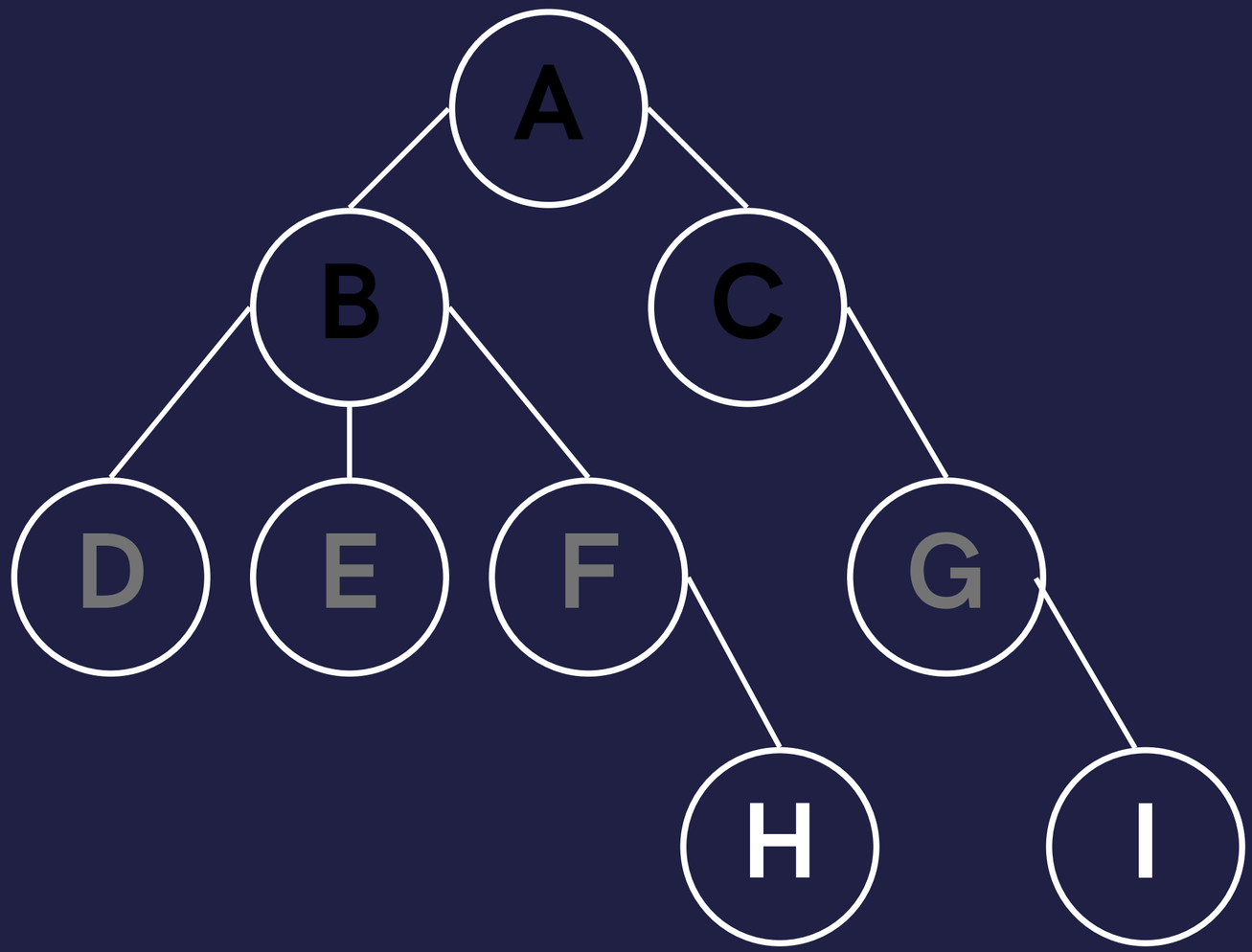


¿Como funciona el BFS?

Queue : [G,F,E,D]

Negro: Visitado

Gris : A visitar

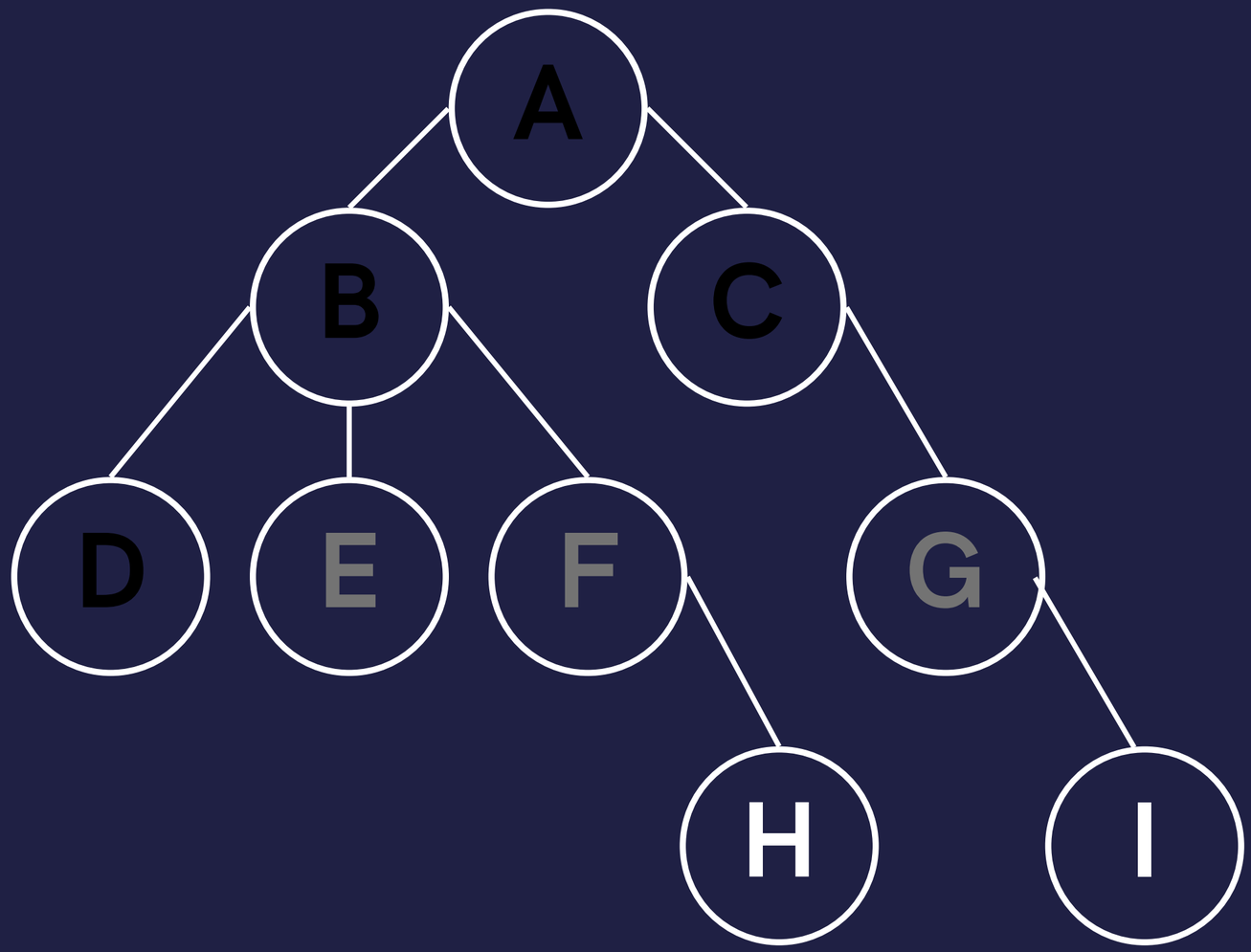


¿Como funciona el BFS?

Queue : [G,F,E]

Negro: Visitado

Gris : A visitar

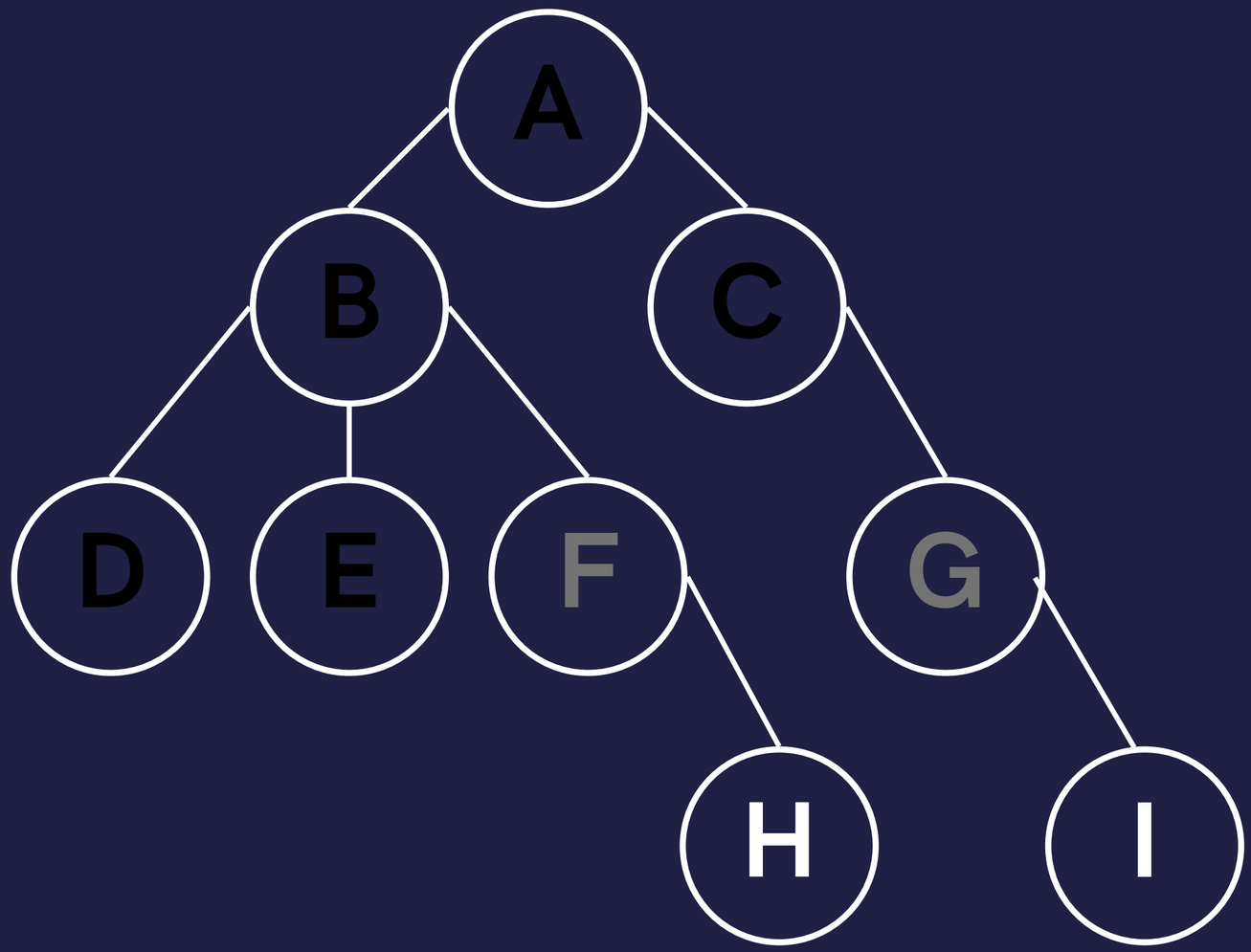


¿Como funciona el BFS?

Queue : [G,F]

Negro: Visitado

Gris : A visitar

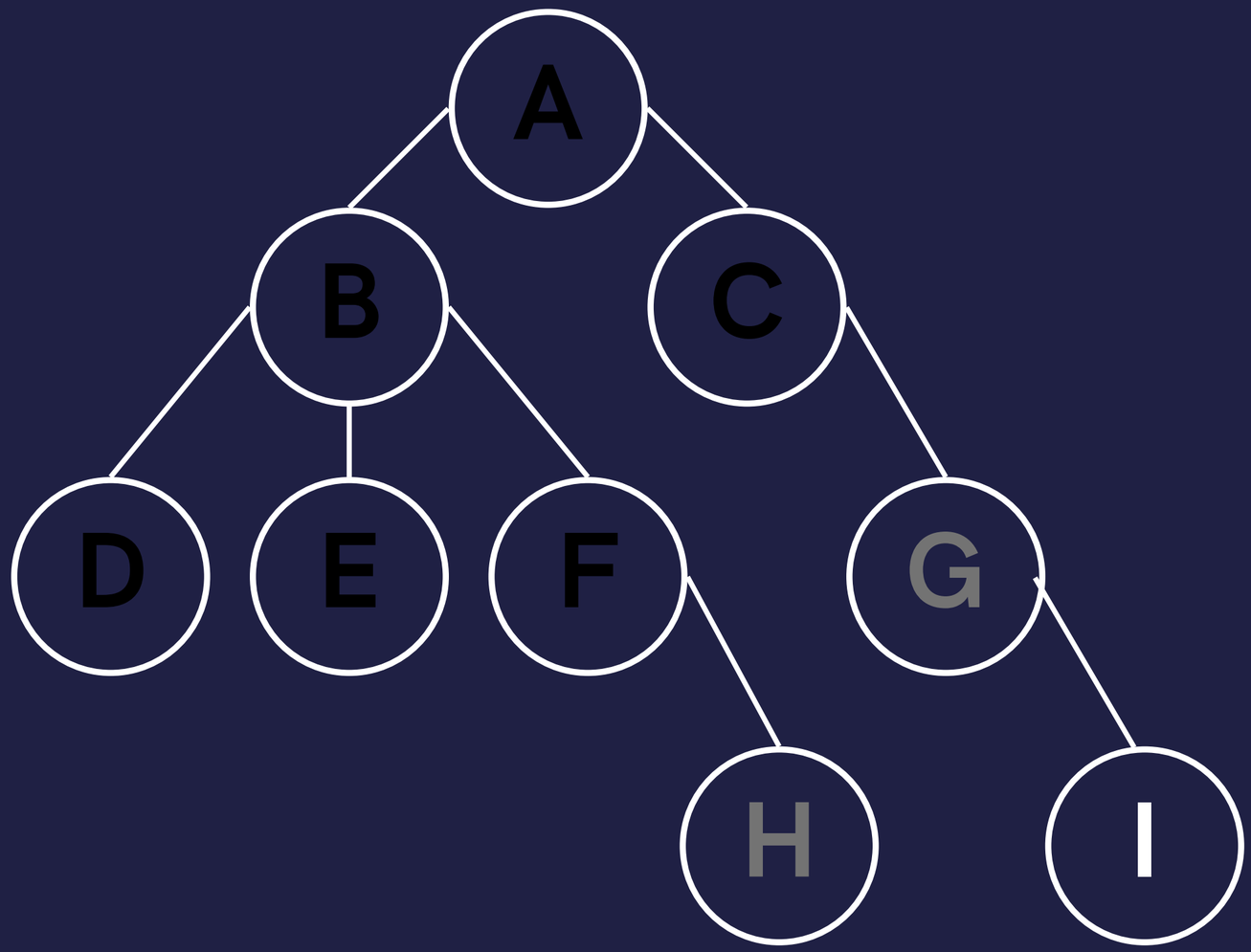


¿Como funciona el BFS?

Queue : [H,G]

Negro: Visitado

Gris : A visitar

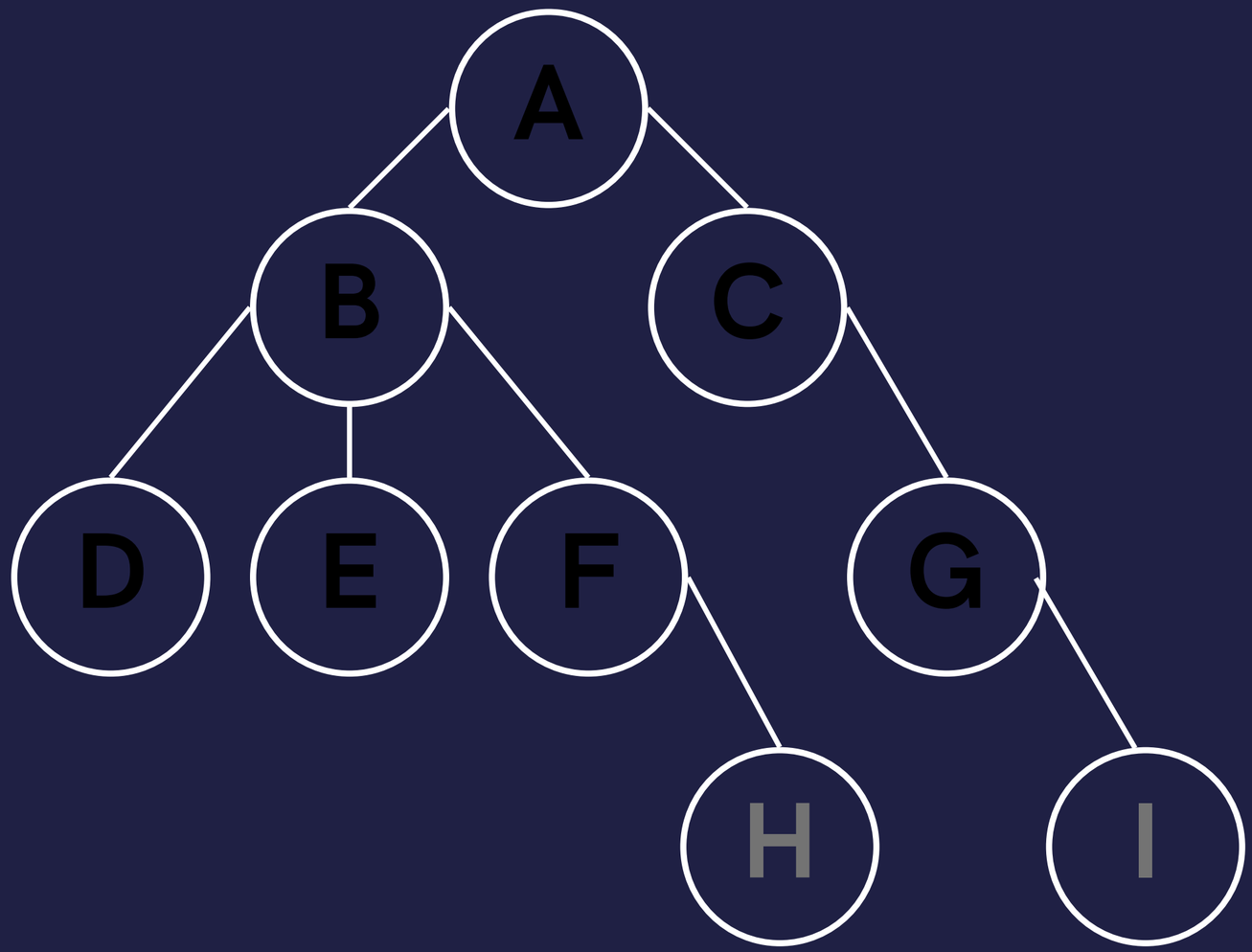


¿Como funciona el BFS?

Queue : [I,H]

Negro: Visitado

Gris : A visitar

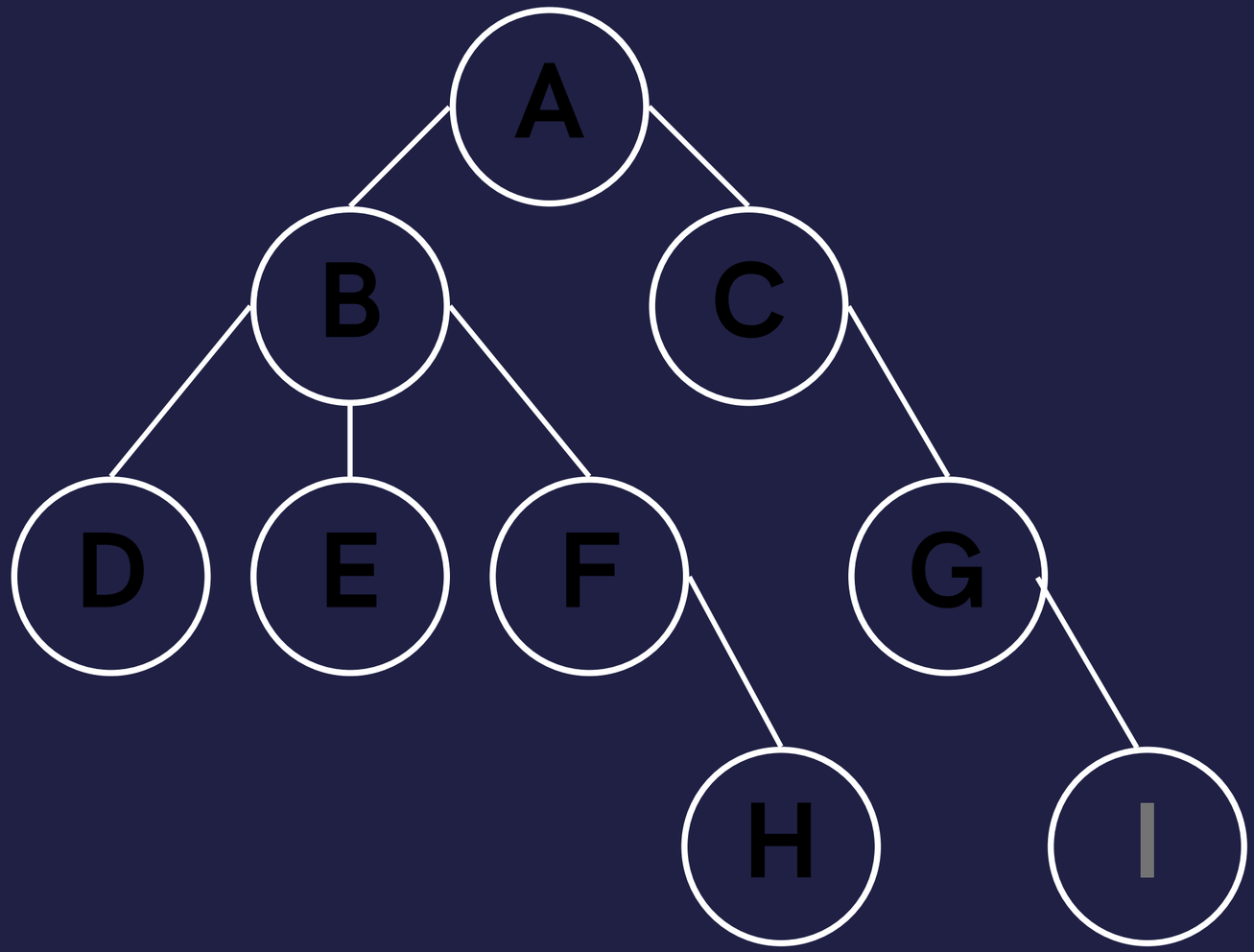


¿Como funciona el BFS?

Queue : [I]

Negro: Visitado

Gris : A visitar

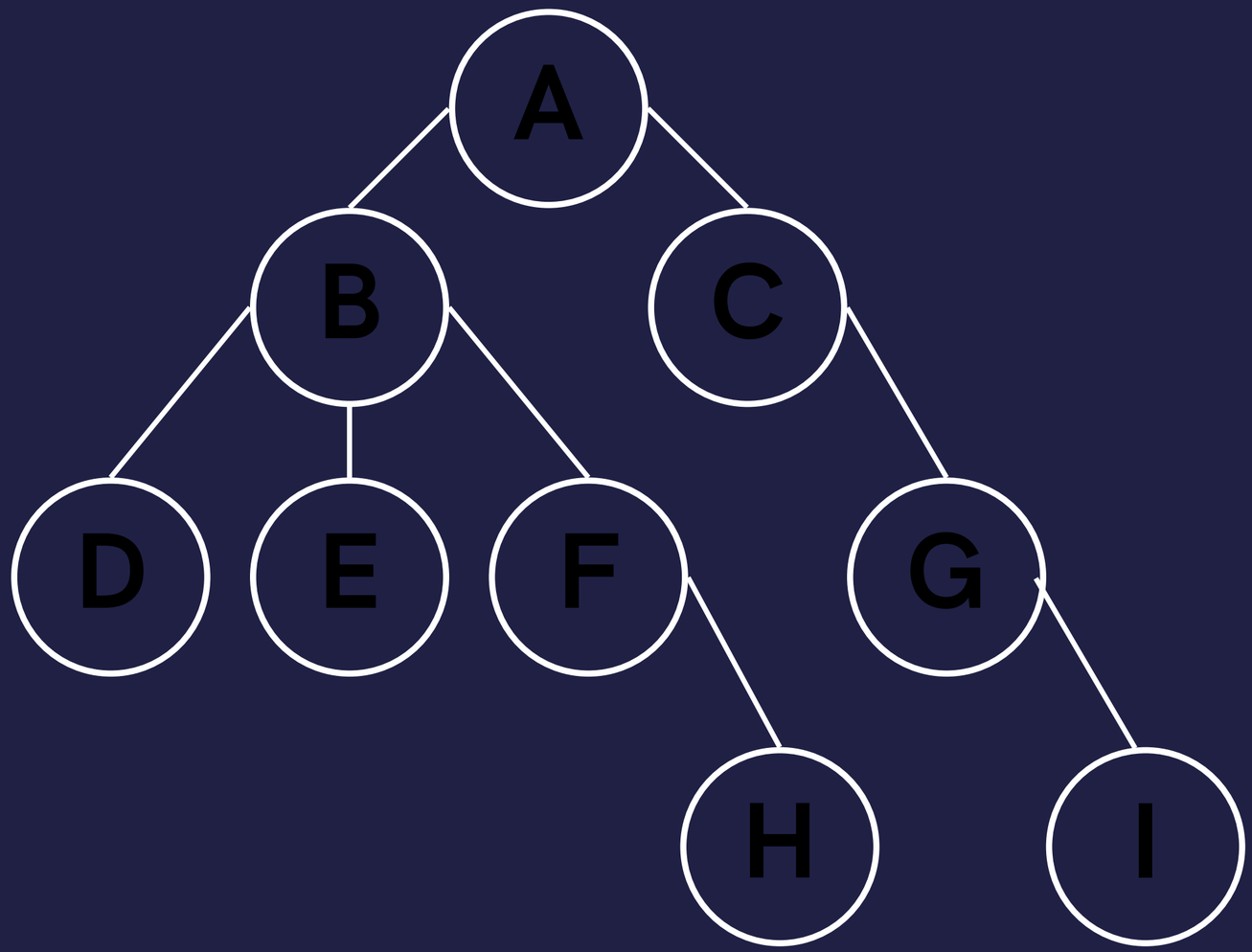


¿Como funciona el BFS?

Queue : []

Negro: Visitado

Gris : A visitar



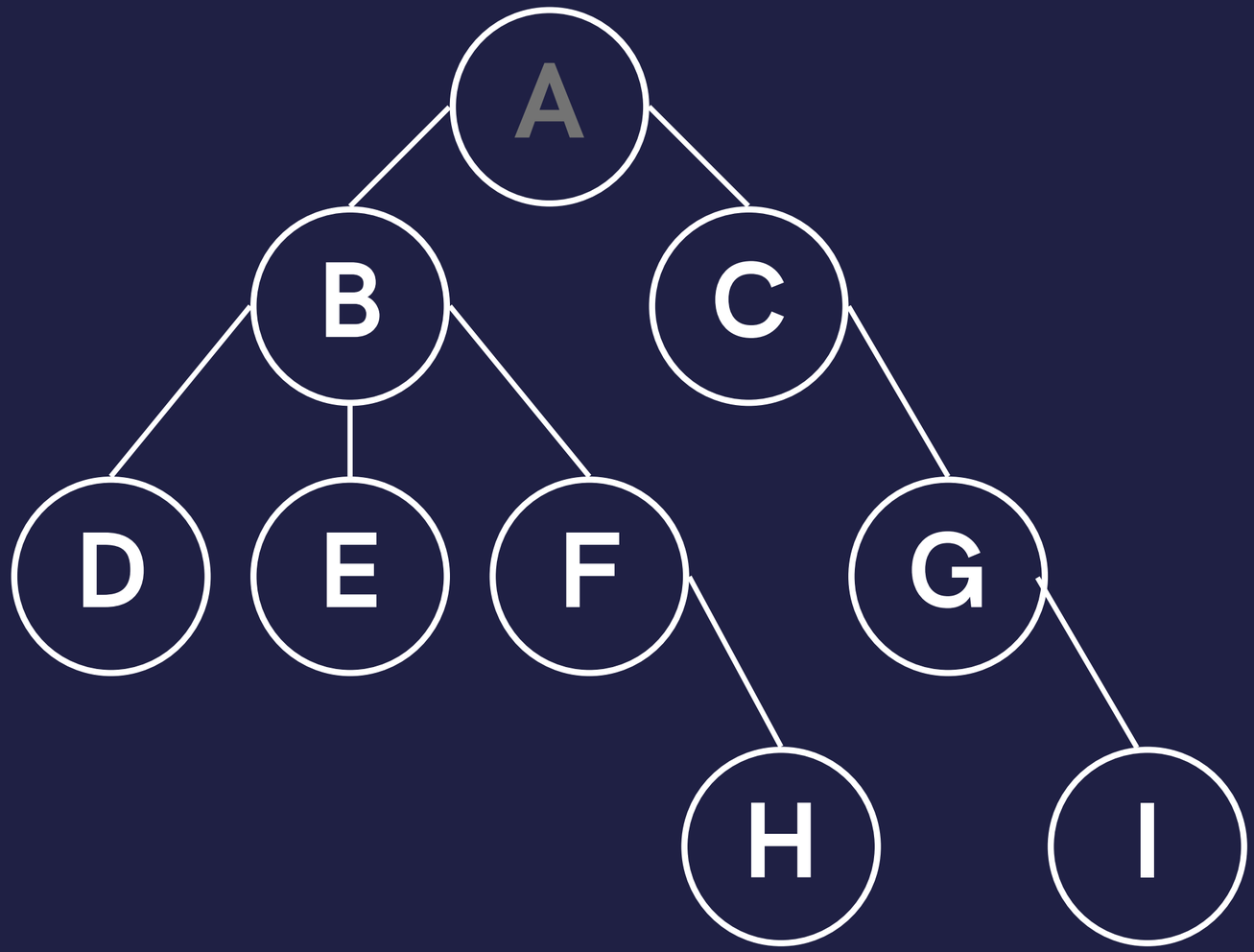
¿Aplicación del BFS?

```
while not frontier.empty():
    # Obtén el nodo actual de la frontera
    node = frontier.remove()

    # Si el nodo actual es el objetivo (target), reconstruye el camino y devuélvelo
    if node.state == target:
        path = []
        while node.parent is not None:
            path.append((node.action, node.state))
            node = node.parent
        path.reverse()
        return path

    # Marca el nodo actual como visitado
    visited.add(node.state)

    # Genera nodos sucesores y agrégalos a la frontera si no han sido visitados
    for action, state in neighbors_for_person(node.state):
        if state not in visited and not frontier.contains_state(state):
            child = Node(state, node, action)
            frontier.add(child)

# Si no se encuentra un camino, devuelve None
return None
```

Tiempo promedio de: 1m35,134s

¿Como funciona el DFS?

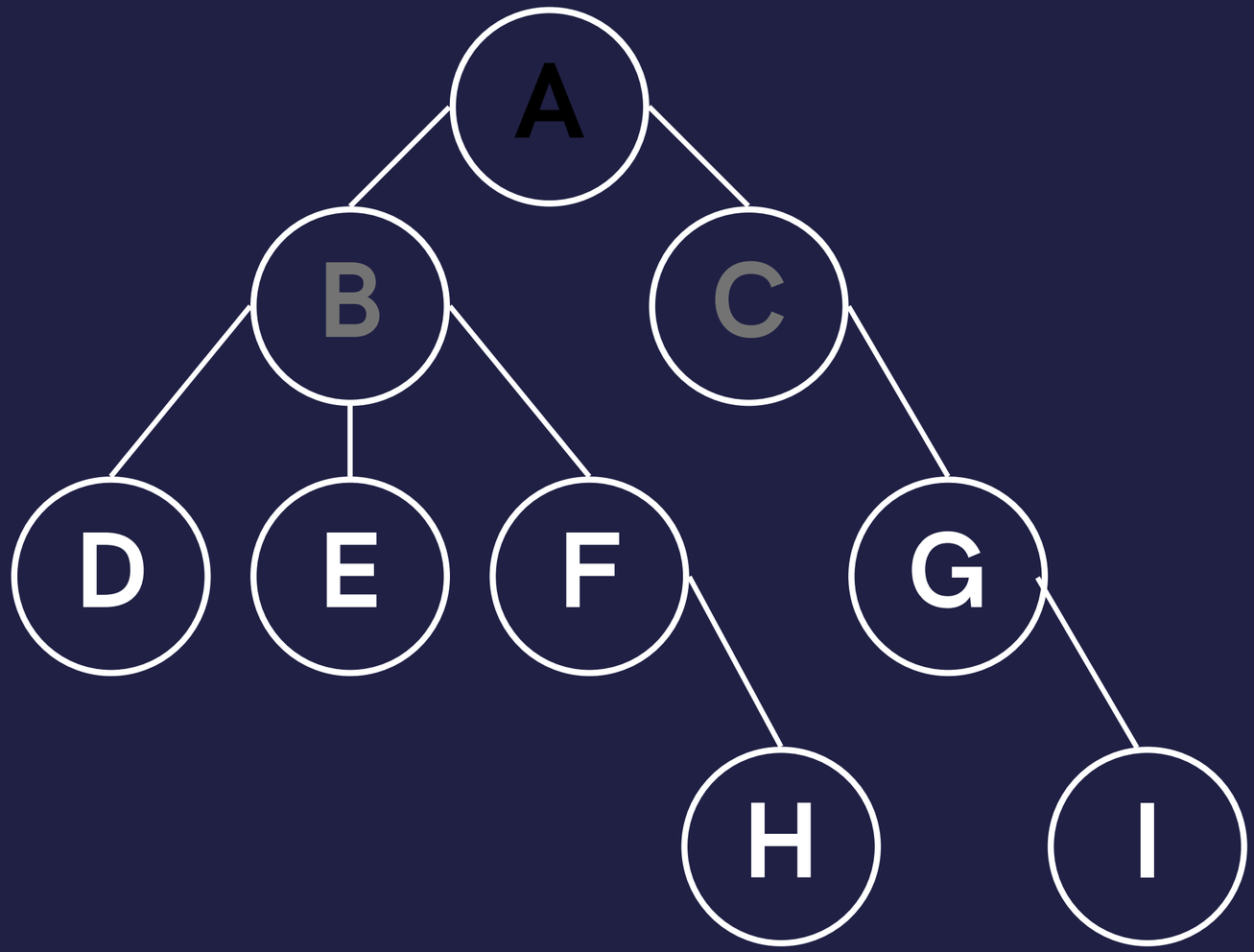
Stack :
[A]



Negro: Visitado
Gris : A visitar

¿Como funciona el DFS?

Stack :
[C]
[B]



Negro: Visitado
Gris : A visitar

¿Como funciona el DFS?

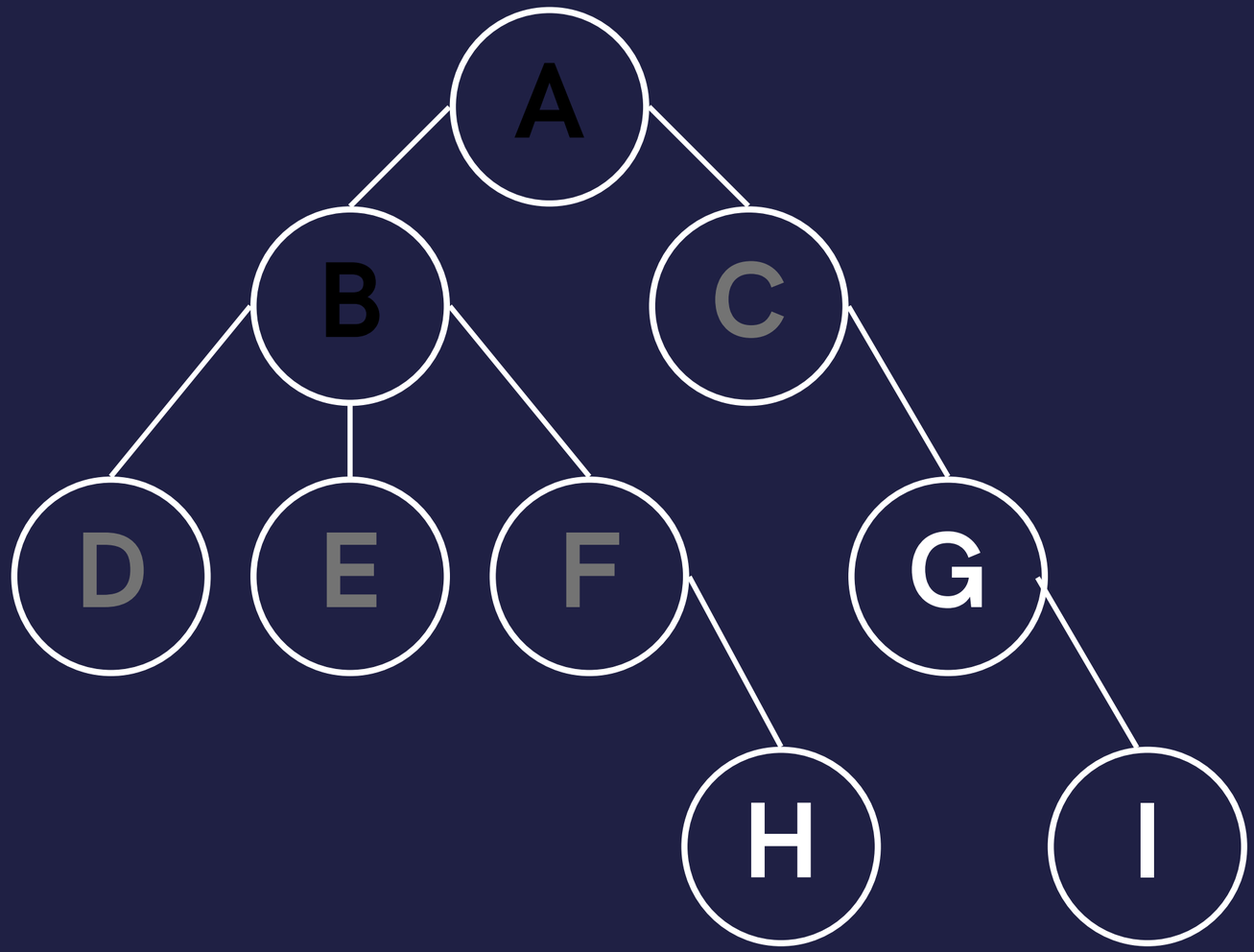
Stack :

[D]

[E]

[F]

[C]



Negro: Visitado

Gris : A visitar

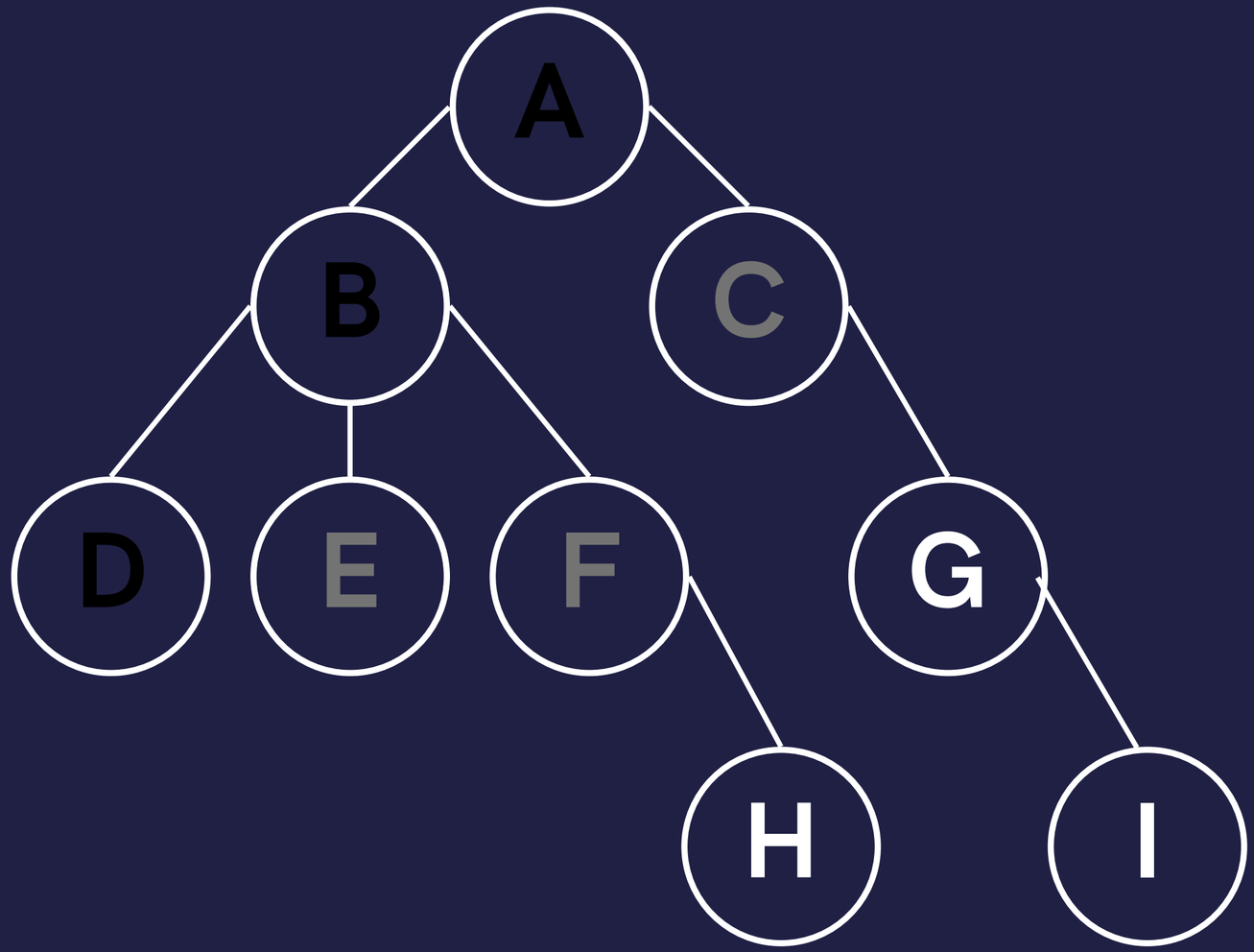
¿Como funciona el DFS?

Stack :

[E]

[F]

[C]



Negro: Visitado

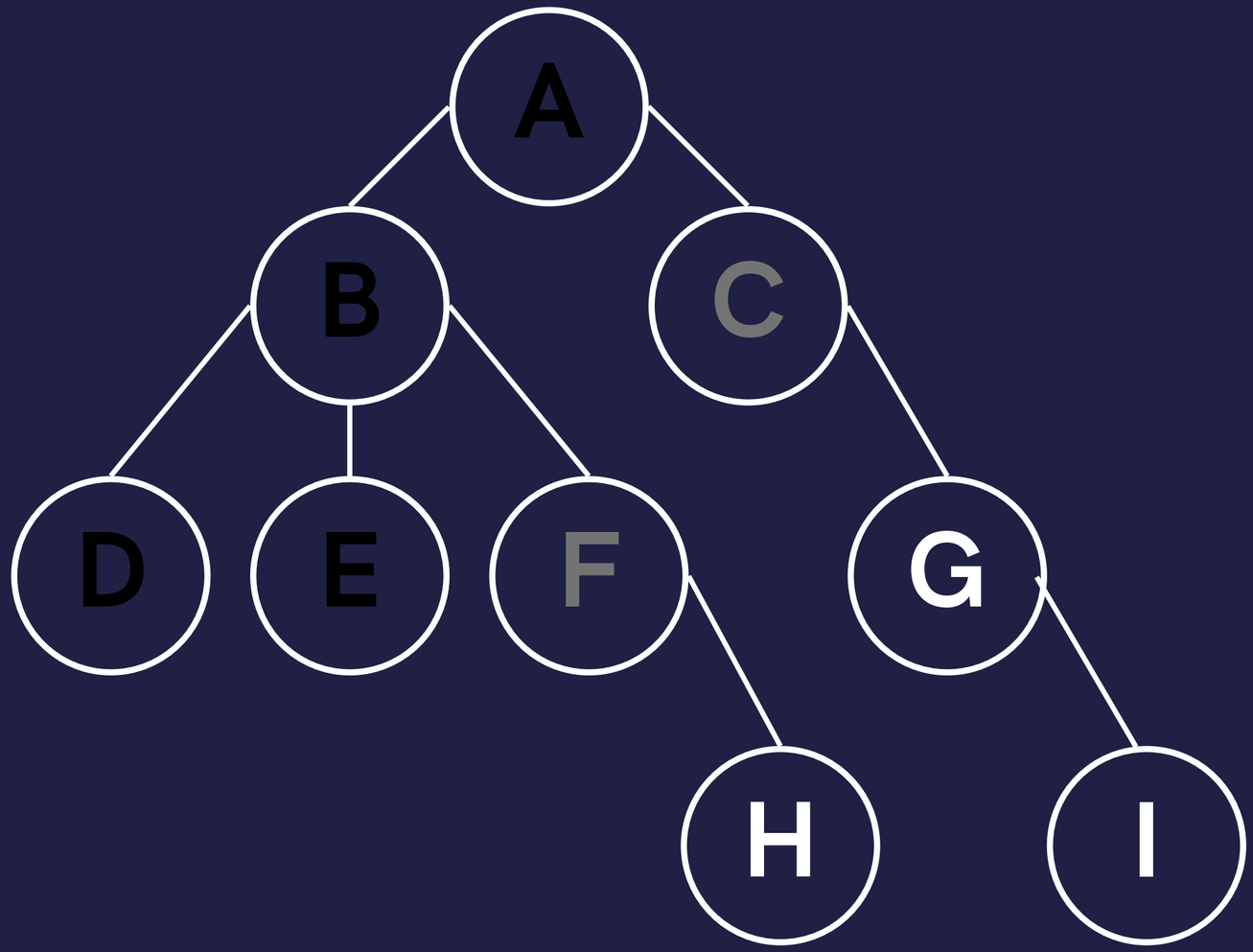
Gris : A visitar

¿Como funciona el DFS?

Stack :

[F]

[C]



Negro: Visitado

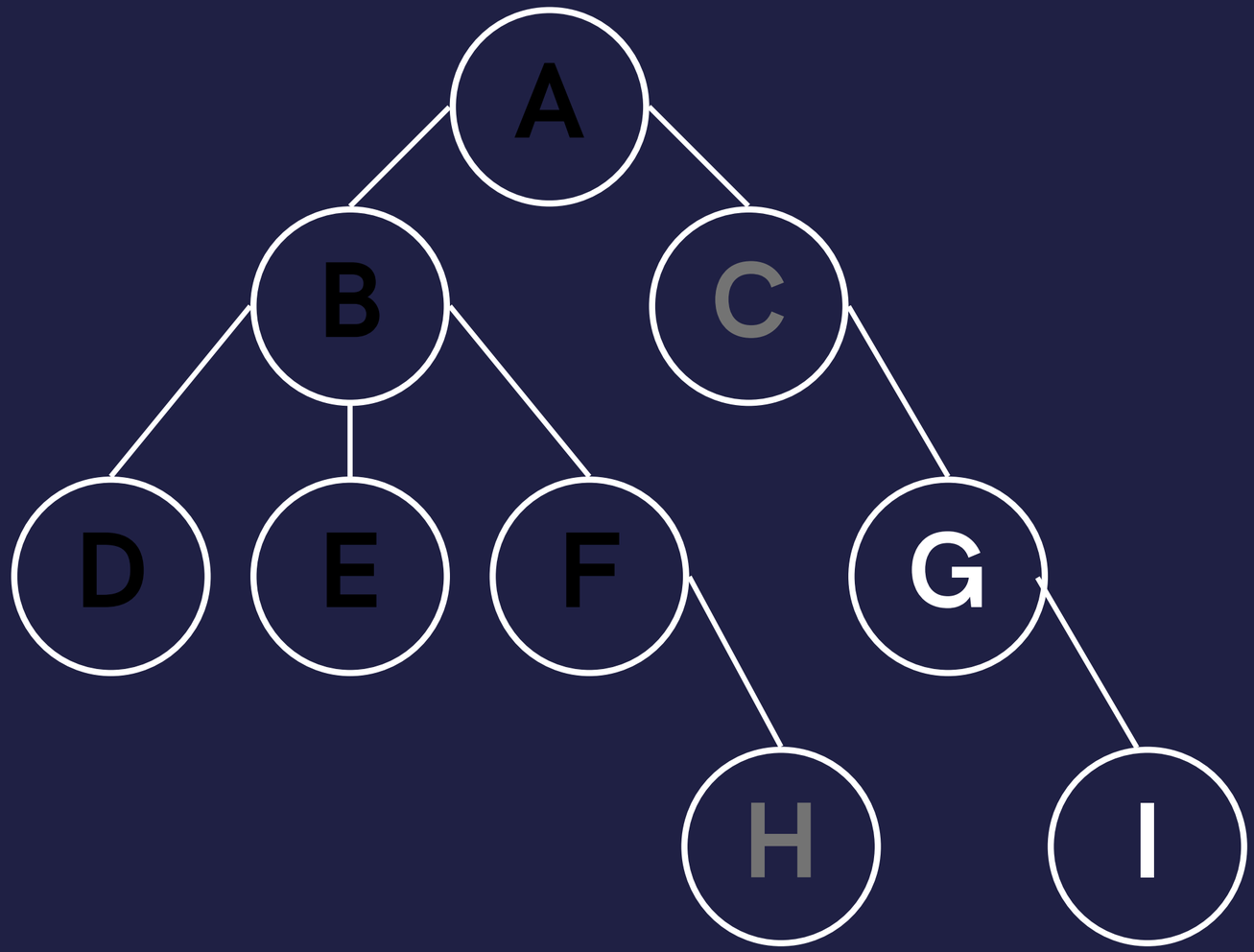
Gris : A visitar

¿Como funciona el DFS?

Stack :

[H]

[C]



Negro: Visitado

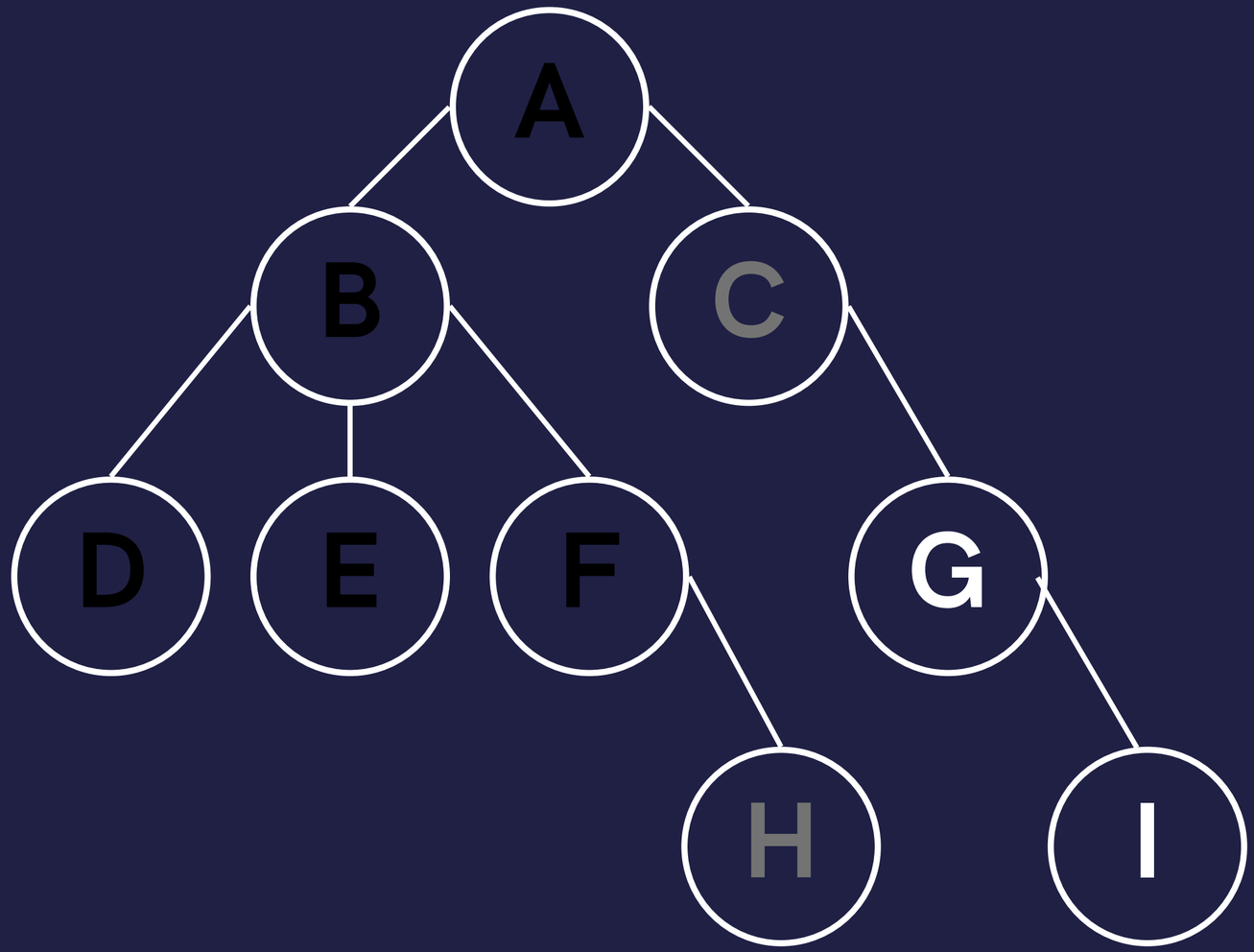
Gris : A visitar

¿Como funciona el DFS?

Stack :

[H]

[C]

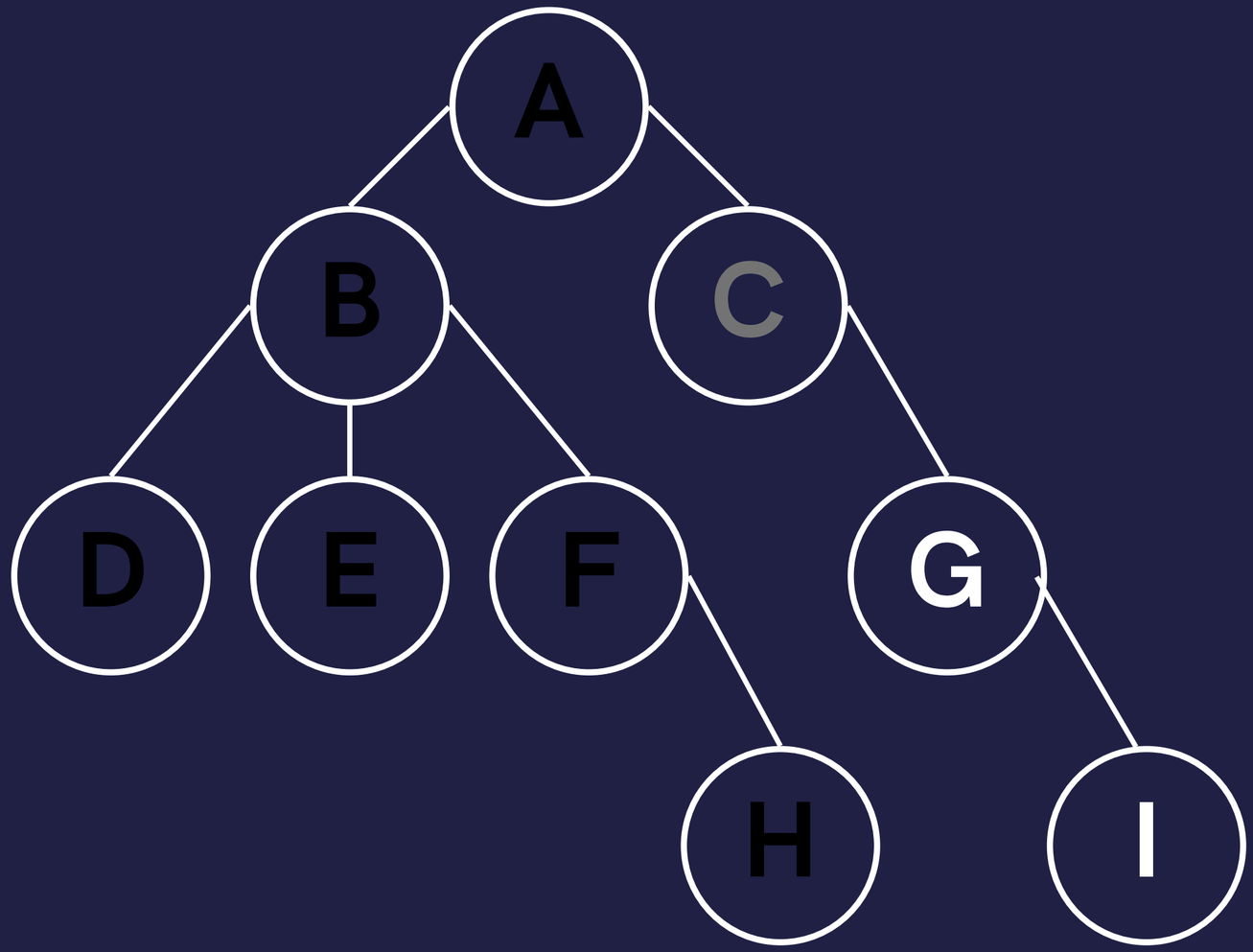


Negro: Visitado

Gris : A visitar

¿Como funciona el DFS?

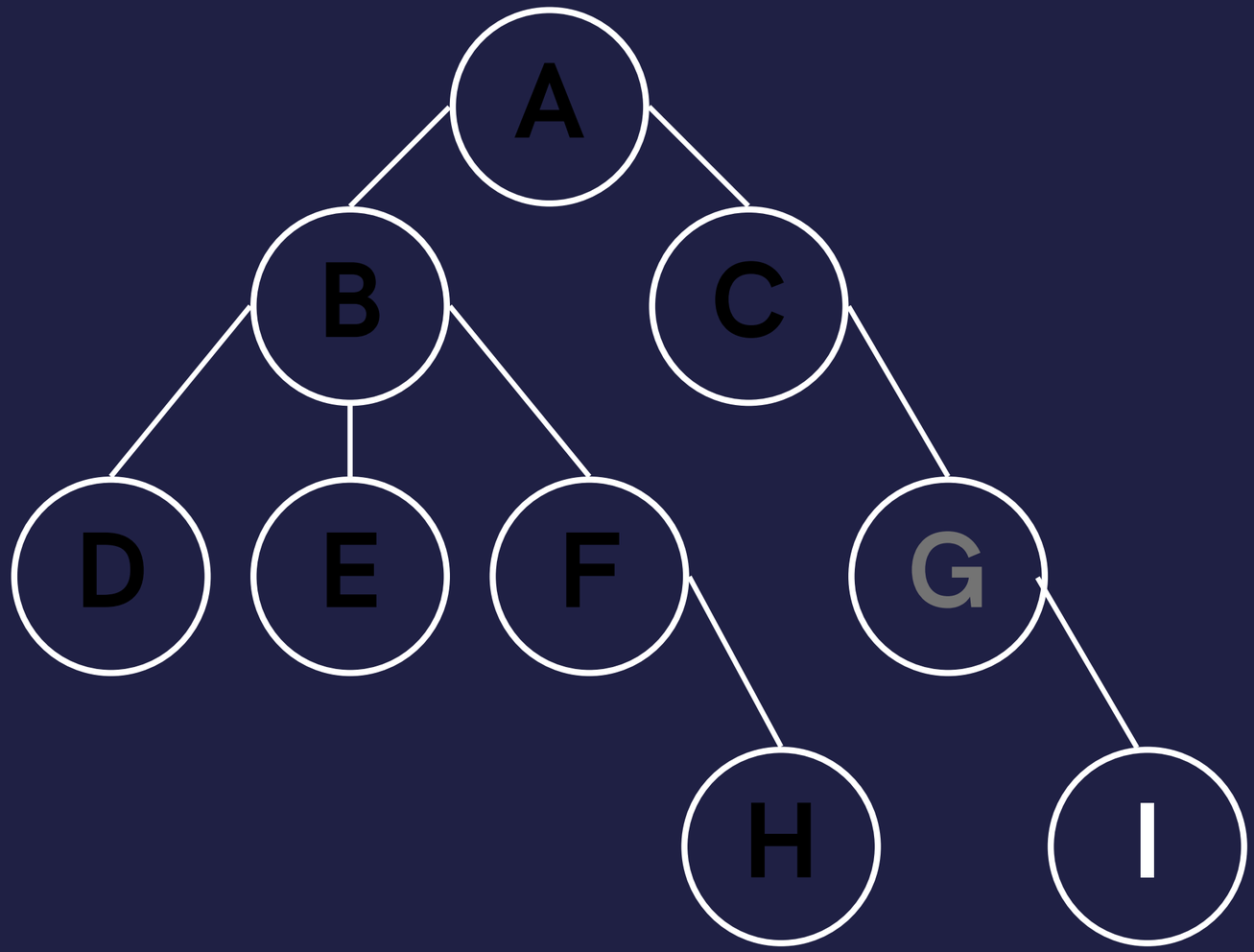
Stack :
[C]



Negro: Visitado
Gris : A visitar

¿Como funciona el DFS?

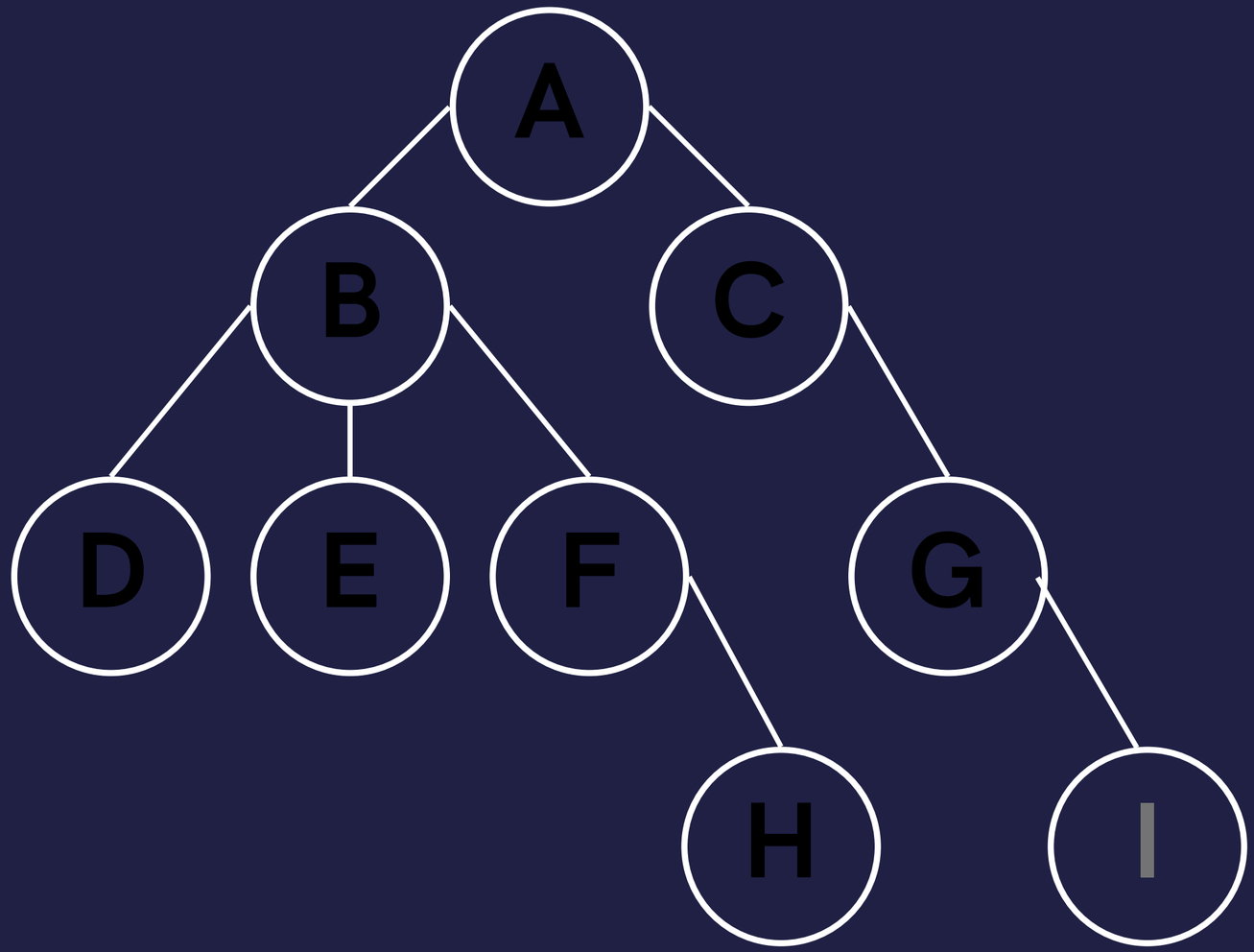
Stack :
[G]



Negro: Visitado
Gris : A visitar

¿Como funciona el DFS?

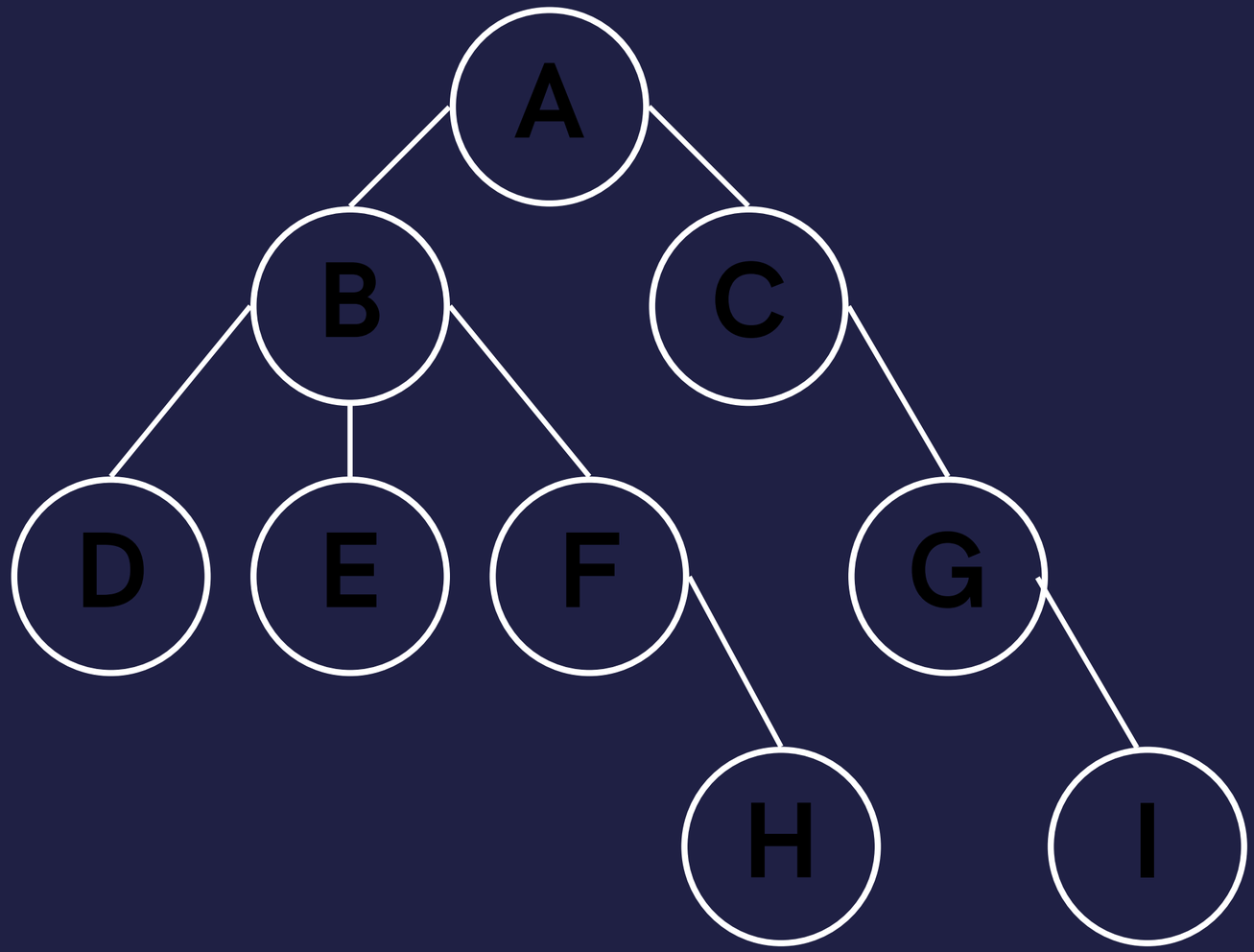
Stack :
[I]



Negro: Visitado
Gris : A visitar

¿Como funciona el DFS?

Stack :



Negro: Visitado

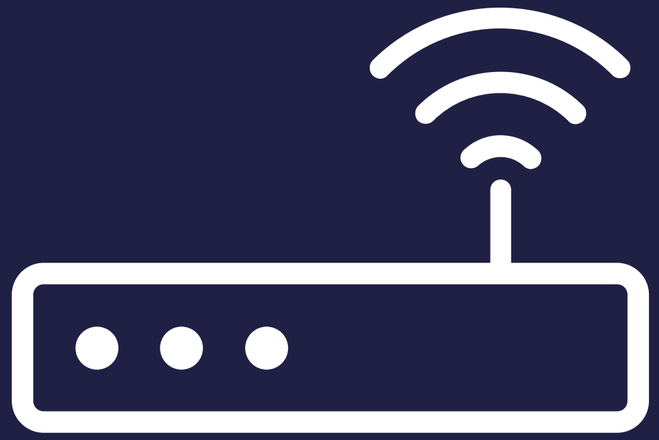
Gris : A visitar

¿Aplicación del DFS?

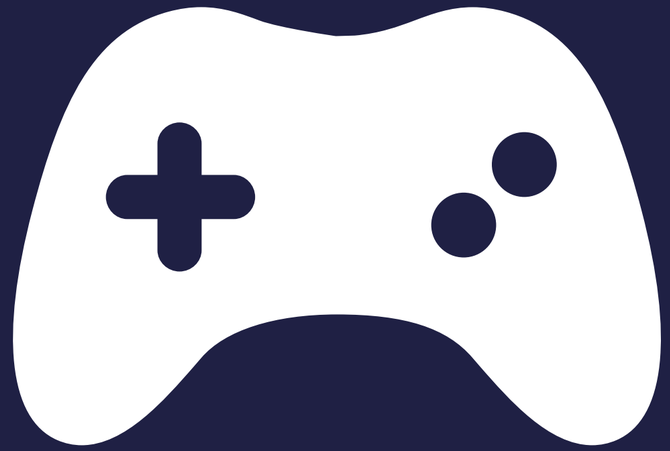
```
while not frontier.empty():
    # Get the current node from the frontier (pop from the stack)
    node = frontier.remove()

    # If the current node is the target, reconstruct and return the path
    if node.state == target:
        path = []
        while node.parent is not None:
            path.append((node.action, node.state))
            node = node.parent
        path.reverse()
        return path

    # Mark the current node as visited
    visited.add(node.state)

    # Generate successor nodes and add them to the frontier if not visited
    for action, state in neighbors_for_person(node.state):
        if state not in visited and not frontier.contains_state(state):
            child = Node(state, node, action)
            frontier.add(child)

# If no path is found, return None
return None
```

Tiempo promedio de: 0m43,928s

Casos de aplicación



Redes



Path finding

GPS

Mejoras

Limite de profundidad

Paralilizar

Bidireccionalidad

Gracias :)

Referencias:

- https://www.geeksforgeeks.org/difference-between-bfs-and-dfs/
- https://www.youtube.com/watch?v=Urx87-NMm6c - **Depth-first search in 4 minutes**
- https://www.youtube.com/watch?v=HZ5YTanv5QE& - **Breadth-first search in 4 minutes**
- **h**ttps://towardsdatascience.com/search-algorithms-concepts-and-implementation-1073594aeda6
- Chat GPT obviamente