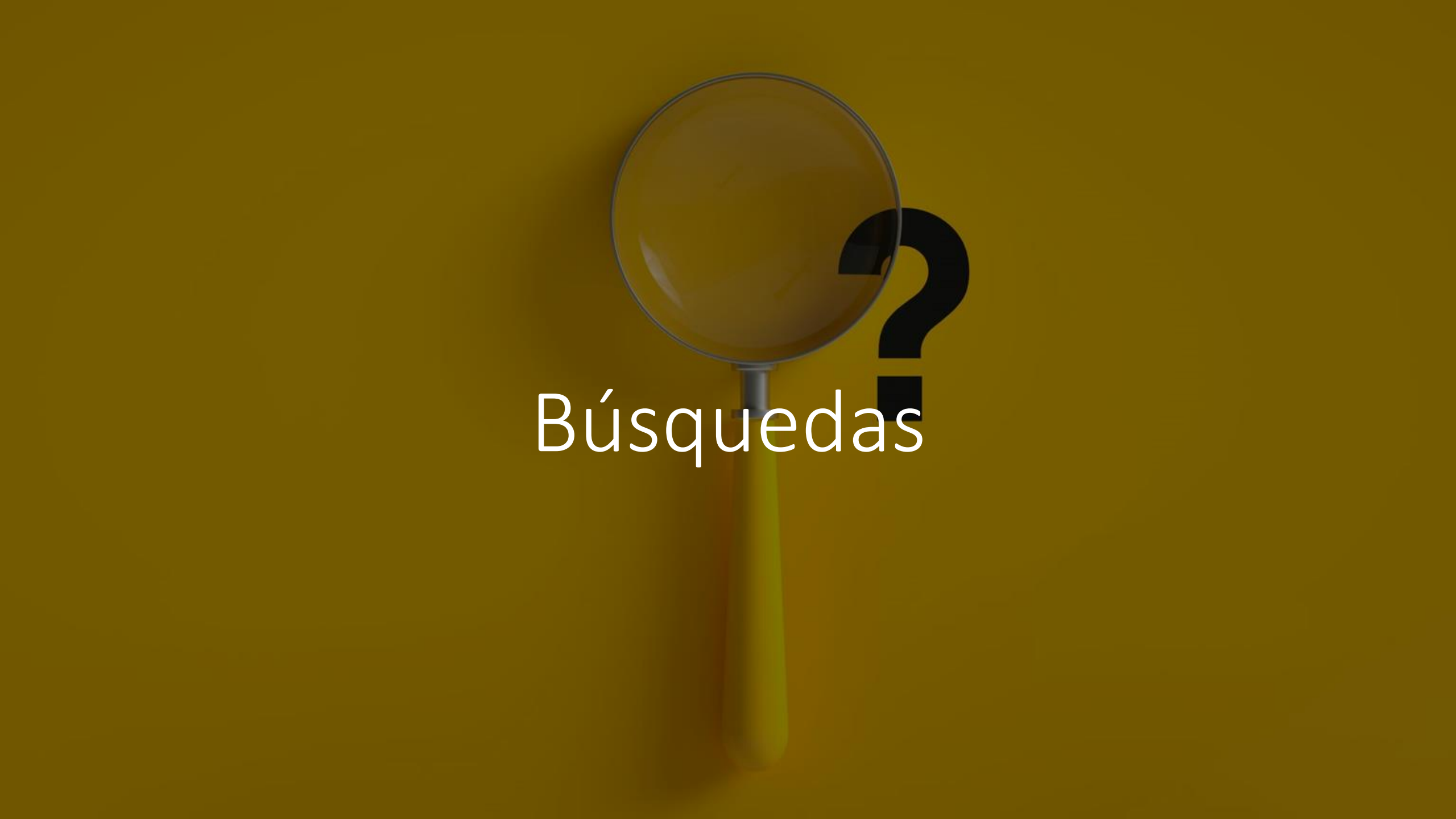


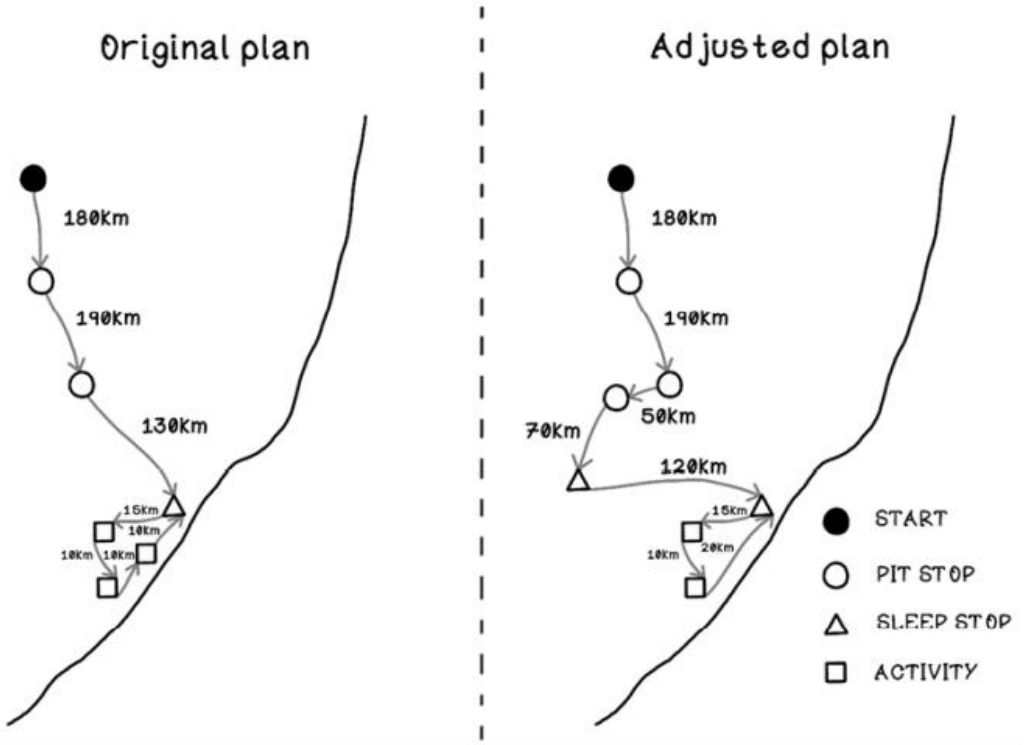
Búsquedas

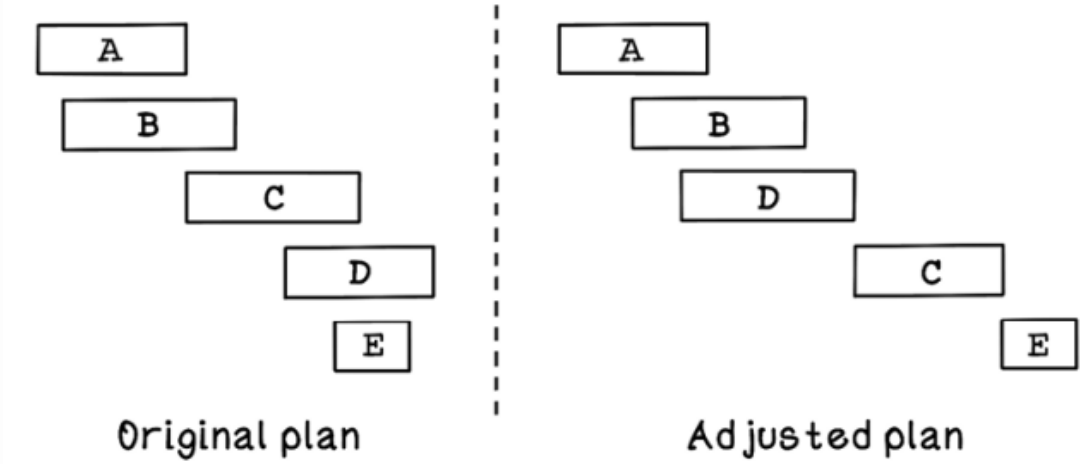
Necesidades

- La intuición de planificar y buscar
- Identificar problemas aptos para ser resueltos utilizando algoritmos de búsqueda.
- Representar espacios problemáticos de una manera adecuada para ser procesados por algoritmos de búsqueda.
- Comprender y diseñar algoritmos de búsquedas son fundamentales para resolver problemas

Planes

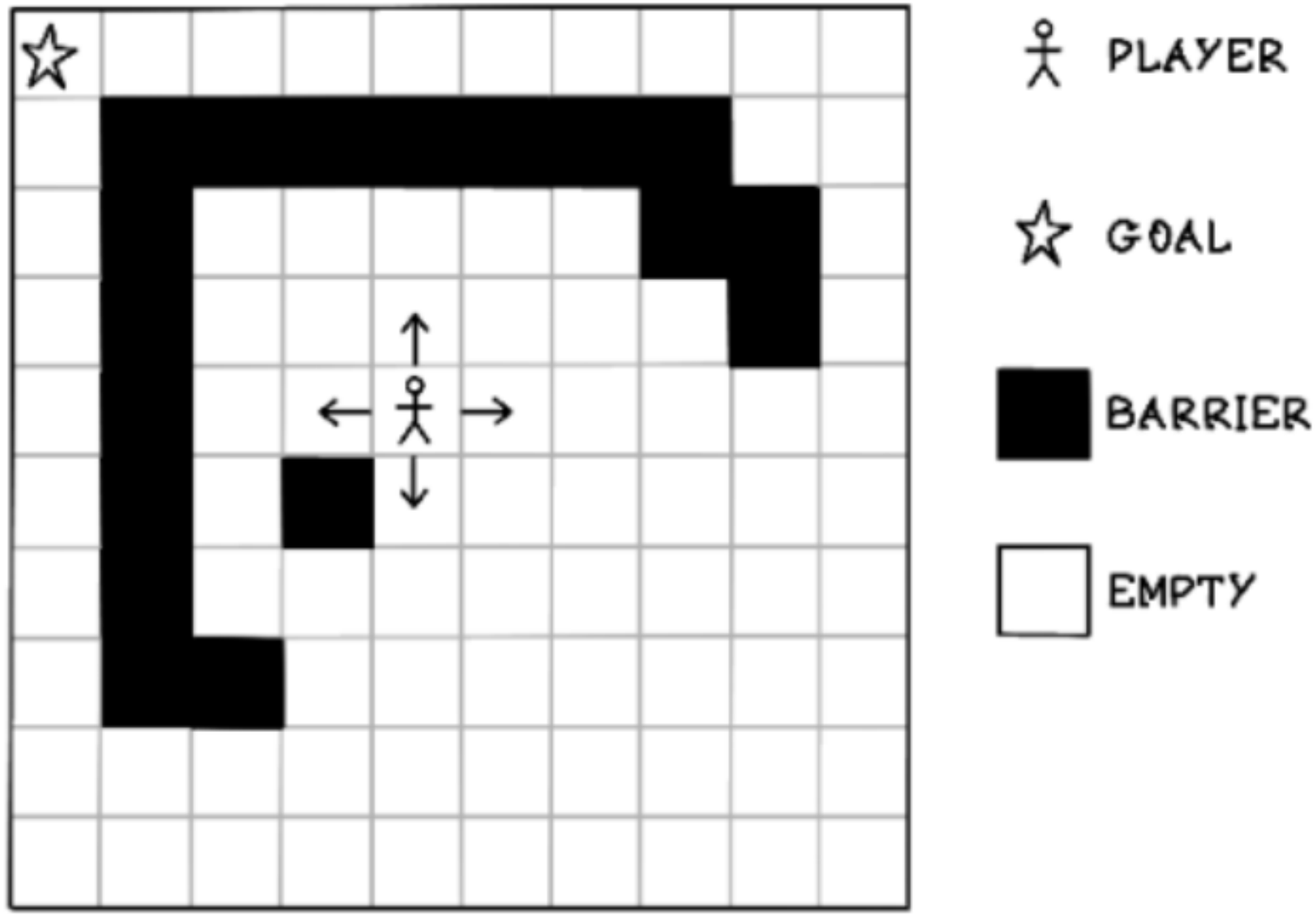
- La búsqueda implica evaluar estados futuros hacia una meta con el objetivo de encontrar un camino óptimo de estados hasta alcanzar la meta.

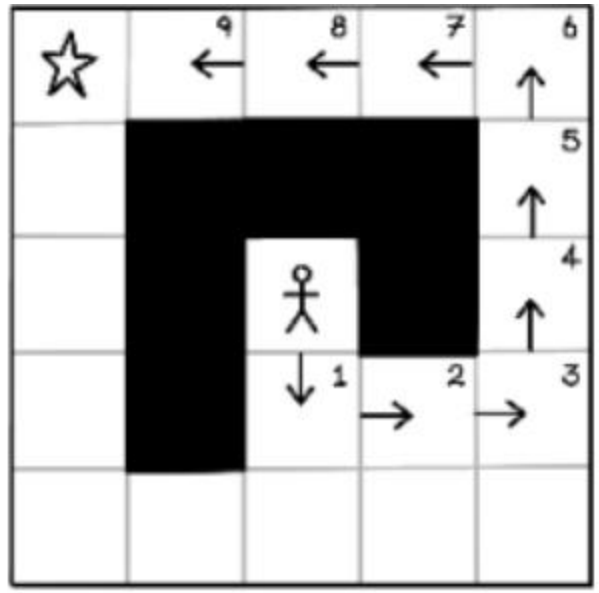


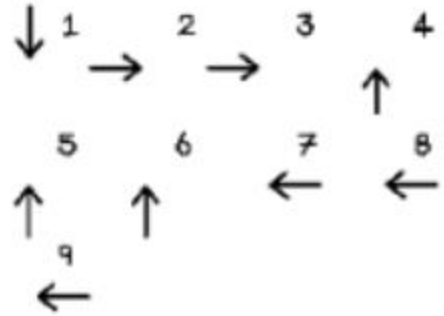


Búsqueda

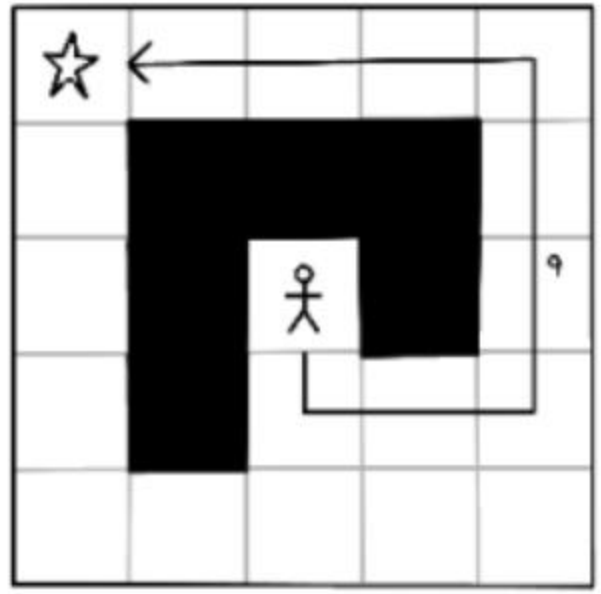
- *La búsqueda* es una forma de guiar la planificación mediante la creación de pasos en un plan.
- Cuando planificamos un viaje, por ejemplo, buscamos rutas a tomar, evaluamos las paradas en el camino y lo que ofrecen, y buscamos alojamientos y actividades que se alineen con nuestro gusto y presupuesto. Dependiendo de los resultados de estas búsquedas, el plan cambia.

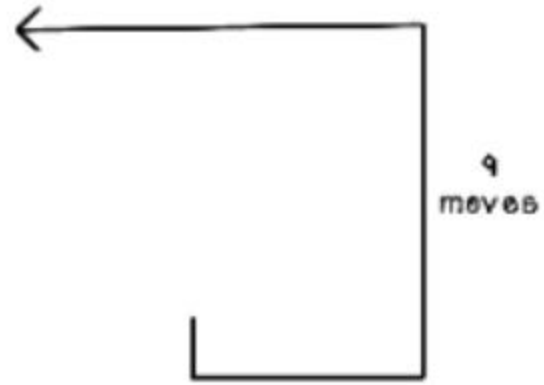

PLAYER
GOAL
BARRIER
EMPTY





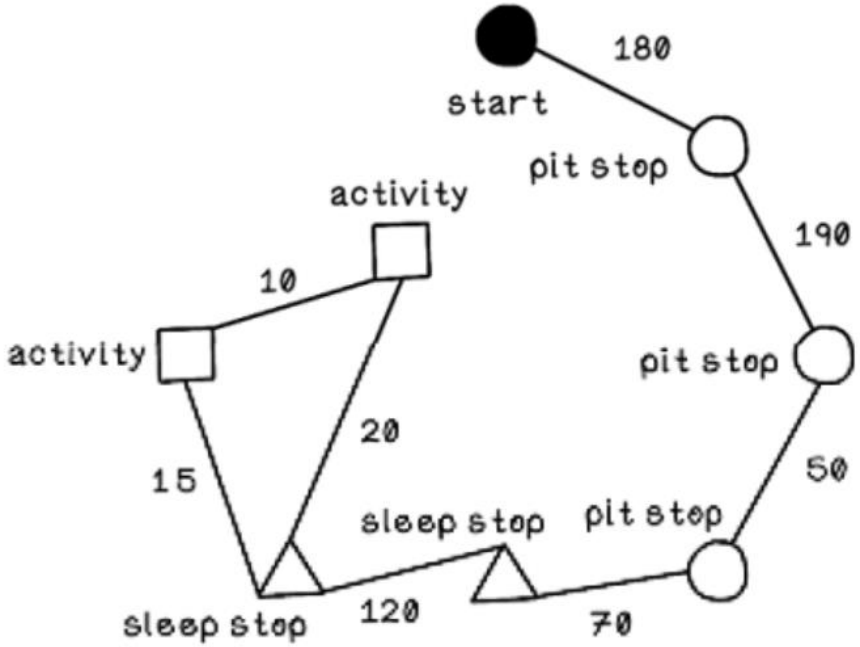
DATA

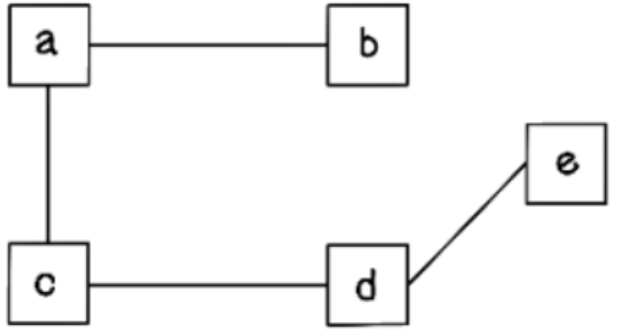




INFORMATION

Gráficos: representación de problemas de búsqueda y soluciones






```
[ [a , b] ,
  [a , c] ,
  [c , d] ,
  [d , e] ]
```

Un *nodo adyacente* es un nodo que está conectado directamente a otro nodo.

Árboles: Las estructuras concretas utilizadas para representar soluciones de búsqueda.

- Un *tree* es una estructura de datos popular que simula una jerarquía de valores u objetos. Una *jerarquía* es una disposición de cosas en la que un solo objeto está relacionado con varios otros objetos debajo de él. Un árbol es un *gráfico acíclico conectado* : cada nodo tiene una arista con otro nodo y no existen ciclos.
- En un árbol, el valor u objeto representado en un punto específico se denomina *nodo* . Los árboles suelen tener una solo nodo raíz con cero o más nodos secundarios que podrían contener subárboles.

- Cada nodo hijo tiene un solo nodo padre. Un nodo sin hijos es un nodo hoja.
- Los árboles también tienen una altura total. El nivel de nodos específicos se llama *profundidad* .

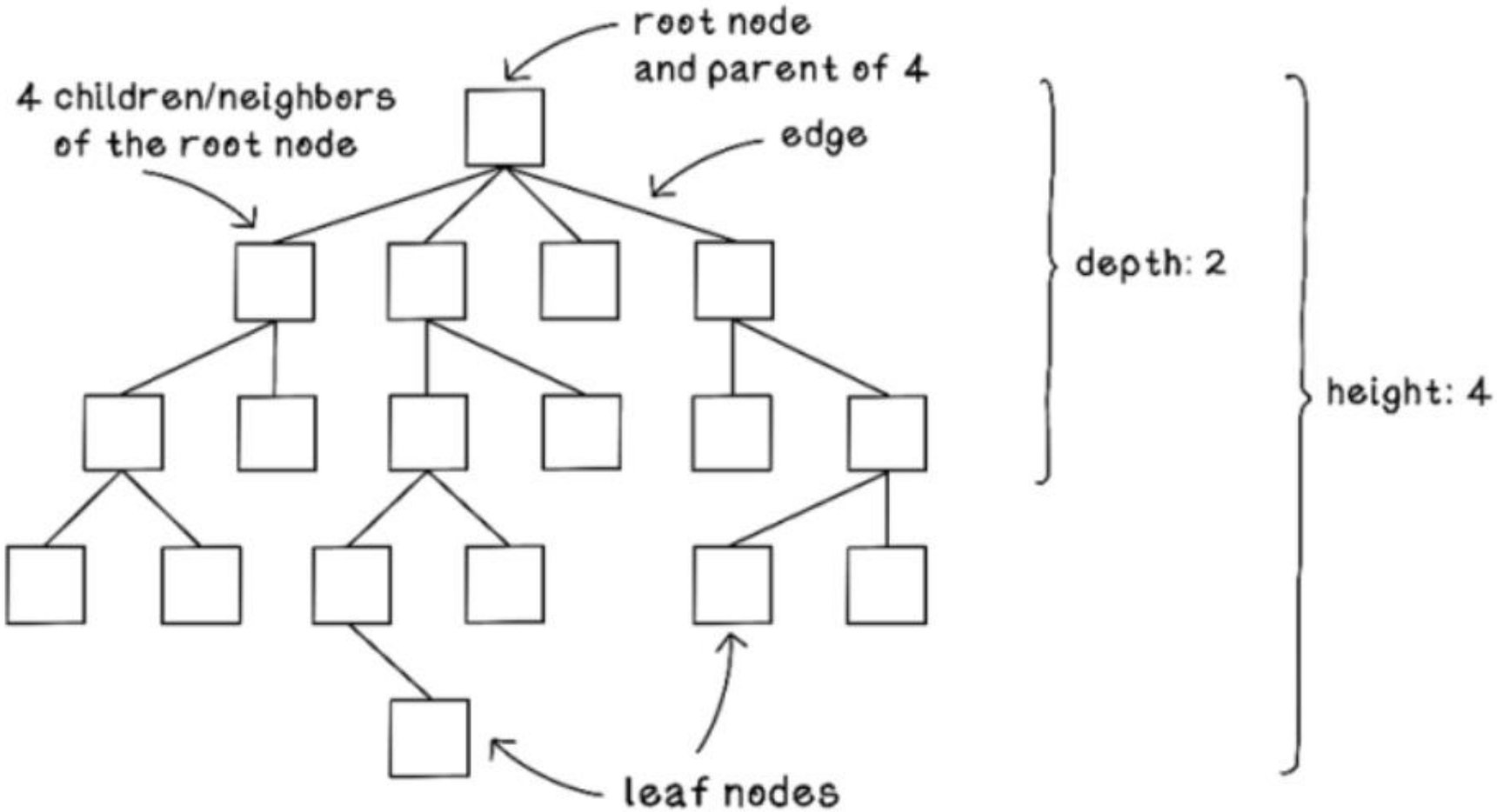
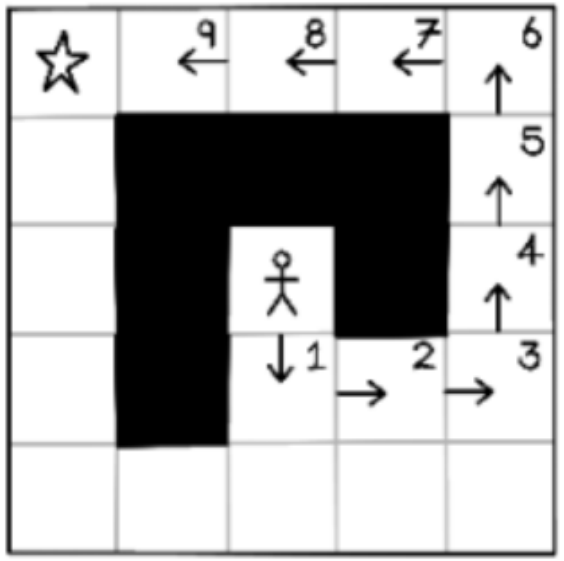
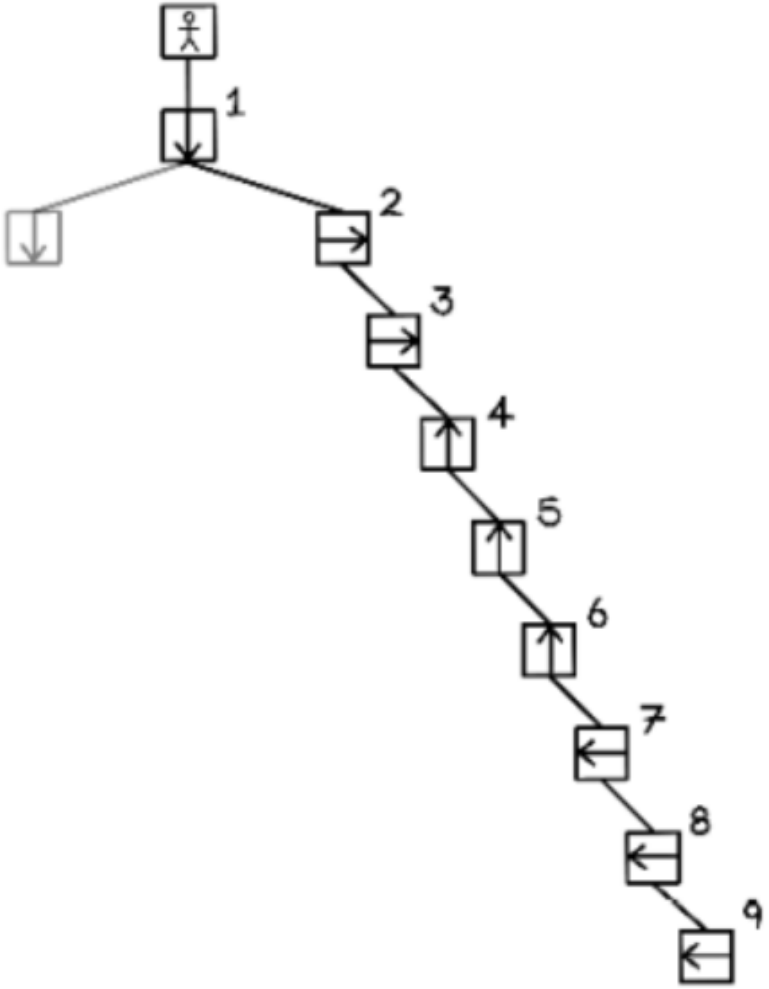


Figura 2.12 Los principales atributos de un árbol

- Depth: profundidad/niveles
- Height: Altura
- Edge: Borde
- Leaf nodes: Nodos hoja

Los árboles son la estructura de datos fundamental para los algoritmos de búsqueda





Algortimos de Búsqueda Sin Información

Búsqueda desinformada: buscando soluciones a ciegas

- *No informado la búsqueda* también conocida como *búsqueda no guiada* , búsqueda *ciega* o *búsqueda de fuerza bruta* .

Búsquedas óptimas

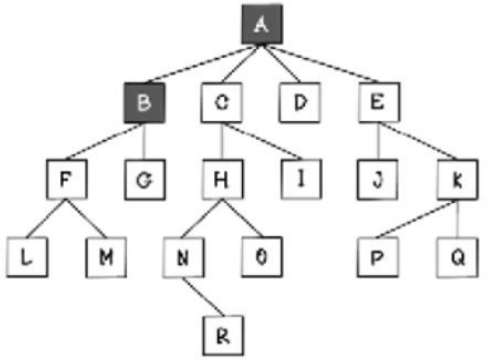
- breadth-first search (BFS) : busca primero en amplitud
- depth-first search (DFS): búsqueda en profundidad

Búsqueda en amplitud, BFS

- **Búsqueda primero en amplitud: mirar a lo ancho antes de mirar a lo profundo**

1

Visit the first child node, B.

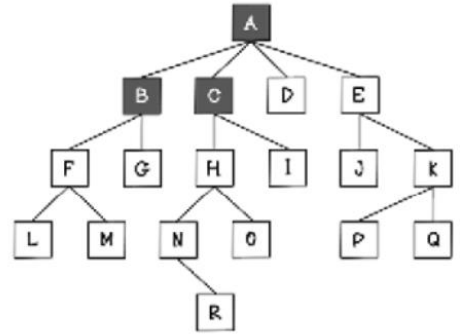


Sequence of processing the queue

A
B

2

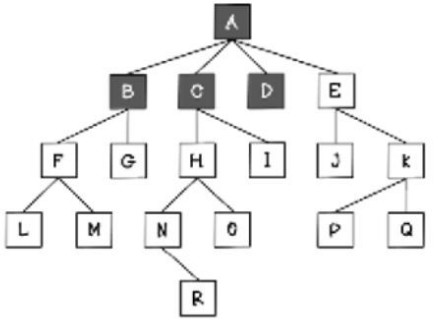
Visit the next child node, C.



A
B
C

3

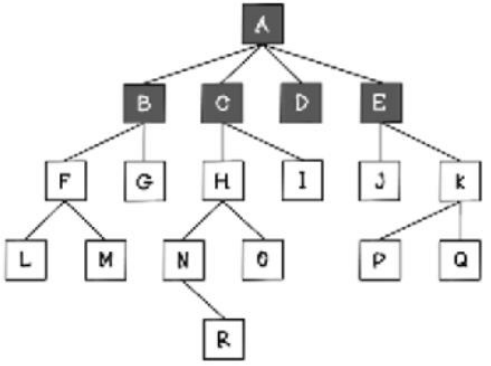
Visit the next child node, D.



A
B
C
D

4

Visit the final node on this depth, E.

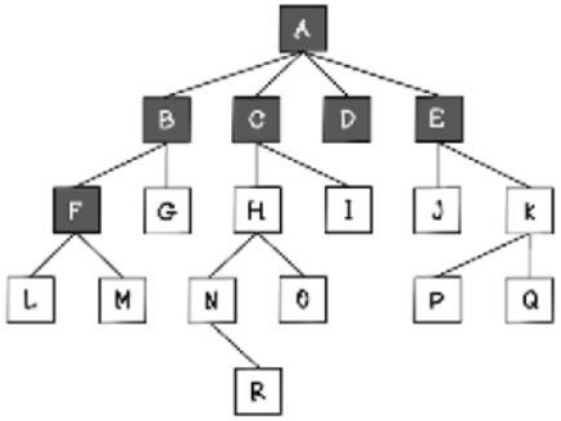


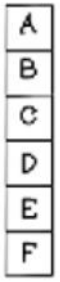
Sequence of processing the queue

A
B
C
D
E

5

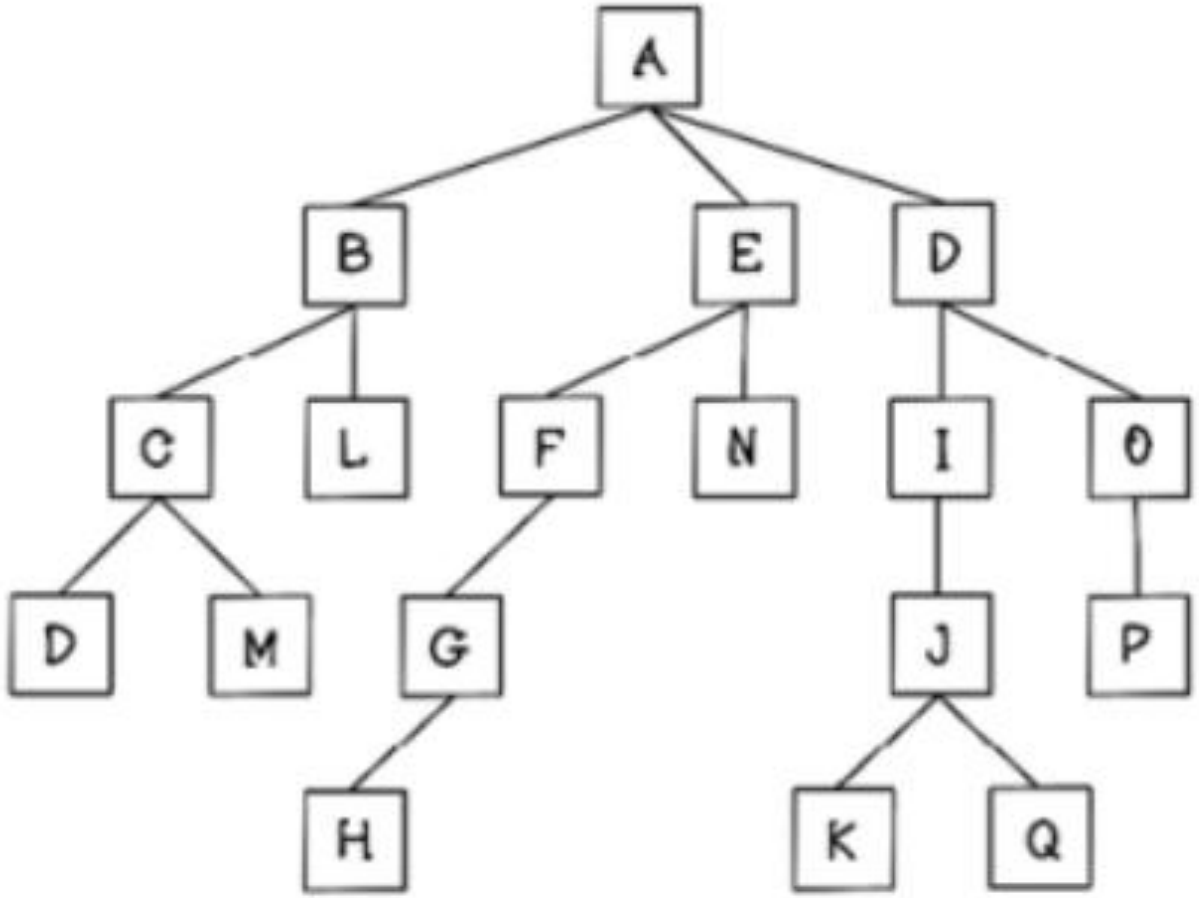
Visit the first child of the first neighbor of A. This is F, first child of B.



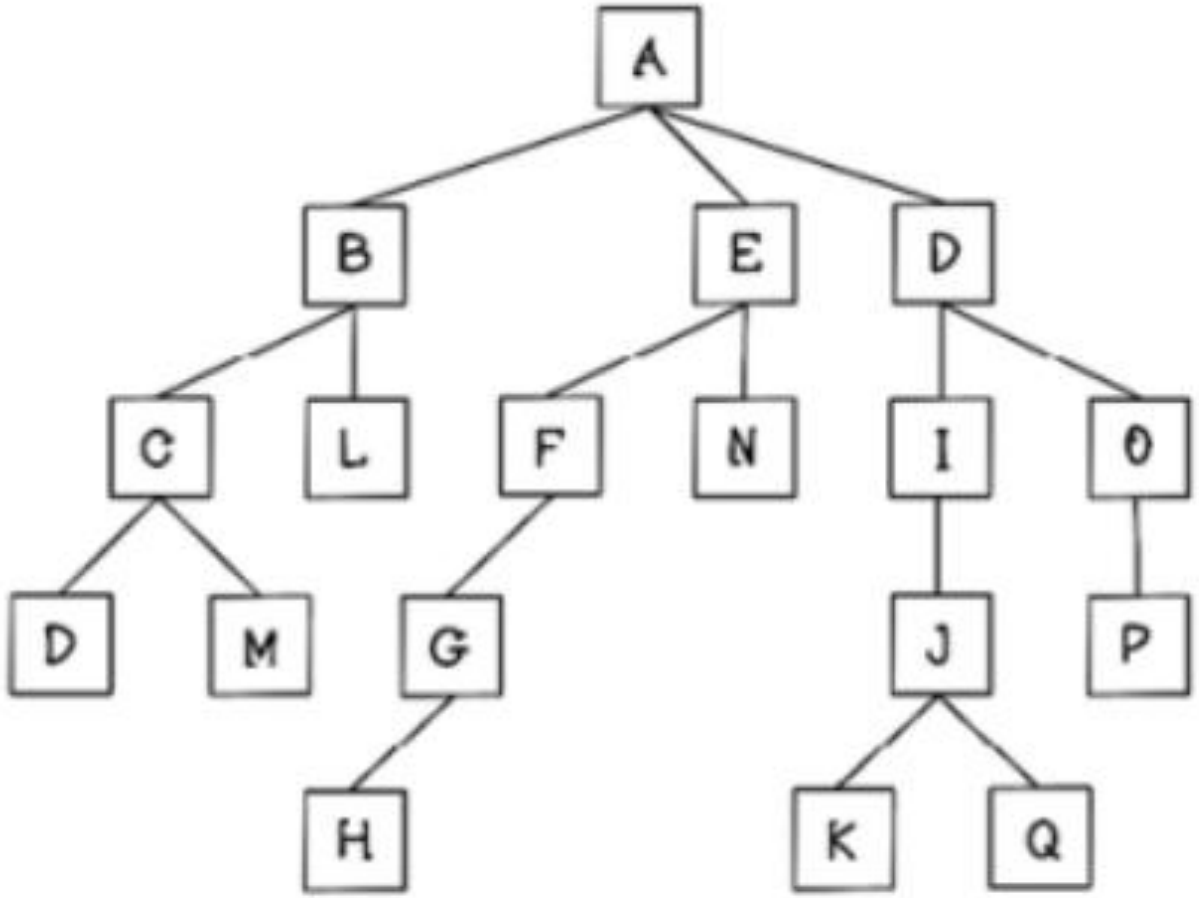


Etc.....

Otro Ejemplo:

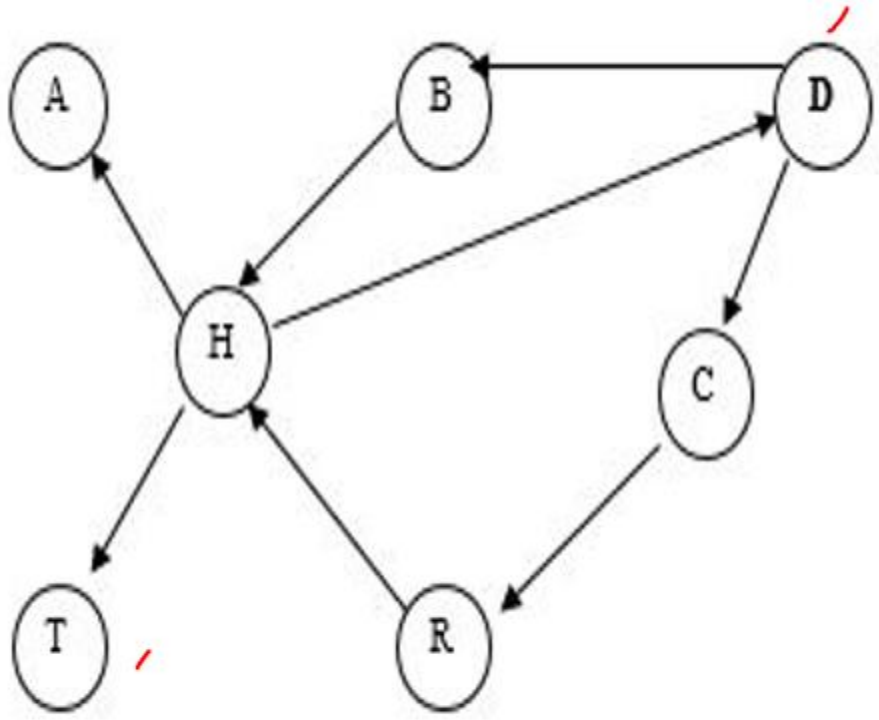


Otro Ejemplo:



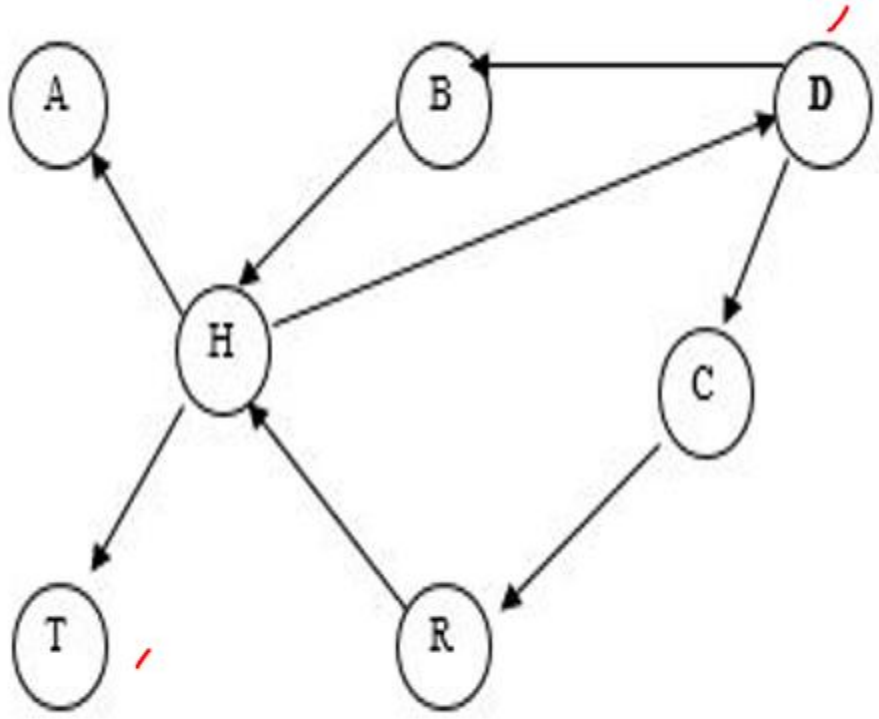
Breadth-first search order:
A, B, E, D, C, L, F, N, I, O, D, M, G, J, P, H, K, Q

Ir de D hasta T por Amplitud

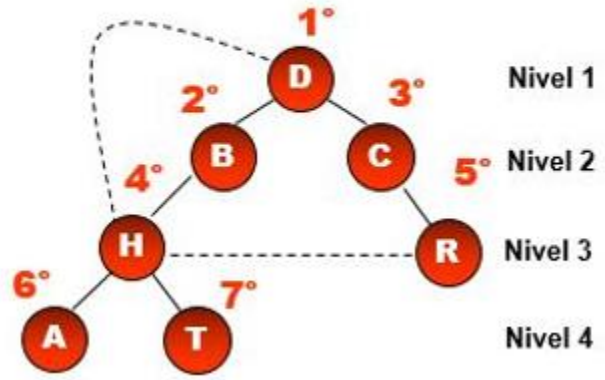


Grafo dirigido

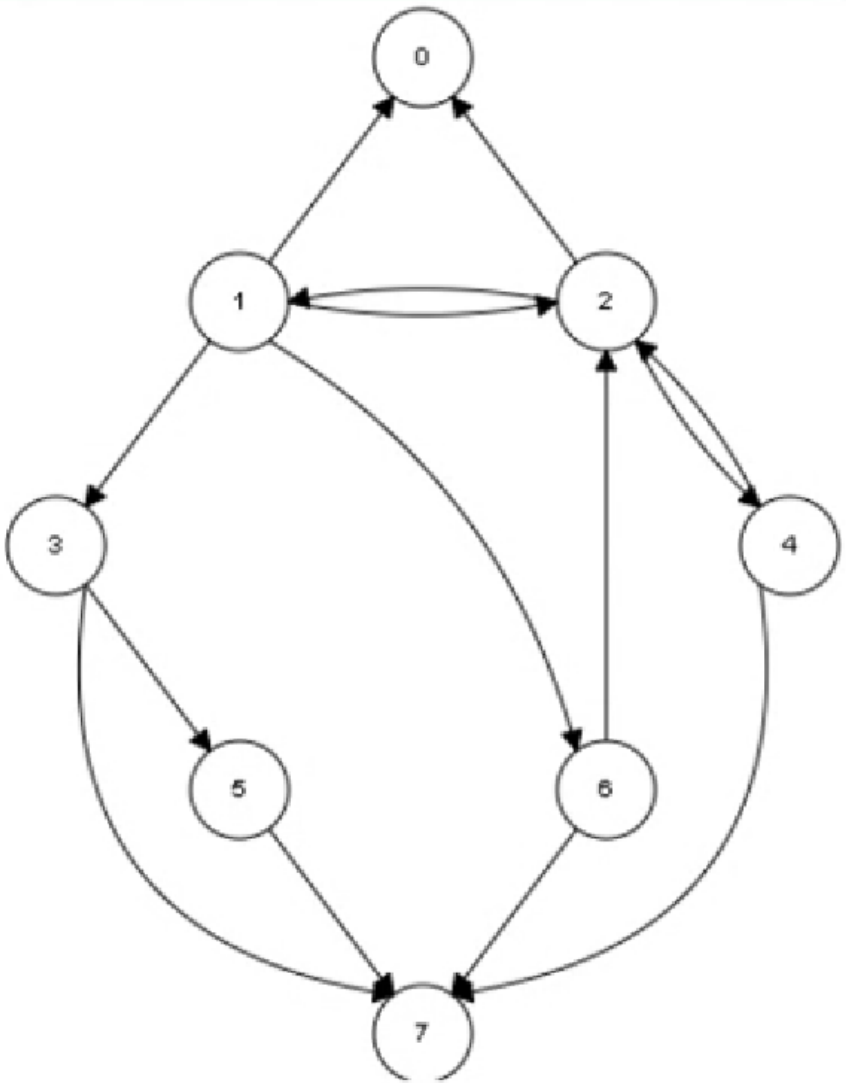
Ir de D hasta T por Amplitud



Grafo dirigido



D ={D, B, C, H, R, A, T }



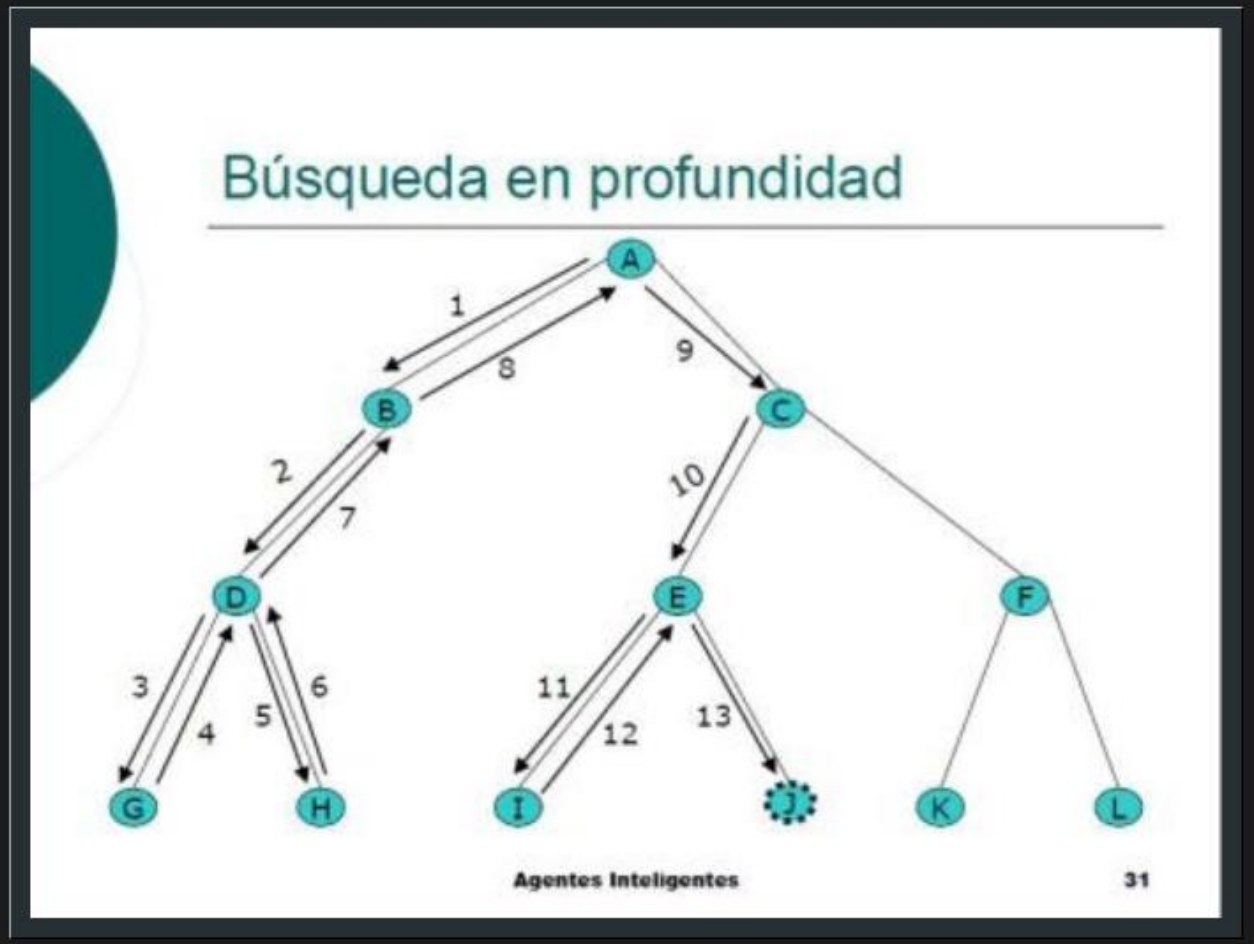
Amplitud nodo inicial 0 y nodo final 7

Búsqueda en profundidad primero: mirar en profundidad antes de mirar a lo ancho

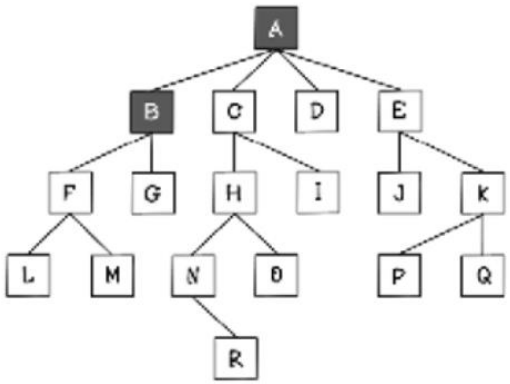
- *La búsqueda en profundidad* es otro algoritmo que se utiliza para atravesar un árbol o generar nodos y rutas en un árbol.
- Este algoritmo comienza en un nodo específico y explora las rutas de los nodos conectados del primer hijo, haciendo esto recursivamente hasta que llega al nodo hoja más alejado antes de retroceder y explorar otras rutas a la hoja nodos a través de otro nodo que han sido visitados.

- Algoritmo búsqueda en profundidad

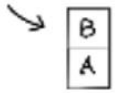
Una **Búsqueda en profundidad** es un algoritmo que permite recorrer todos los nodos de un grafo o árbol (teoría de grafos) de manera ordenada, pero no uniforme. Su funcionamiento consiste en ir expandiendo todos y cada uno de los nodos que va localizando, de forma recurrente, en un camino concreto. Cuando ya no quedan más nodos que visitar en dicho camino, regresa (Backtracking), de modo que repite el mismo proceso con cada uno de los hermanos del nodo ya procesado.



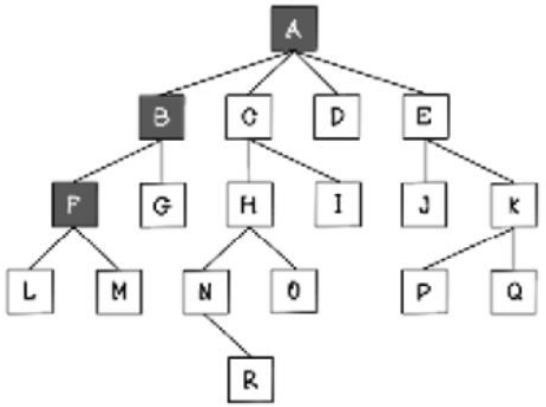
Similarly to breadth-first search, visit the first child of node A. This is B.

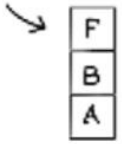


Sequence of processing the stack

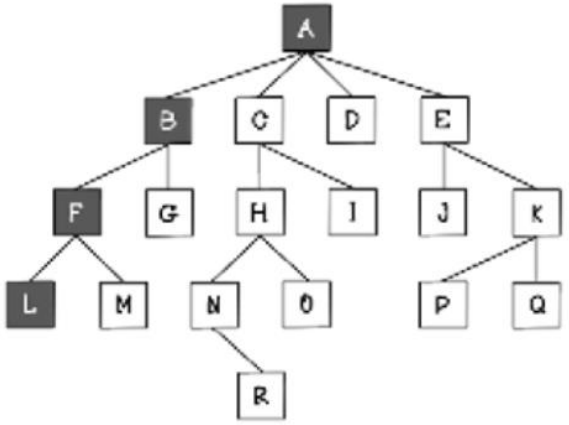


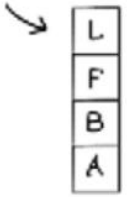
Instead of visiting other child nodes of A, the first child of B is visited – in this case, F.



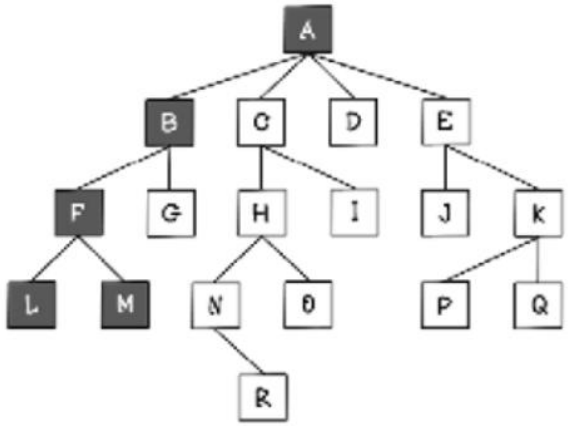


Again, the first child of F is visited. This is L.

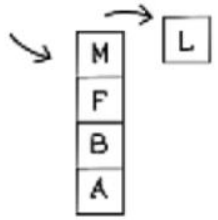




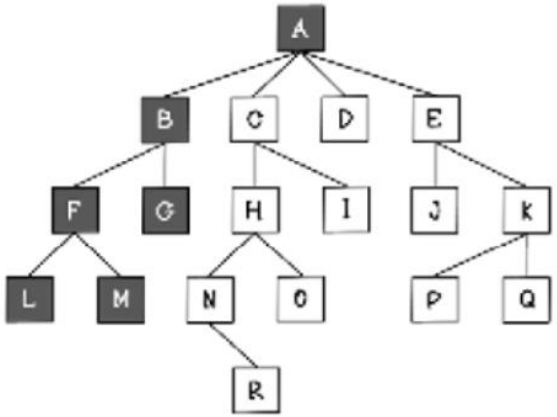
Because L is a leaf node (it has no children), the algorithm backtracks to visit the next child of F, which is M.

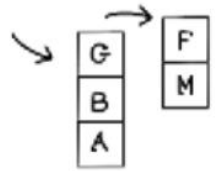


Sequence of processing the stack

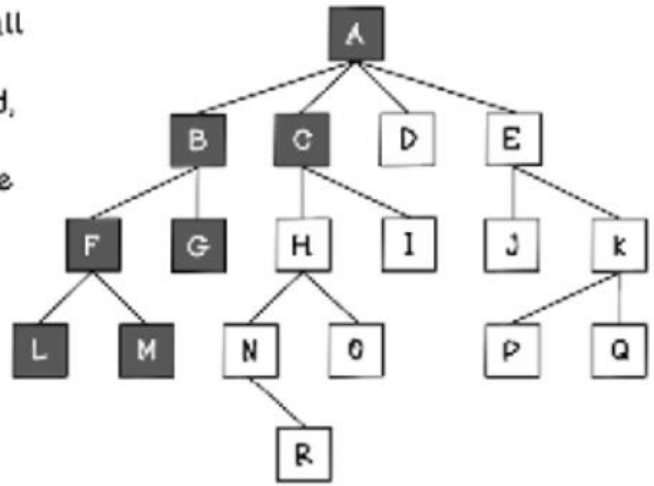


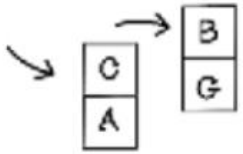
Because M is a leaf node, the algorithm backtracks to visit the next child of B. Because F has no unvisited children, this child is G.



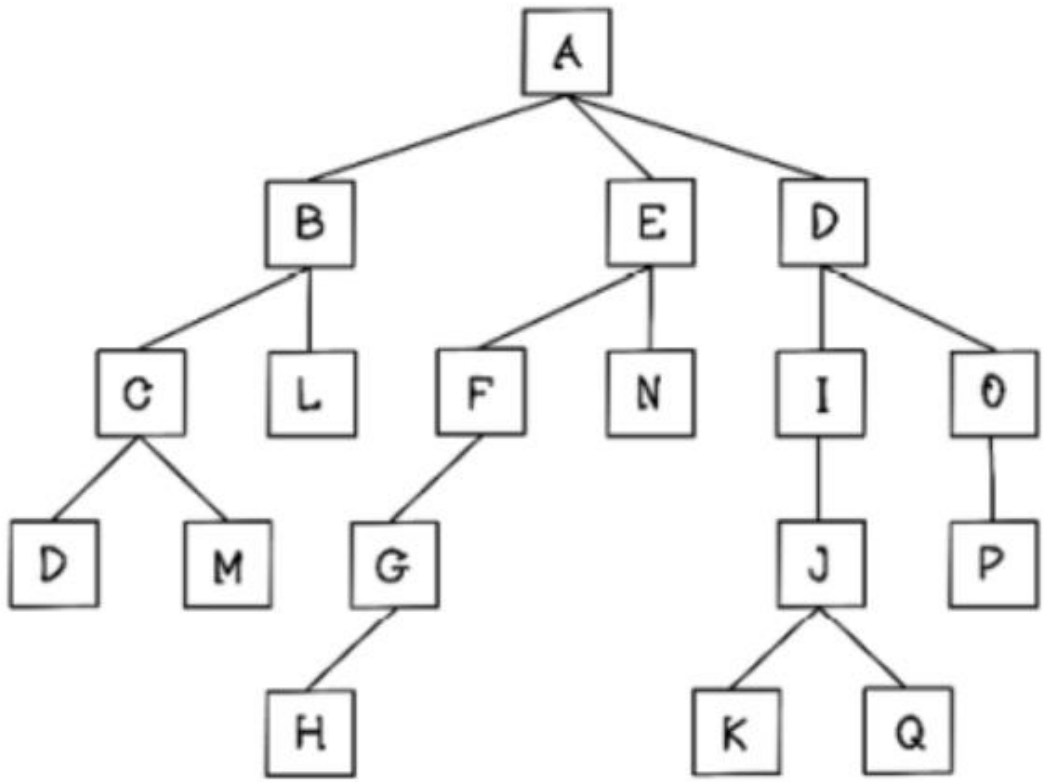


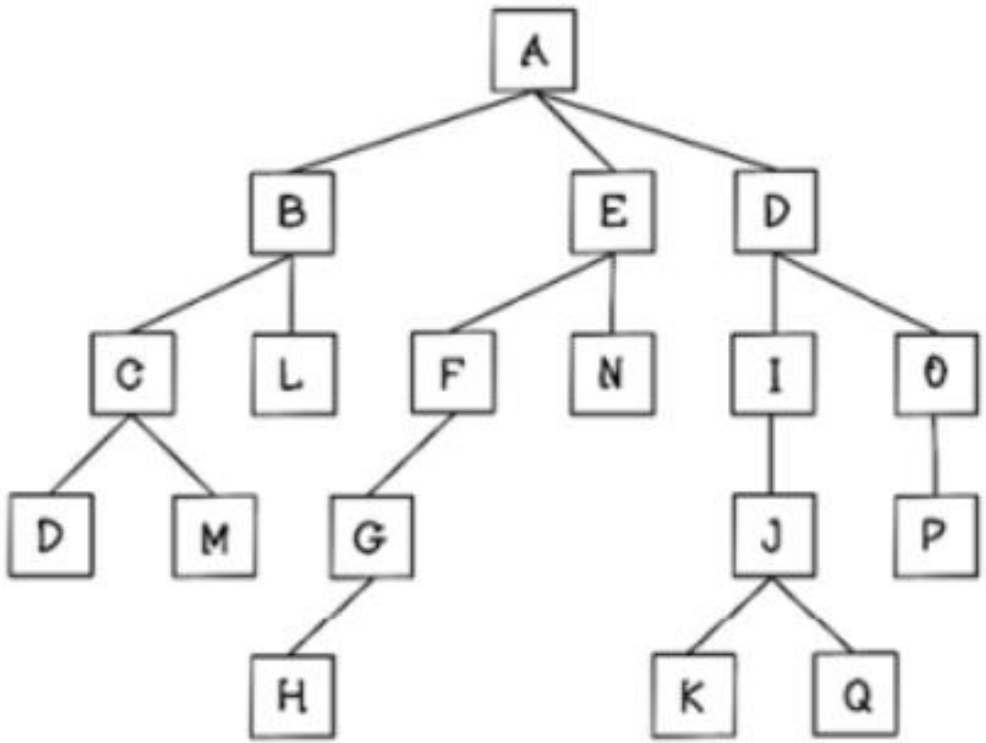
Finally, because all children of B have been visited, the algorithm backtracks to the next child of A, which is C.



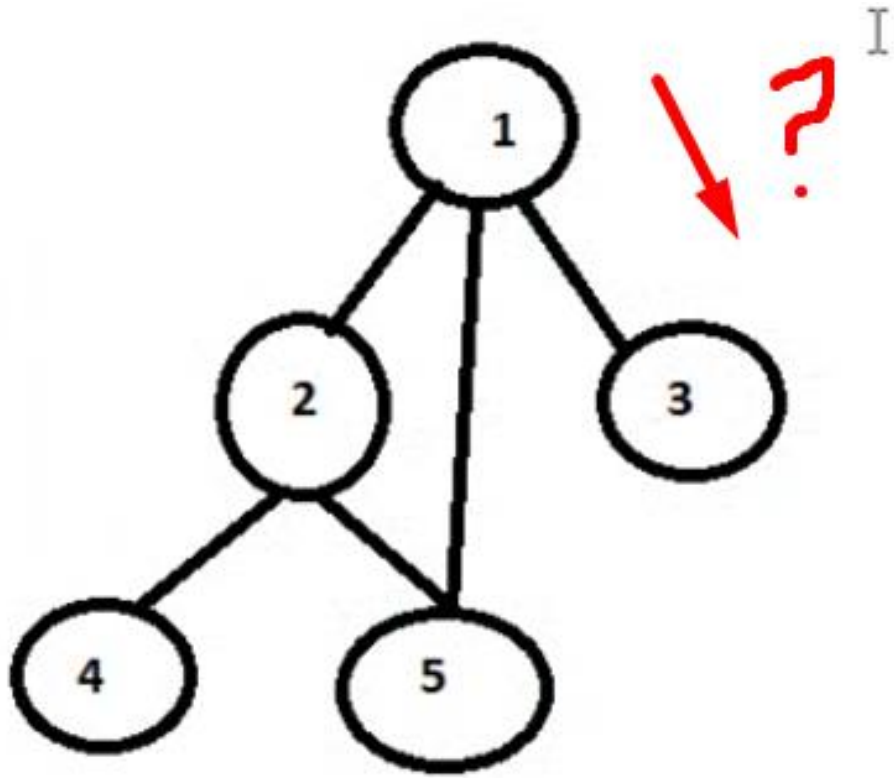


¿Cuál sería el orden de las visitas en profundidad-primero buscar el siguiente árbol?

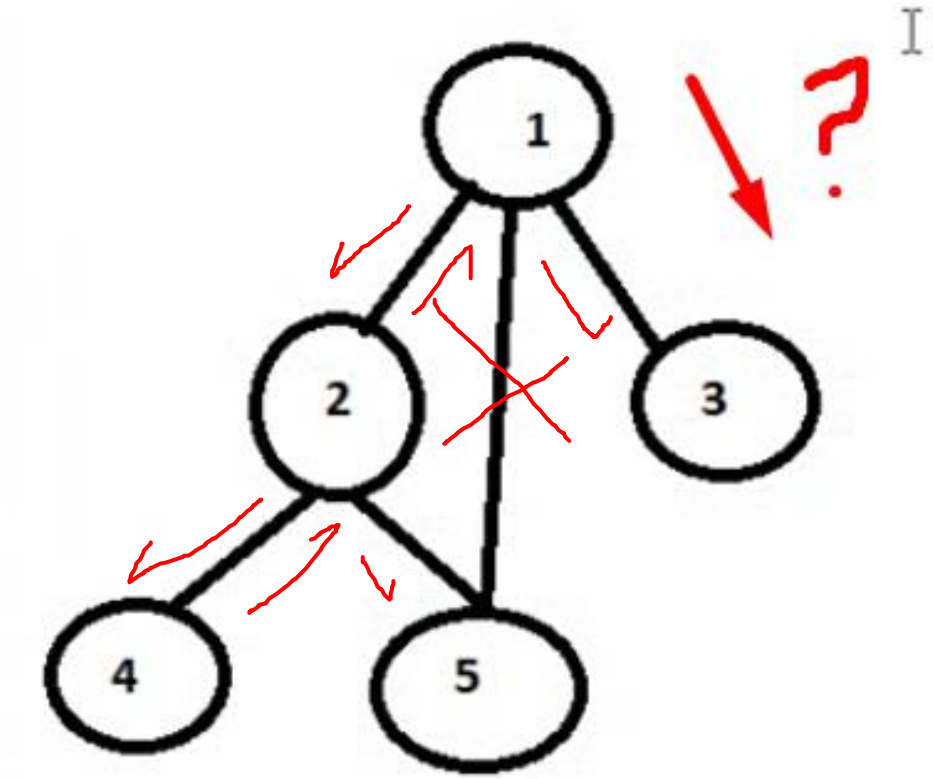




Depth-first search order:
A, B, C, D, M, L, E, F, G, H, N, D, I, J, K, Q, O, P

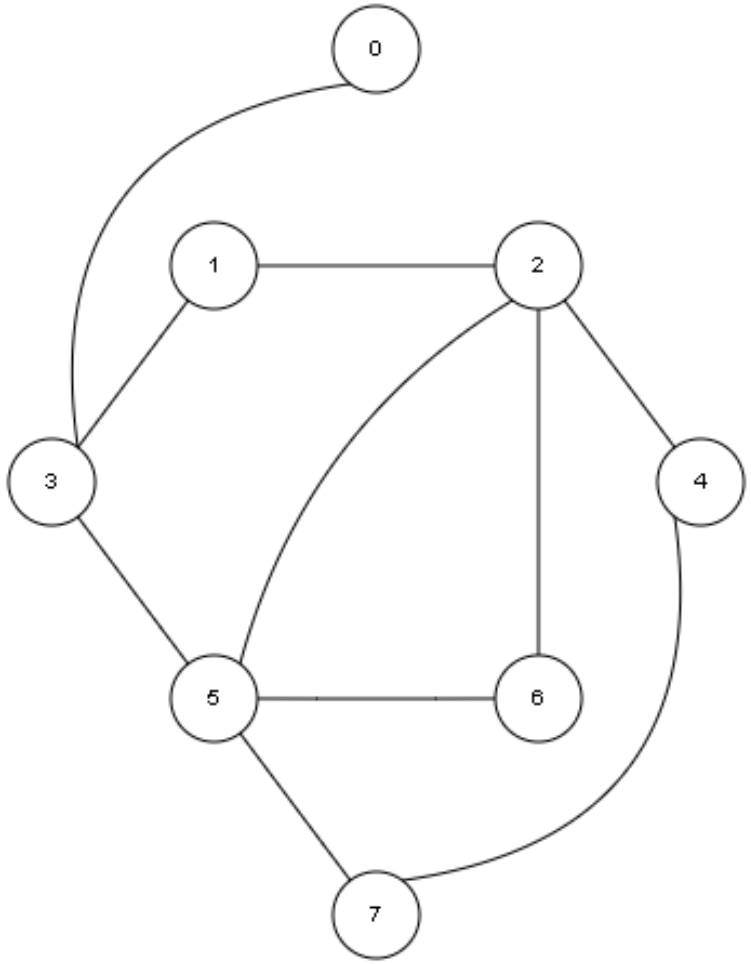


Recorrido en Profundidad?



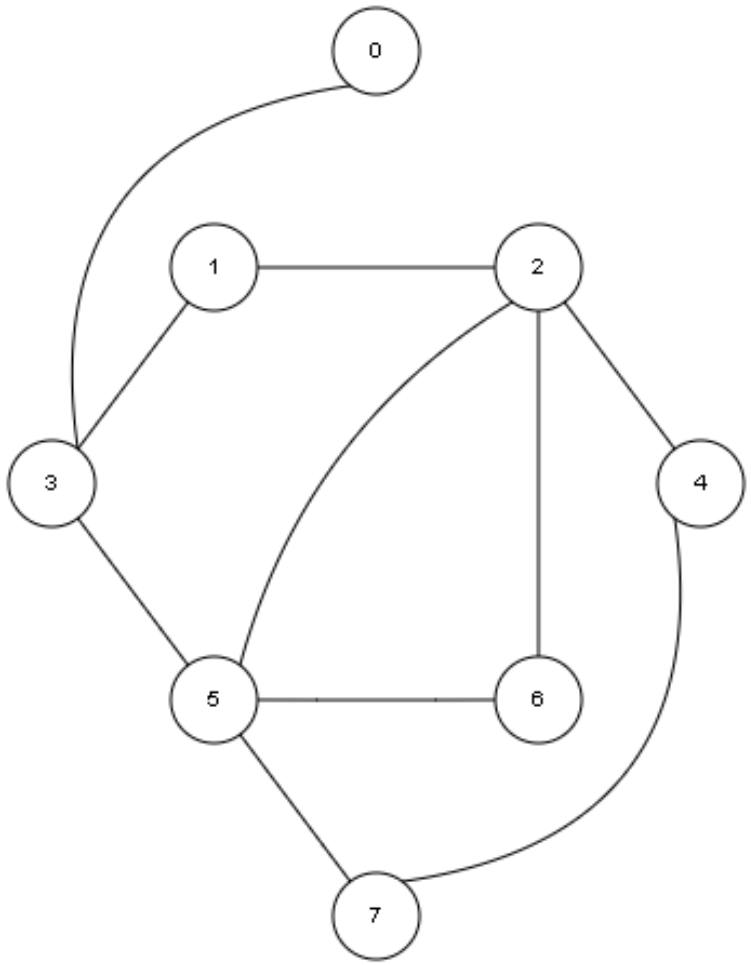
1 -2 -4 – 5- 3

Recorrido en Profundidad

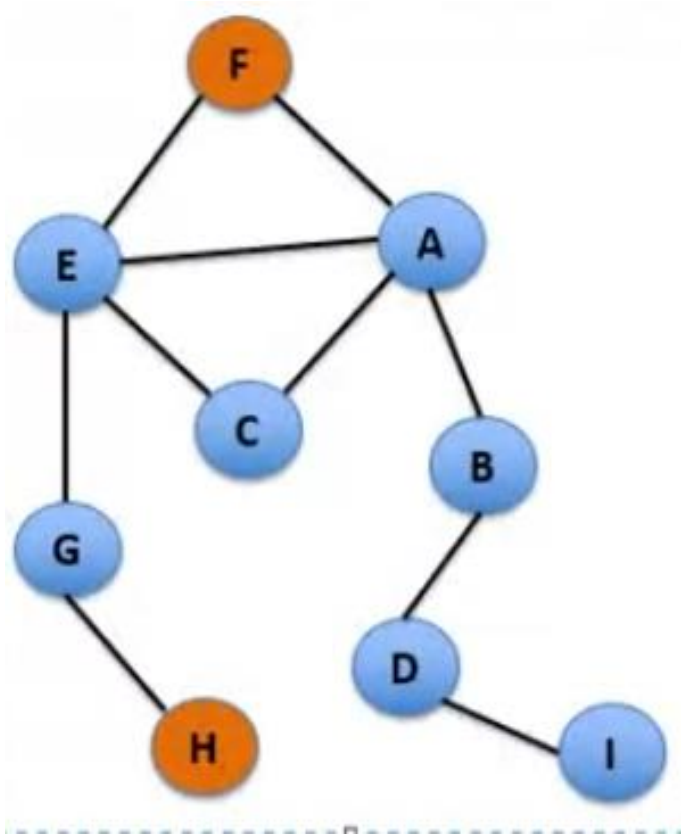


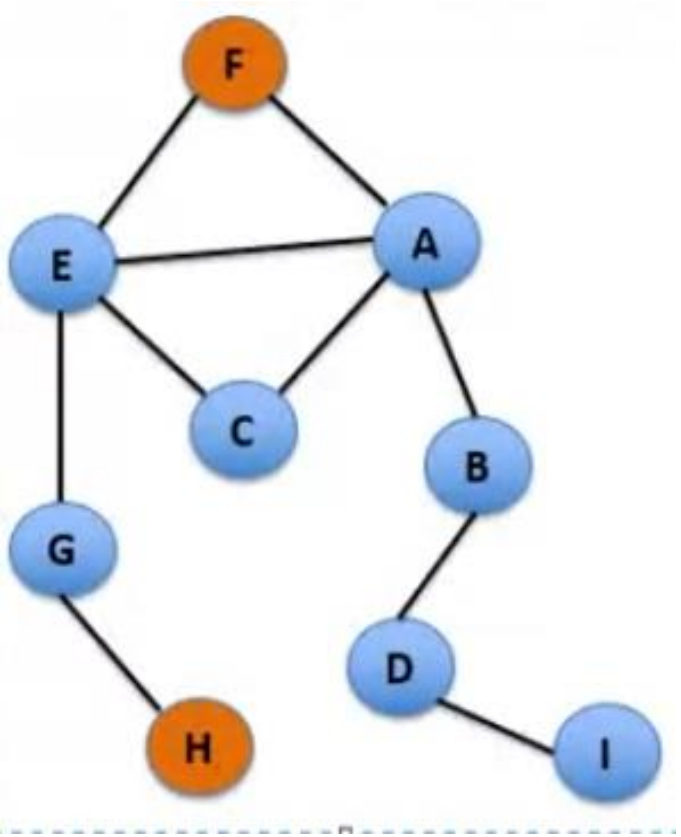
Profundidad nodo inicial 0 a nodo final 7

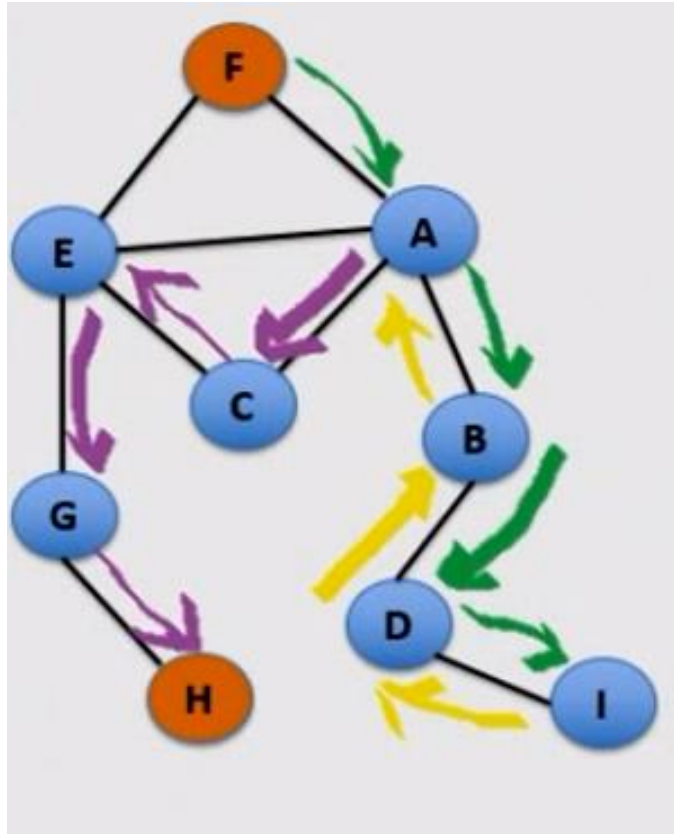
Profundidad de 0 a 7

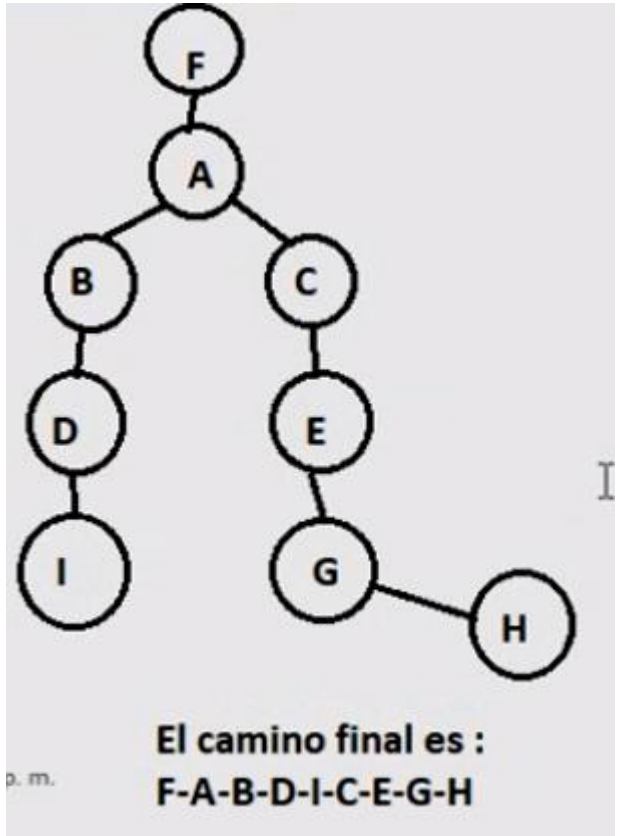


0
3
1
2
4
7


F
E
A
C
B
G
D
H
I


F
E
A
C
B
G
D
H
I


F
E
A
C
B
G
D
H
I


F
A
B
C
D
E
I
G
H
El camino final es :
F-A-B-D-I-C-E-G-H

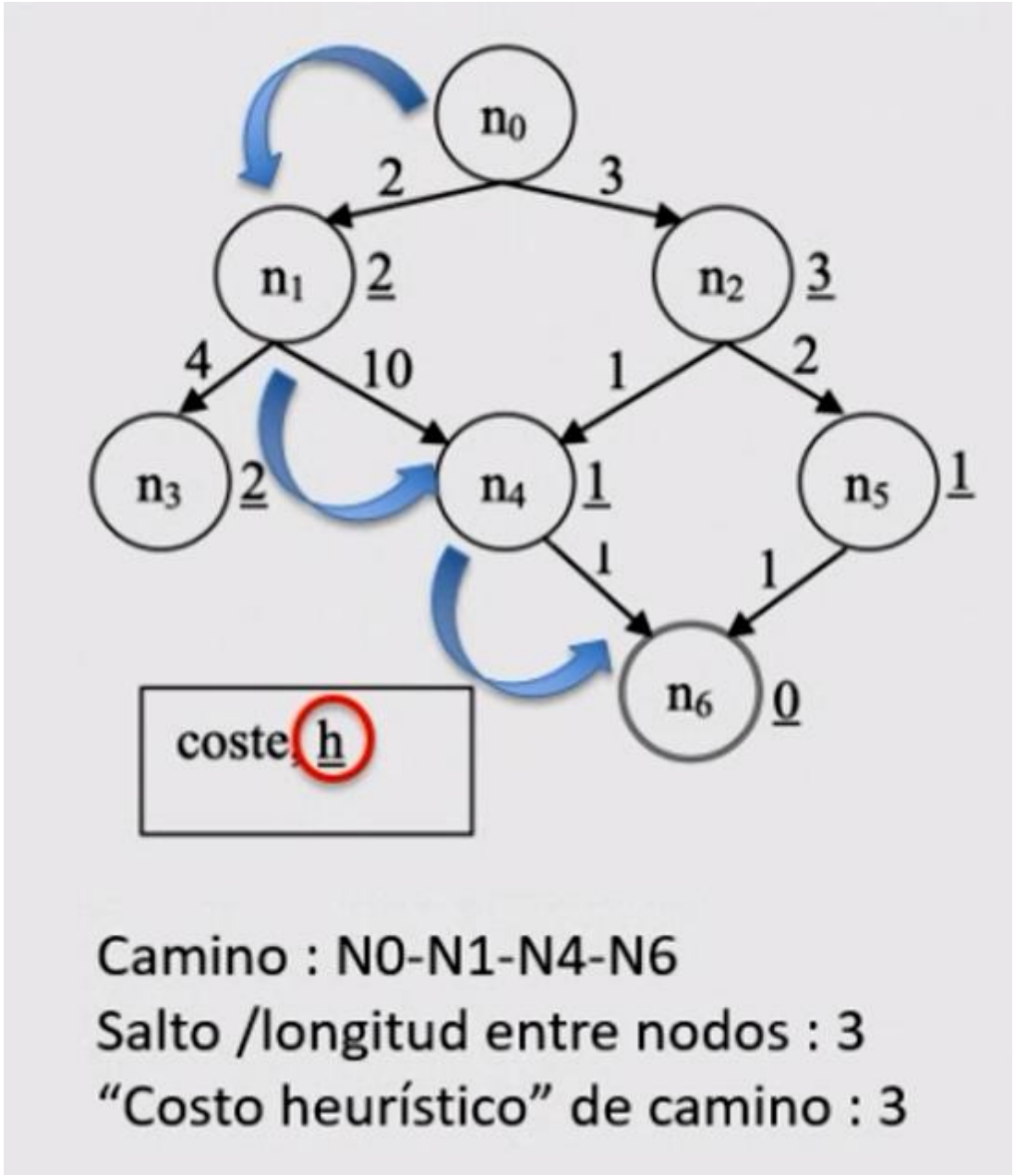
Casos de Uso de Alg. No Informados

- *Encontrar rutas entre nodos en una red*
- *Rastreo de páginas web*
- *Búsqueda de conexiones de redes sociales*

Método heurístico

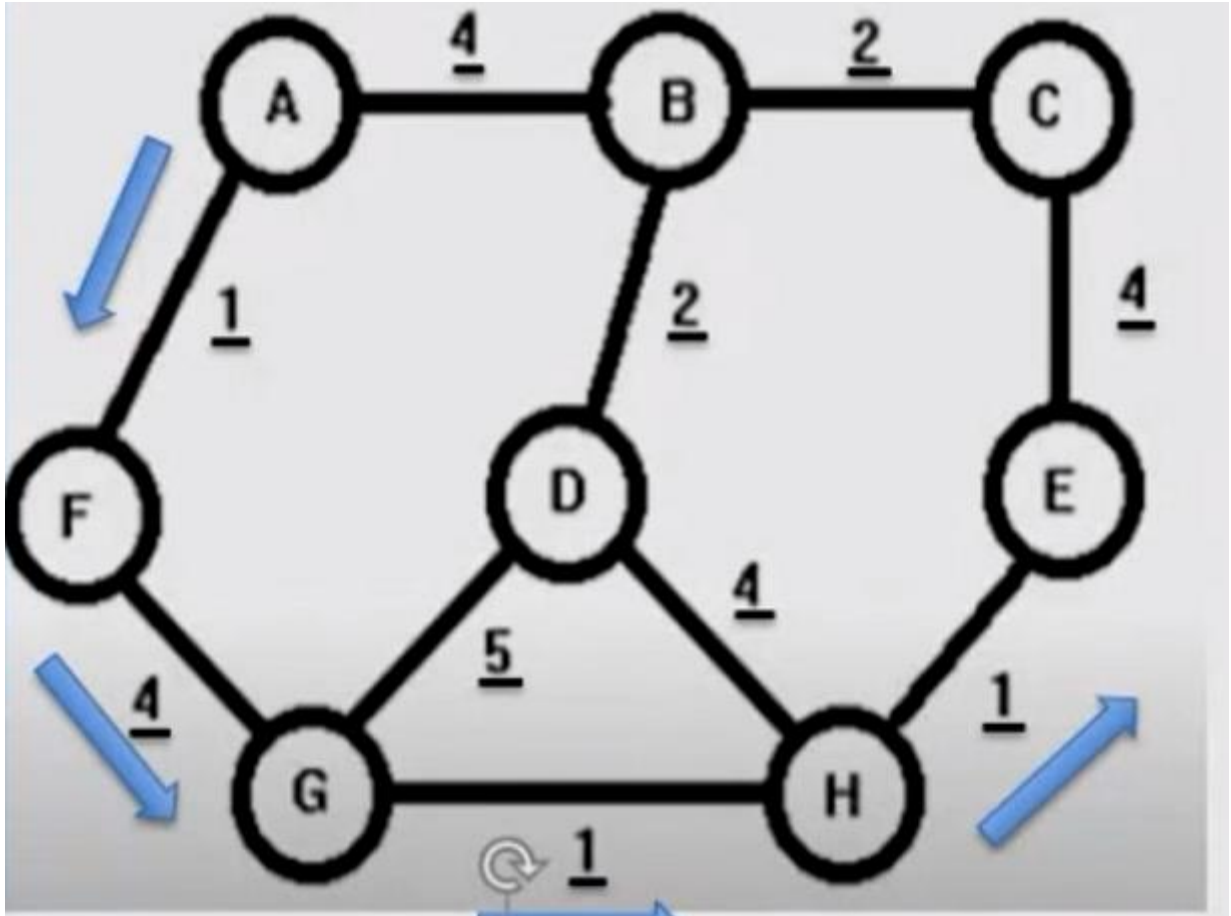
Heurística

- Las heurísticas son criterios, métodos o principios para **decidir cuál de entre varias acciones promete ser la mejor para alcanzar una determinada meta.**
- En la generación del árbol de búsqueda, nos podemos guiar con heurísticas que nos den una indicación acerca de cómo de bueno o prometedor es un determinado estado u operador.
- El uso de heurísticas nos permite guiar nuestra búsqueda de una solución, lo que nos permitirá obtener una solución más rápidamente que si utilizásemos estrategias de búsqueda a ciegas.



2

Heurística



Camino : A-F-G-H-E

Salto /longitud entre nodos : 4

“Costo heurístico” de camino : 7

Búsqueda de Costo Uniforme

Algoritmo no informado

¿Qué es el Algoritmo de Costo Uniforme?

Es un algoritmo de **búsqueda no informada** utilizado para recorrer sobre grafos el camino de costo mínimo entre un nodo raíz y un nodo destino.

La búsqueda comienza por el nodo raíz y continúa visitando el siguiente nodo que tiene menor costo total desde la raíz.

Los nodos son visitados de esta manera hasta que el nodo destino es alcanzado.

Puntos Importantes

- Se caracteriza por buscar el camino con el costo mas pequeño
- Debido a su forma de trabajo, no se preocupa por el numero de pasos, si no por el costo final

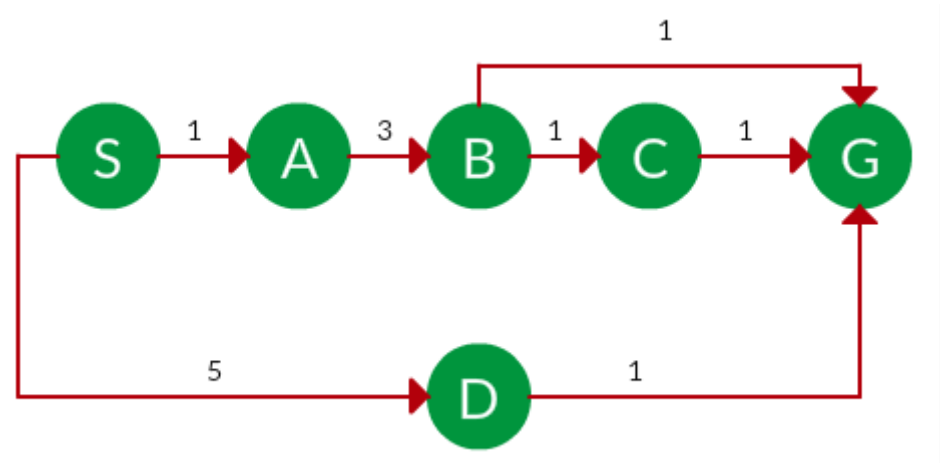
FUNCION DE COSTO

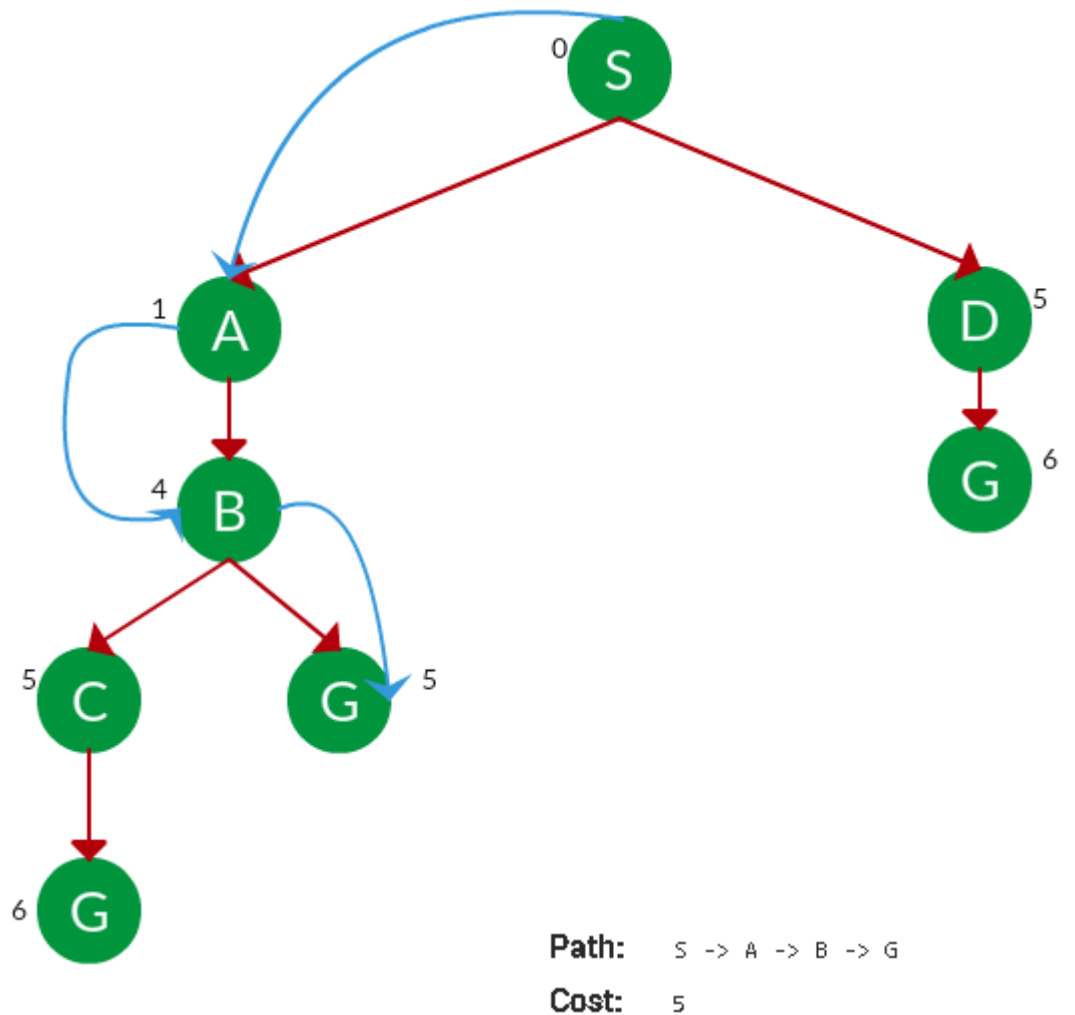
- g(n) : costo de la ruta para trasladarse del estado inicial al estado n

Proceso o Acciones:

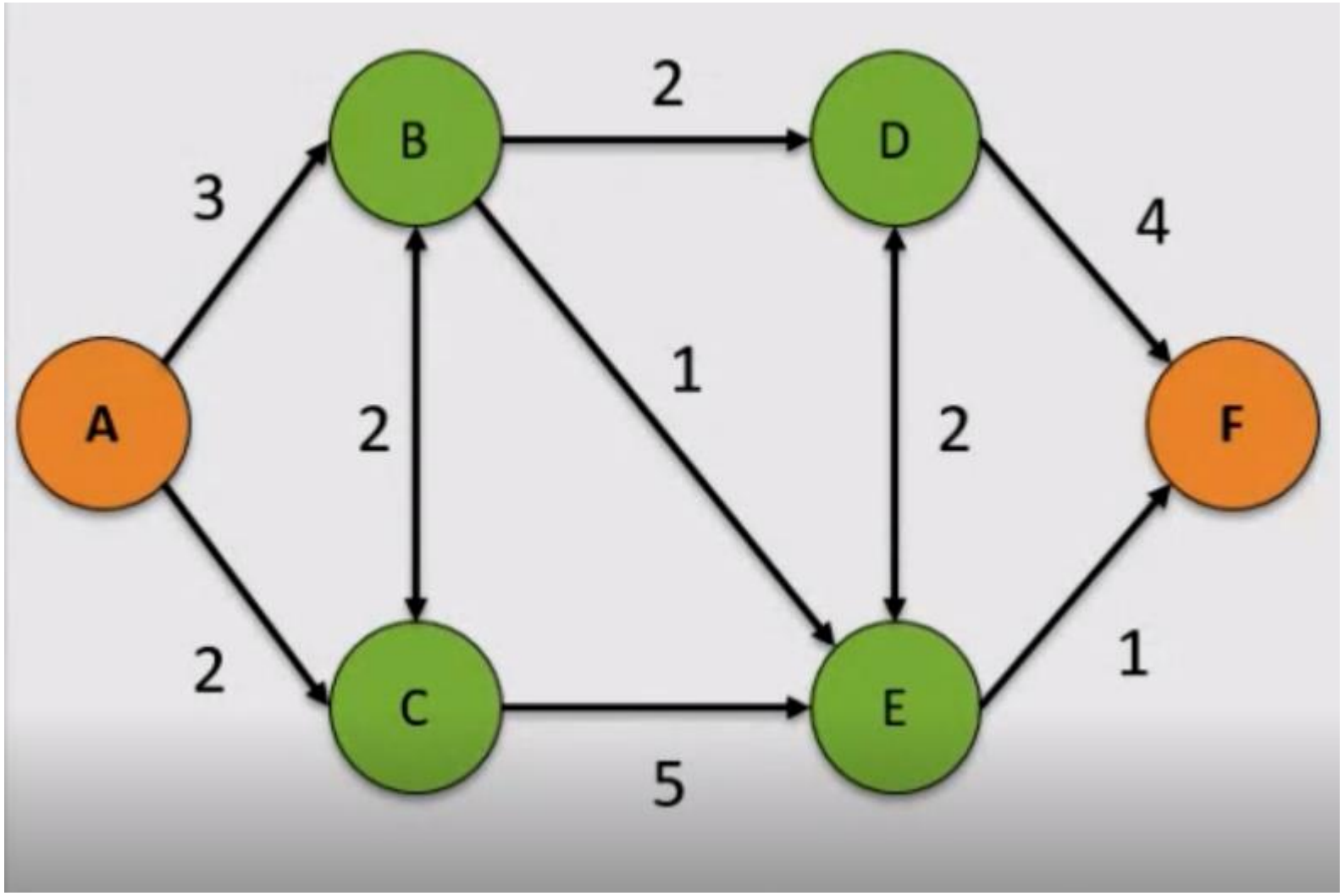
- Trasladarse de un nodo a otro adyacente

Costo Uniforme

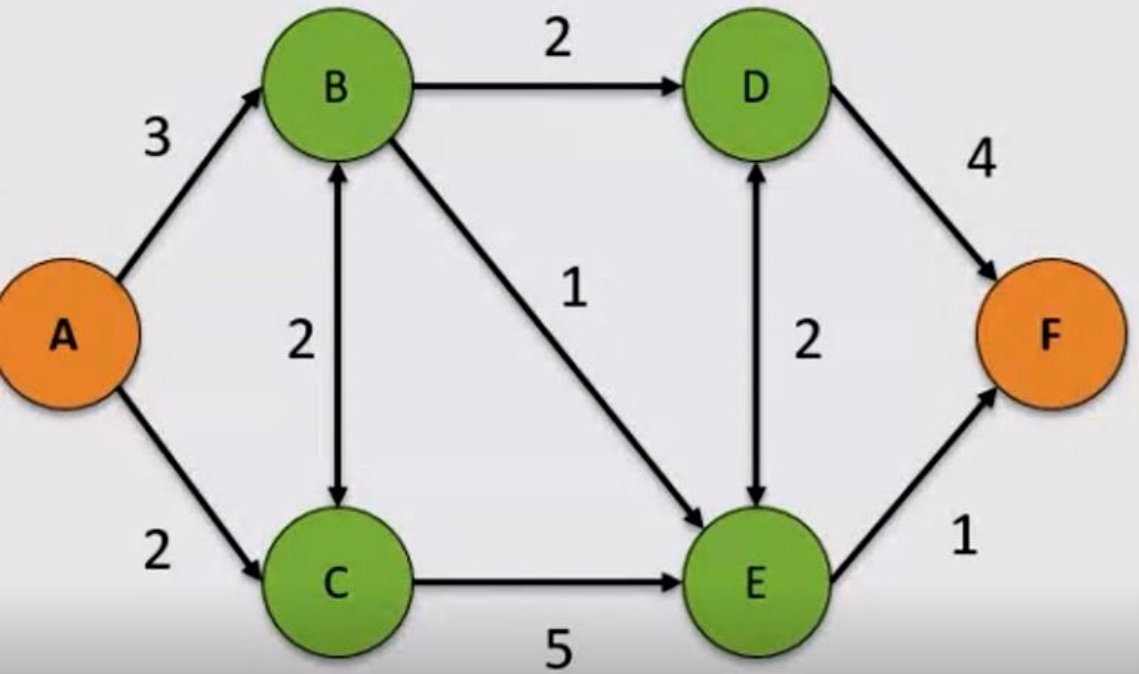




De A hacia F



consideración realizar la frontera



A) Orden nodos expandidos

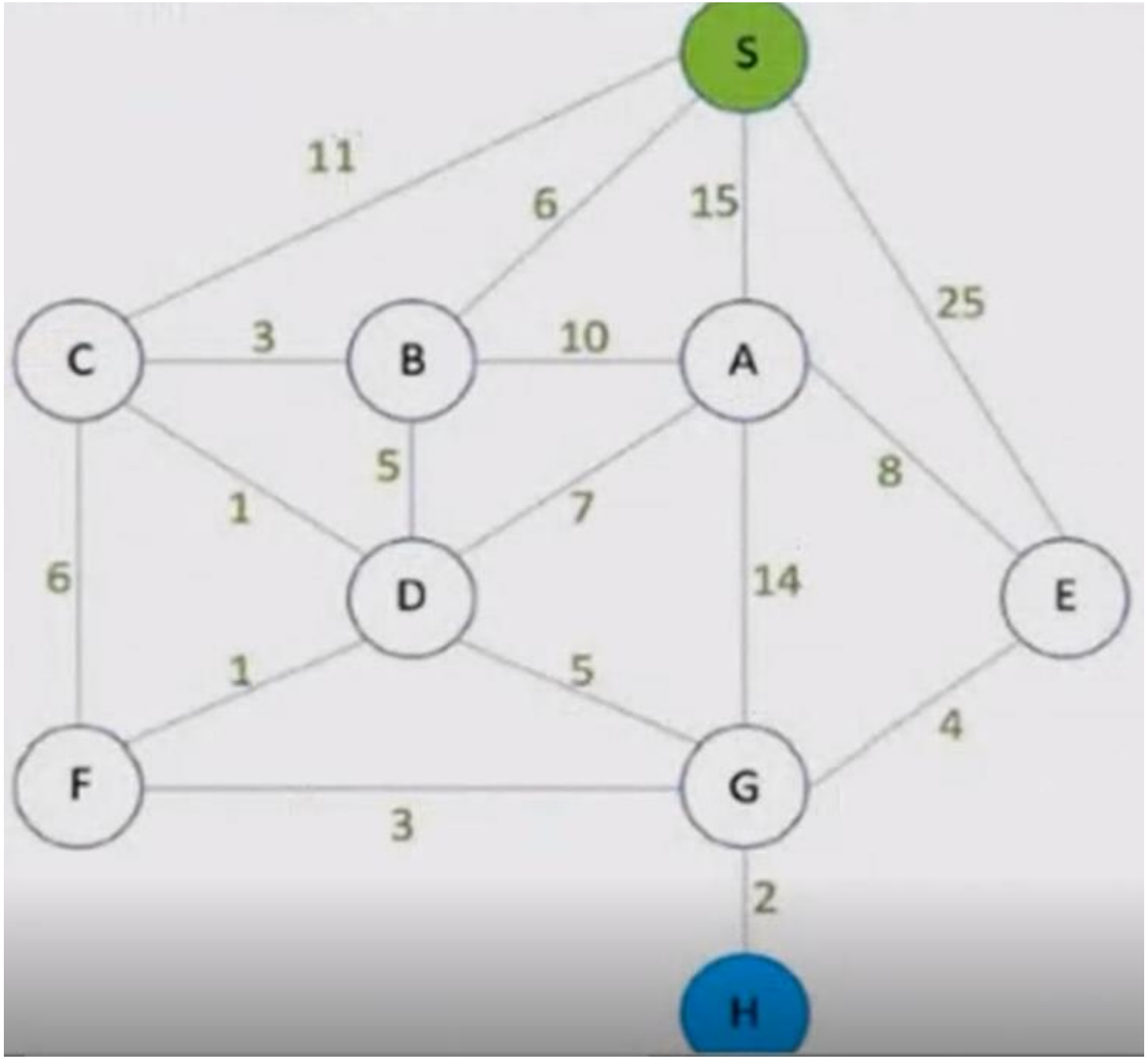
Front1	Zona Explorada
A(0)	( )
C(2) , B(3)	(A)
B(3) , ~~B(4)~~ , E(7)	(A,C)
E(4) , D(5) , ~~E(7)~~	(A,C,B)
D(5),F(5),~~D(6)~~	(A,C,B,E)
F(5) , ~~F(9)~~	(A,C,B,E,D)

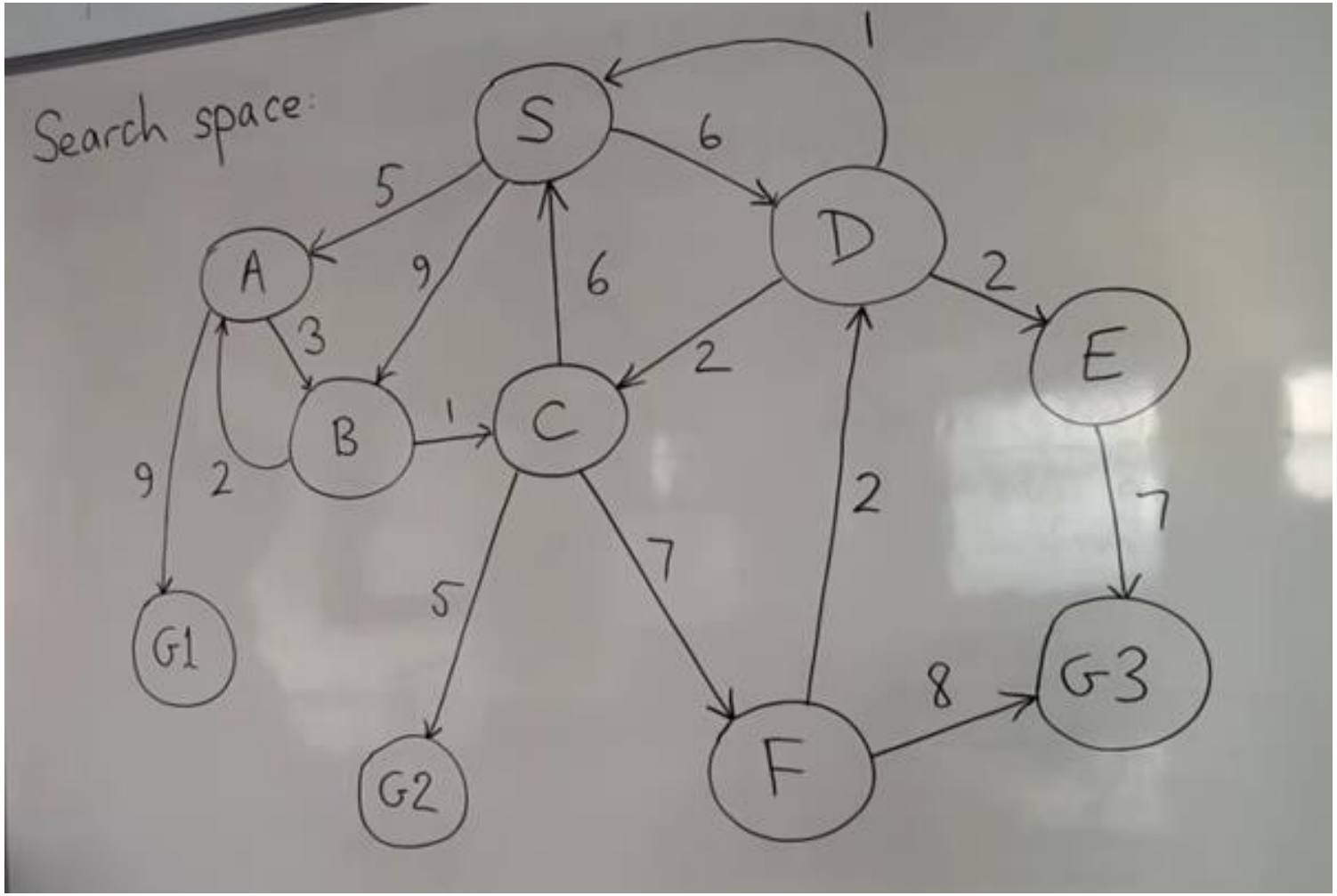
B) # nodos no expandidos: 0

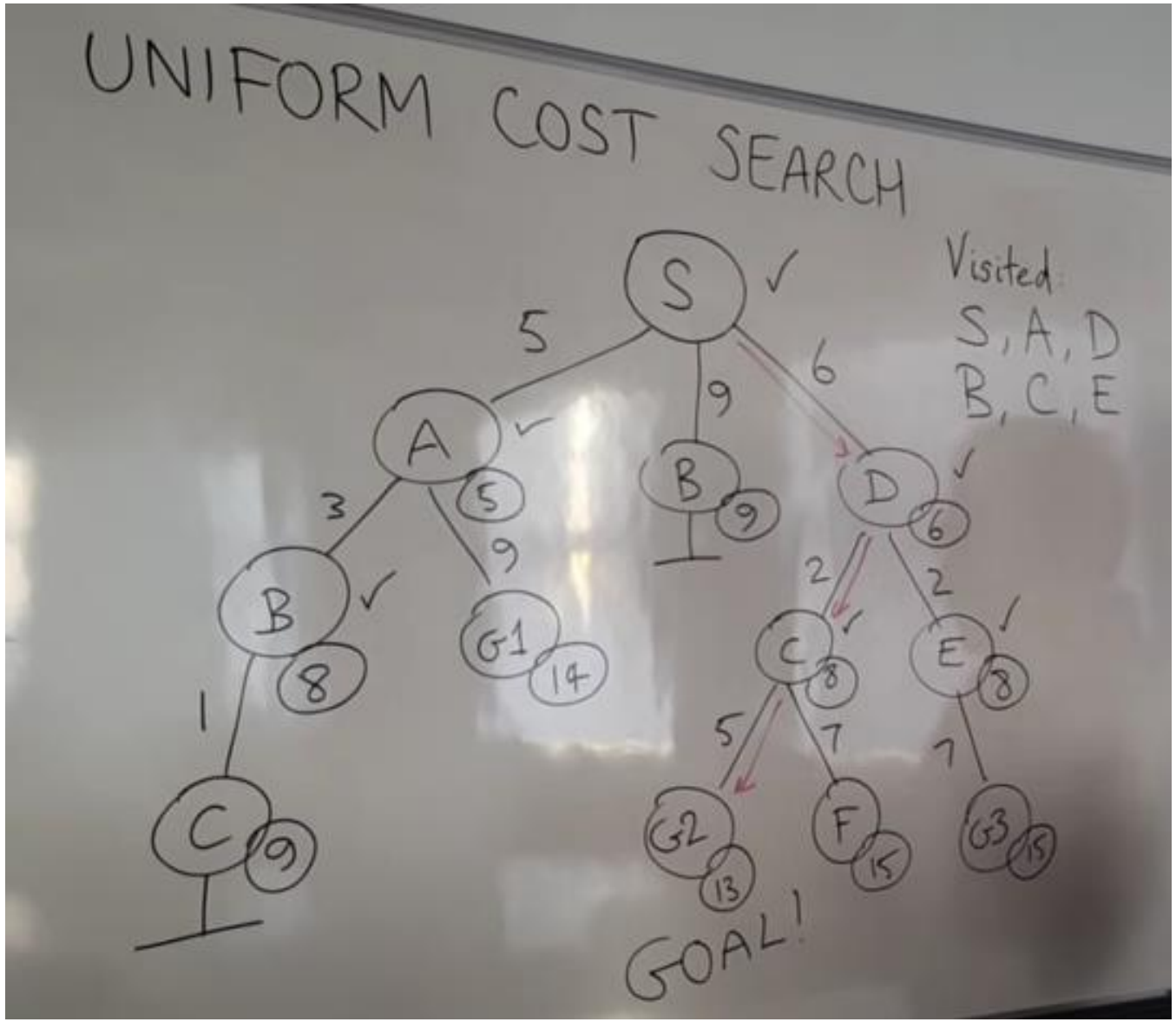
C) El camino objetivo que consume menos recursos es : A-B-E-F

D) Longitud camino objetivo: 3

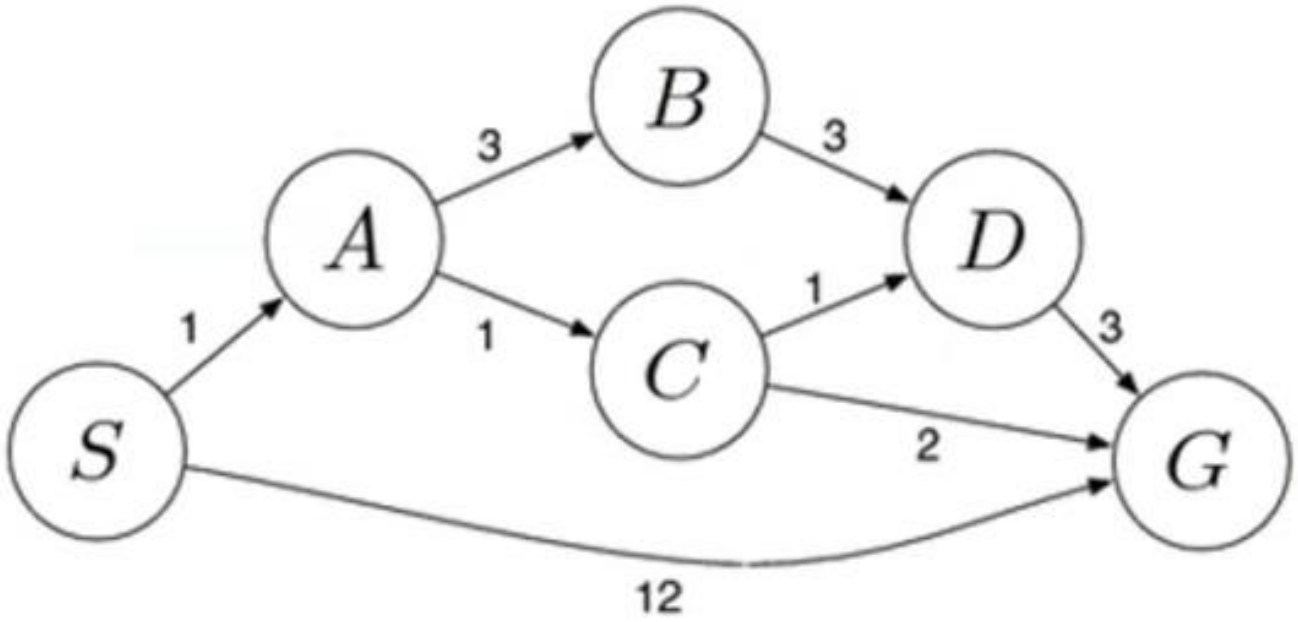
E) Costo del camino: 5


S
11
6
15
25
C
3
B
10
A
5
8
1
7
6
D
14
E
1
5
4
F
G
3
2
H


Search space:
S
A
B
C
D
E
F
G1
G2
G3
1
6
5
9
6
3
2
1
2
9
2
7
5
7
8


UNIFORM COST SEARCH
Visited:
S, A, D
B, C, E
S
A
B
D
B
G1
C
C
E
G2
F
G3
GOAL!

Cuál es la ruta de costo mínimo de A a G?

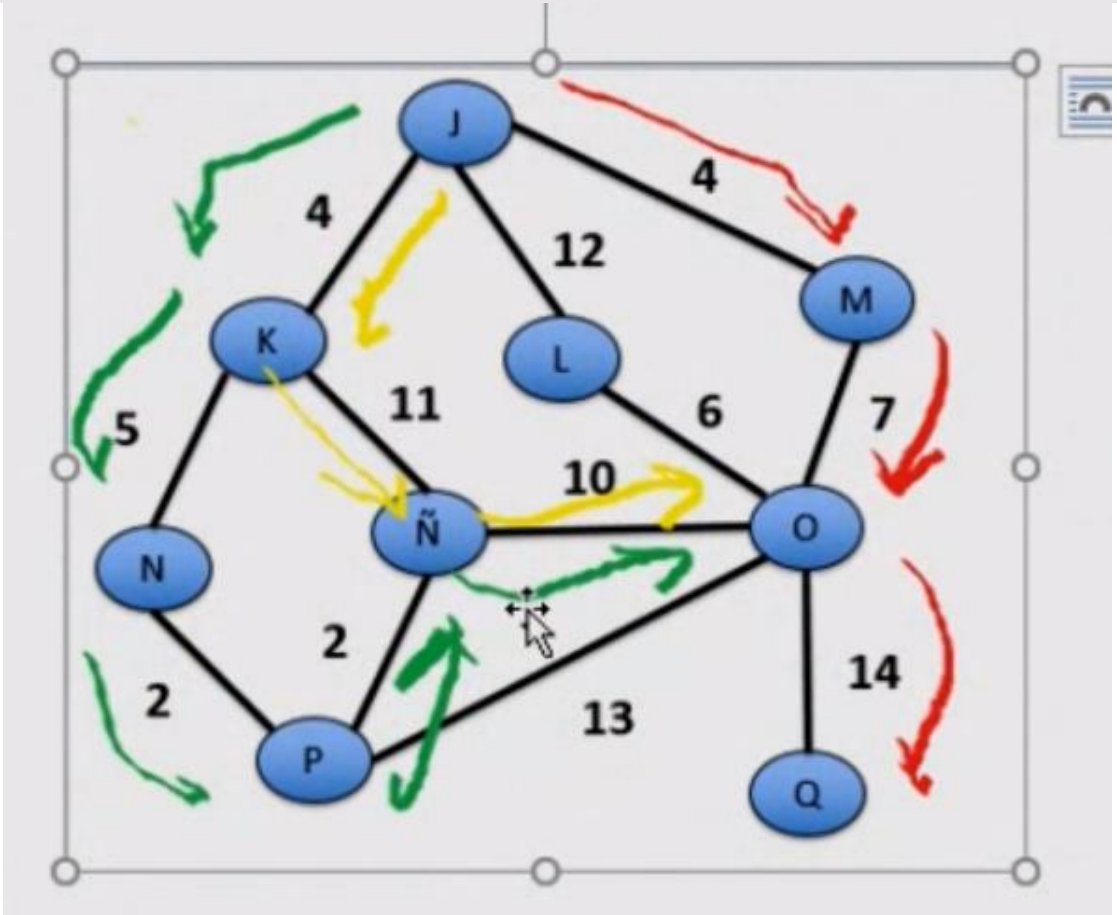


B) # nodos no expandidos: 1 → L

C) Camino objetivo : J –M – O -Q

D) Longitud camino : 3

E) Coste de camino : 25 → acumulas costes y le añades la heurística final



Aplicar el algoritmo de profundidad para ir del nodo 16 al 32 y encontrar el camino ideal