

# Busca em Profundidade

Adaptado de Humberto C. B. Oliveira

# Última aula

- Representação computacional:
  - Matriz de adjacência;
  - Matriz de incidência;
  - Lista de adjacência.

# Busca em Grafos

- Alguns **objetivos** da busca em grafos são:
  - determinar quais **vértices** são **alcançáveis** através de um vértice inicial...
  - Determinar se um determinado objeto está presente no grafo...
  - Identificar algumas características dos grafos...
- Aplicações???

# Busca em Grafos

- Aplicações:
  - Compiladores;
  - Resolução de problemas (xadrez, por exemplo);
    - Este é um exemplo de uma grande classe de problemas que são resolvidos por enumeração;
    - Ou seja, busca em grafos pode auxiliar a resolver inúmeros outros problemas combinatórios;
  - Função “localizar arquivo” no sistema operacional;
  - Detecção de *deadlocks*;
  - Dentre centenas de outras aplicações....

# Busca em Grafos

- Adaptações nos algoritmos de busca nos permitem construir algoritmos para os problemas de:
  - Árvore Geradora Mínima (AGM);
  - Caminho Mínimo;
  - Componentes Fortemente Conectados;
  - Ordenação Topológica.

# Busca em Grafos

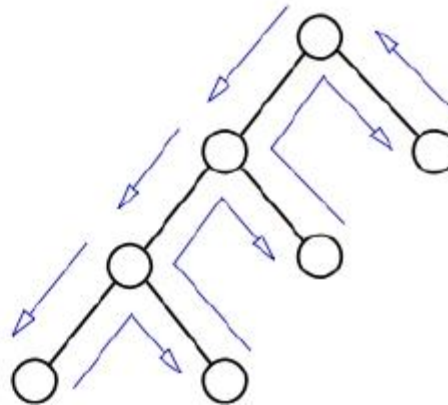
- Algoritmos clássicos de Busca:
  - Busca em Largura;
  - Busca em Profundidade;

# Busca em Profundidade

A series of horizontal lines in teal and light blue colors, with varying lengths and slight offsets, creating a layered effect across the middle of the slide.

# Busca em Profundidade

- A **busca em profundidade** (do inglês *depth-first search* - **DFS**) é um algoritmo para caminhar no grafo;
- Seu núcleo se **concentra em buscar**, sempre que possível, **o mais fundo no grafo**.



- As arestas são exploradas a partir do vértice  $v$  mais recentemente descoberto que ainda possui vértices adjacentes não explorados.



# Busca em Profundidade

- Quando todas arestas adjacentes a  $v$  tiverem sido exploradas, a busca “anda para trás” (do inglês *backtrack*) para explorar vértices do qual  $v$  foi descoberto;
- O processo continua até que sejam descobertos todos os vértices que são alcançáveis a partir do vértice original;
- Se todos os vértices já foram descobertos, então é o fim.
- Caso contrário o processo continua a partir de um novo vértice de origem ainda não descoberto (grafos desconexos).
  - Este é um ponto diferenciado da busca em árvore que vocês já conhecem;
  - Pois ao final de uma busca simples, pode haver vértices que não foram alcançados.

# Busca em Profundidade

- Legenda para algoritmo:
  - Vértice Branco – Ainda não visitado...
  - Vértice cinza – Visitado, mas seus adjacentes ainda não foram todos visitados;
  - Vértice preto – Visitado, e seus adjacentes já foram todos visitados.

# Busca em Profundidade

- Legenda para descoberta e finalização...

a



Vértice desconhecido

b



Vértice encontrado

c



Vértice encontrado, com fecho positivo totalmente visitado

- *d*: marcador do instante que o vértice c foi descoberto;
- *f*: marcador do instante que o fecho transitivo do vértice c foi totalmente visitado (considerado então finalizado).

# Busca em Profundidade

*DFS(G)*

1 *para cada vértice*  $u \leftarrow V[G]$

2      $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

4 *para cada vértice*  $u \in V[G]$

5     *se*  $cor[u] = BRANCO$

6         *DFS* – *VISIT*( $u$ )

*DFS* – *VISIT*( $u$ )

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 *para cada vértice*  $v \in Adj(u)$

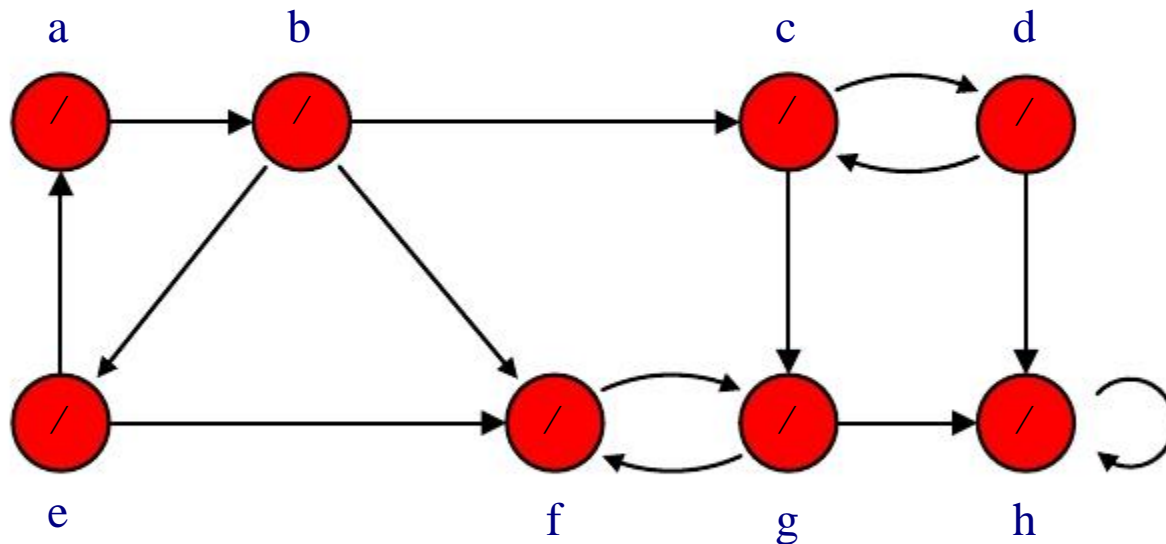
5     *se*  $cor[v] = BRANCO$

6         *DFS* – *VISIT*( $v$ )

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

# Busca em Profundidade



Lista: [c,a,b,d,e,f,g,h]

DFS(G)

1 para cada vértice  $u \leftarrow V[G]$

2     $cor[u] \leftarrow \text{BRANCO}$

3  $tempo \leftarrow 0$

4 para cada vértice  $u \in V[G]$

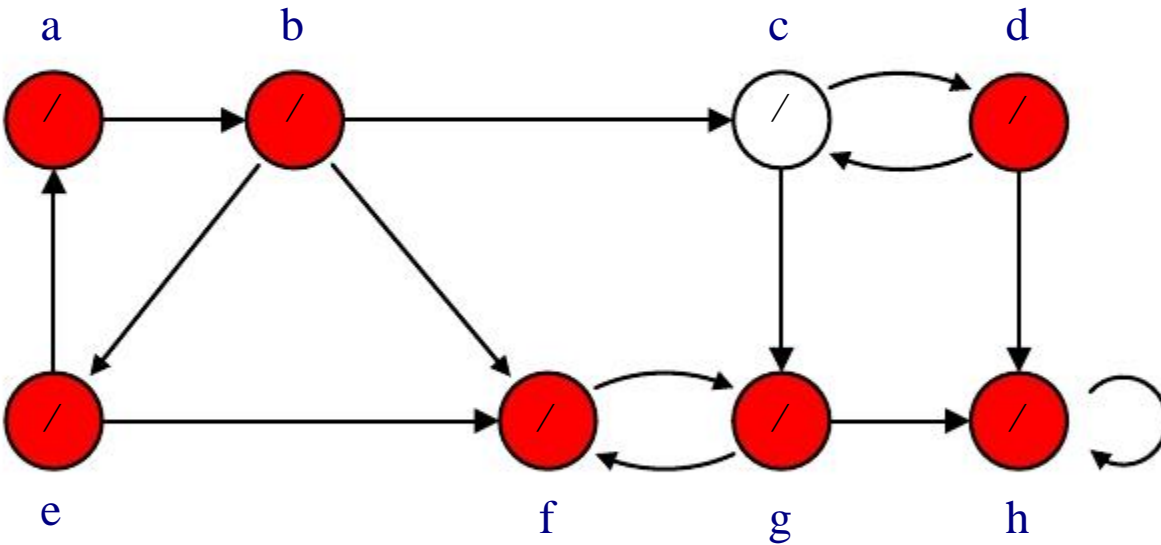
5    se  $cor[u] = \text{BRANCO}$

6        DFS-VISIT(u)

- Dado um Grafo, temos uma lista de todos os vértices...

# Busca em Profundidade


- Linha 2: Colorindo vértice **c** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]



DFS(G)

  $l$  para cada vértice  $u \leftarrow V[G]$

➡ 2 cor[u] ← BRANCO

$$3 \text{ tempo} \leftarrow 0$$

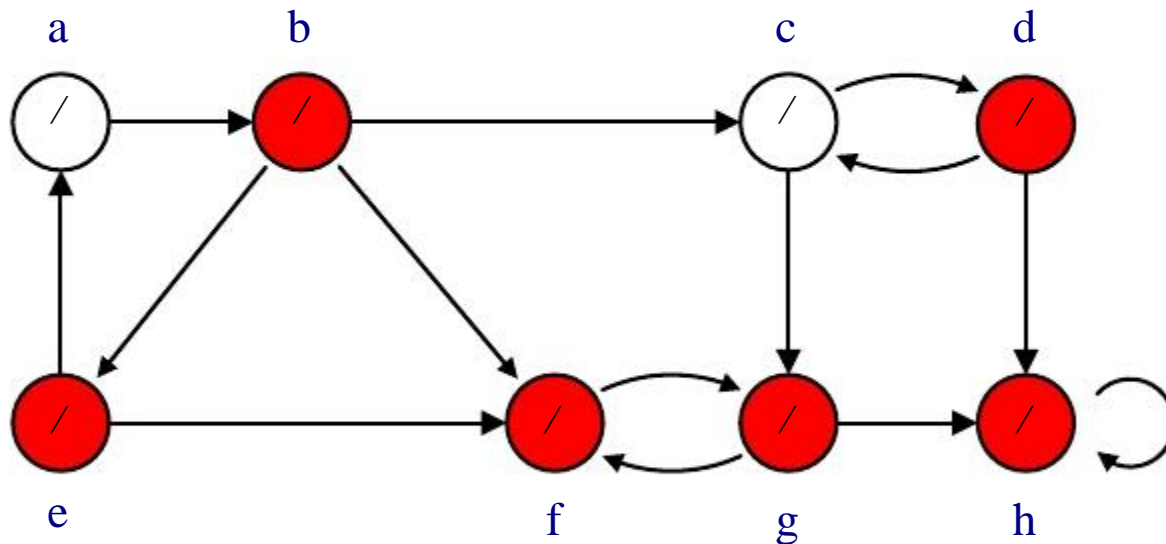
4 para cada vértice  $u \in V[G]$

5 se cor[u]=BRANCO

6 DFS – VISIT(*u*)

# Busca em Profundidade

- Linha 2: Colorindo vértice **a** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]

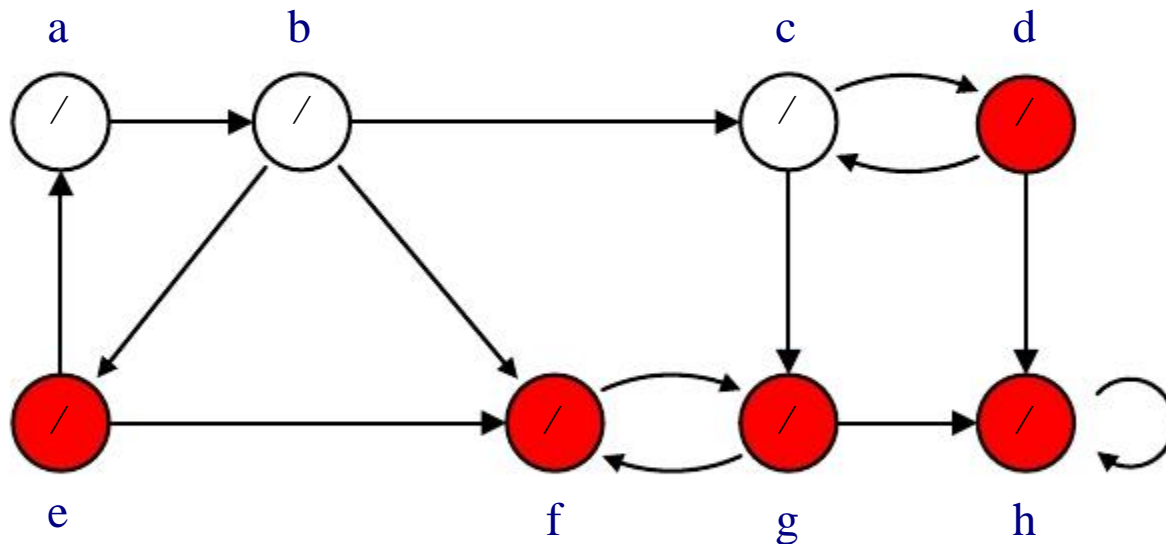


DFS(G)

1 para cada vértice  $u \leftarrow V[G]$   
2  $cor[u] \leftarrow \text{BRANCO}$   
3  $tempo \leftarrow 0$   
4 para cada vértice  $u \in V[G]$   
5 se  $cor[u] = \text{BRANCO}$   
6     DFS-VISIT(u)

# Busca em Profundidade

- Linha 2: Colorindo vértice **b** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]



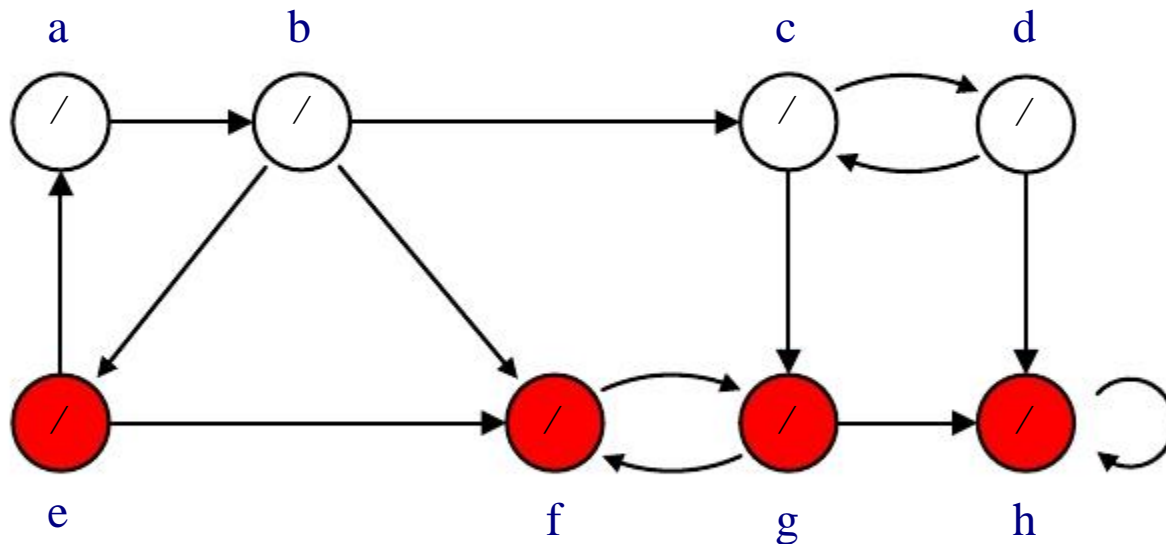
DFS(G)

1 para cada vértice  $u \leftarrow V[G]$   
2 cor[u] ← BRANCO  
3 tempo ← 0  
4 para cada vértice  $u \in V[G]$   
5 se cor[u] = BRANCO  
6 DFS-VISIT(u)



# Busca em Profundidade

- Linha 2: Colorindo vértice **d** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]

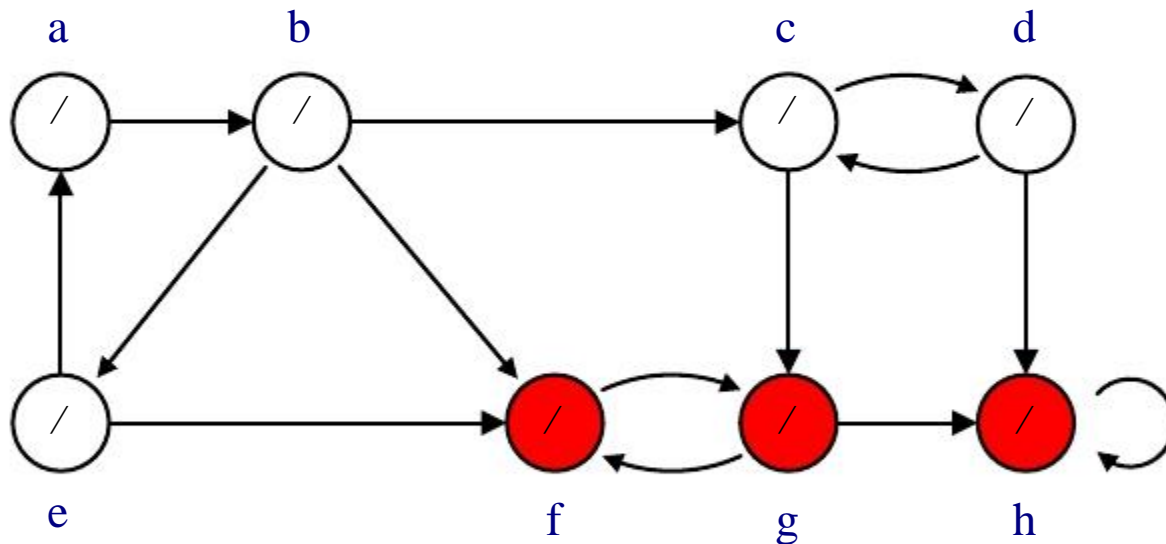


*DFS(G)*

- ➡ 1 para cada vértice  $u \leftarrow V[G]$
- ➡ 2  $cor[u] \leftarrow \text{BRANCO}$
- 3  $tempo \leftarrow 0$
- 4 para cada vértice  $u \in V[G]$
- 5 se  $cor[u] = \text{BRANCO}$
- 6  $\text{DFS-VISIT}(u)$

# Busca em Profundidade

- Linha 2: Colorindo vértice **e** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]

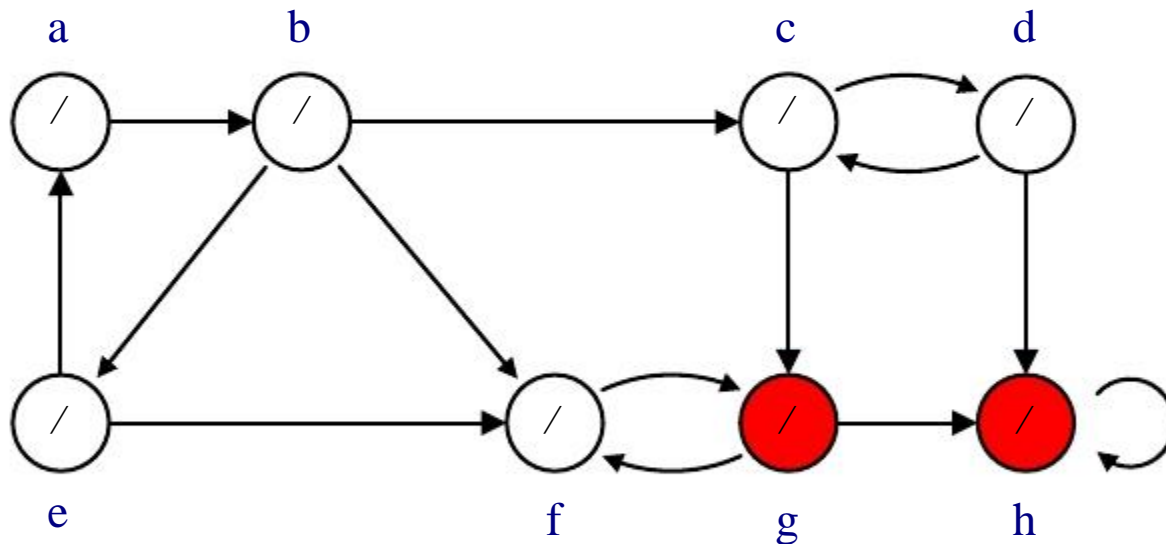


DFS(G)

1 para cada vértice  $u \leftarrow V[G]$   
2 cor[u]  $\leftarrow$  BRANCO  
3 tempo  $\leftarrow 0$   
4 para cada vértice  $u \in V[G]$   
5 se cor[u] = BRANCO  
6 DFS-VISIT(u)

# Busca em Profundidade

- Linha 2: Colorindo vértice **f** de BRANCO;



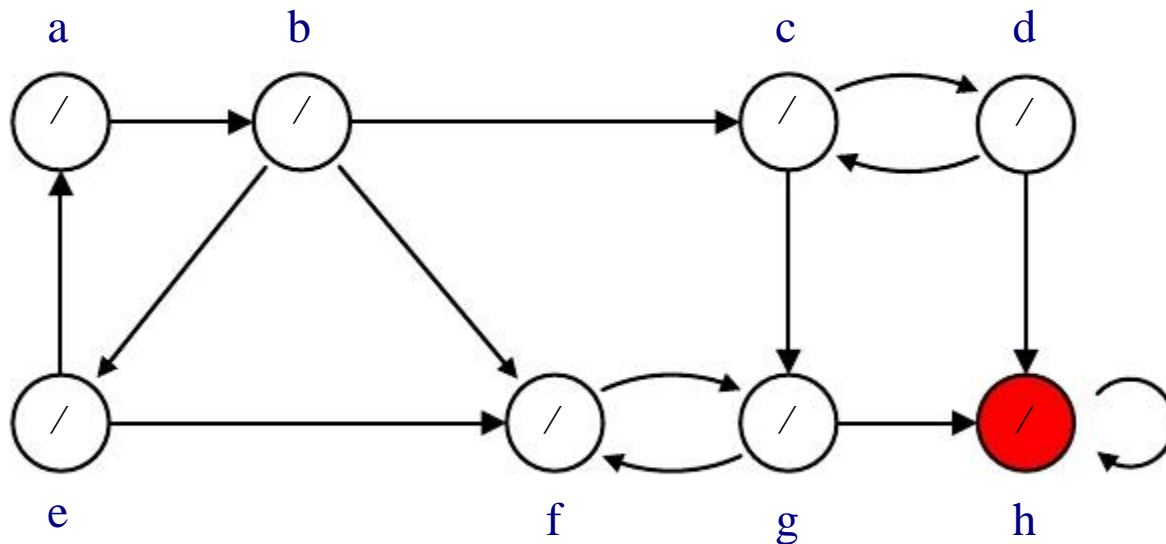
Lista: [c,a,b,d,e,f,g,h]

DFS(G)

1 para cada vértice  $u \leftarrow V[G]$   
2 cor[u] ← BRANCO  
3 tempo ← 0  
4 para cada vértice  $u \in V[G]$   
5 se cor[u] = BRANCO  
6 DFS-VISIT(u)

# Busca em Profundidade

- Linha 2: Colorindo vértice **g** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]

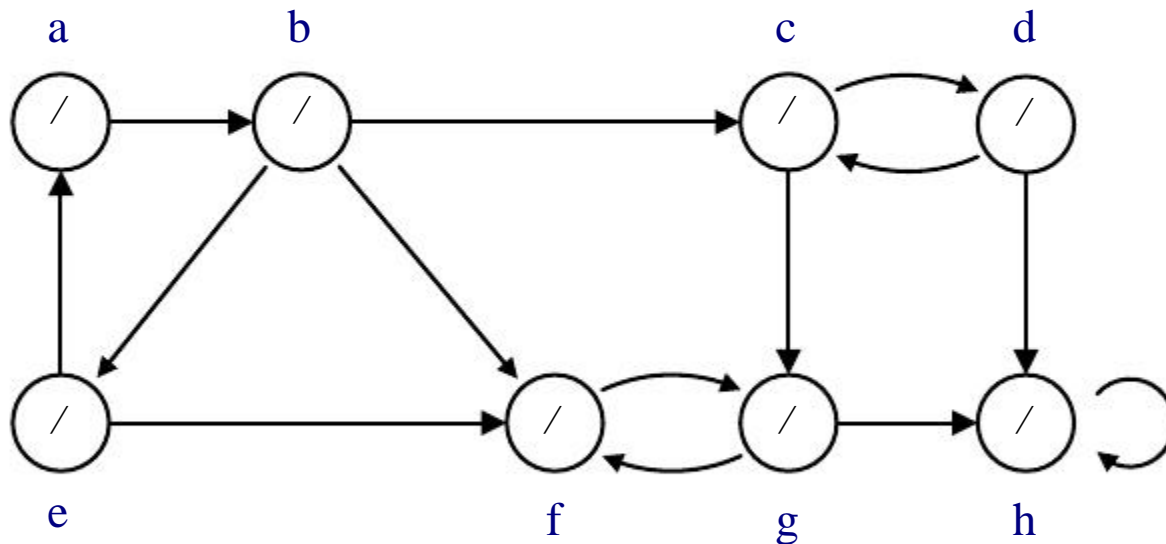


*DFS(G)*

- ➡ 1 para cada vértice  $u \leftarrow V[G]$
- ➡ 2  $cor[u] \leftarrow BRANCO$
- 3  $tempo \leftarrow 0$
- 4 para cada vértice  $u \in V[G]$
- 5 se  $cor[u] = BRANCO$
- 6  $DFS-VISIT(u)$

# Busca em Profundidade

- Linha 2: Colorindo vértice **h** de BRANCO;



Lista: [c,a,b,d,e,f,g,h]

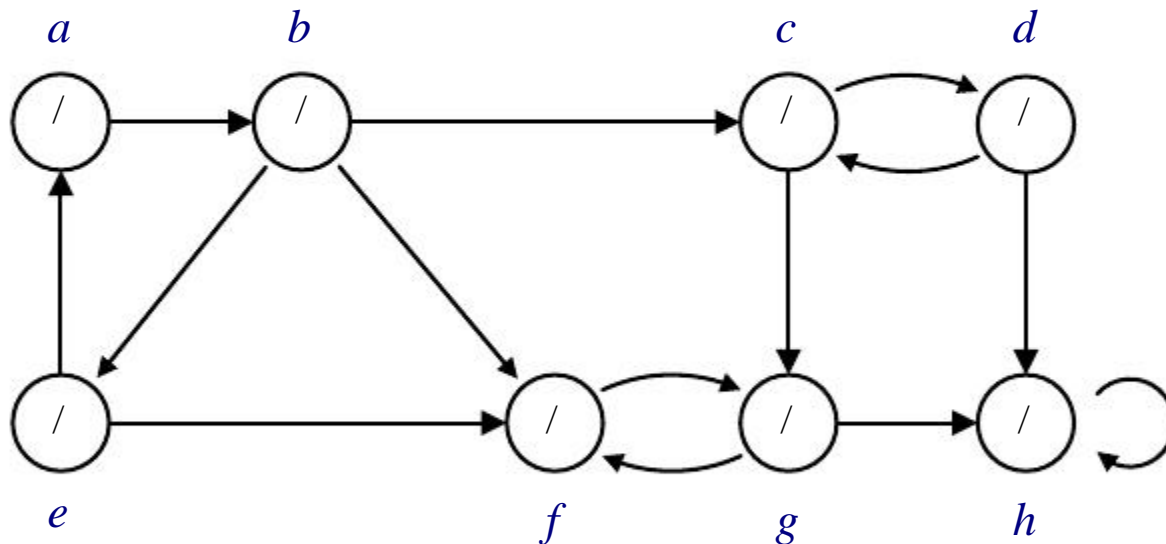


DFS(G)

- 1 para cada vértice  $u \leftarrow V[G]$
- 2 cor[u] ← BRANCO
- 3 tempo ← 0
- 4 para cada vértice  $u \in V[G]$
- 5 se cor[u] = BRANCO
- 6 DFS-VISIT(u)

# Busca em Profundidade

- Inicializando variável *tempo*;



Lista: [c,a,b,d,e,f,g,h]

*DFS(G)*

1 para cada vértice  $u \leftarrow V[G]$

2  $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

4 para cada vértice  $u \in V[G]$

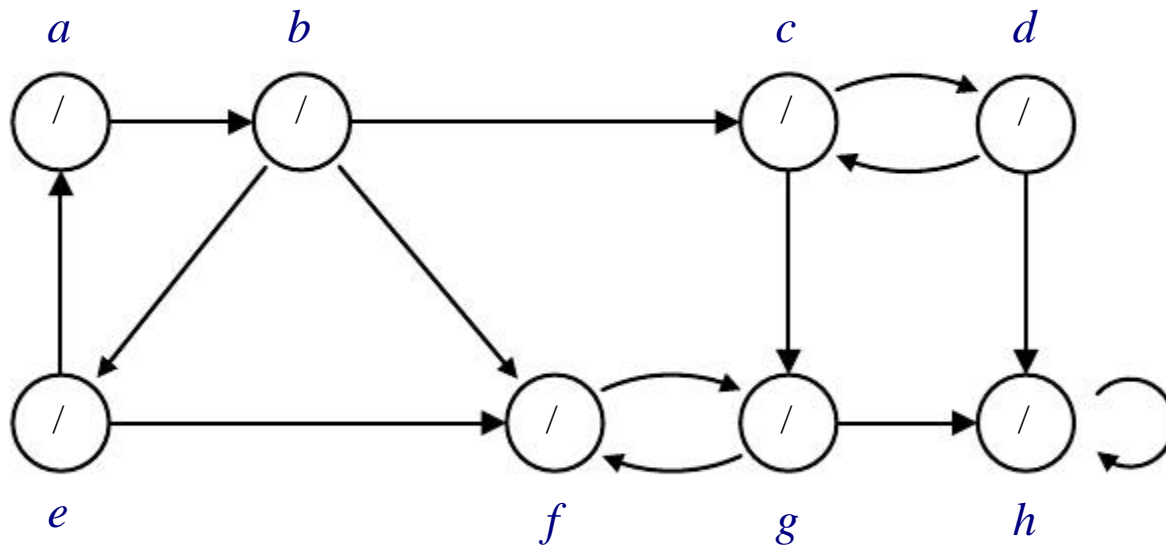
5 se  $cor[u] = BRANCO$

6  $DFS-VISIT(u)$

$tempo = 0$

# Busca em Profundidade

- Para todos os vértices do grafo...



Lista: [c,a,b,d,e,f,g,h]



tempo = 0

*DFS(G)*

1 para cada vértice  $u \leftarrow V[G]$

2  $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

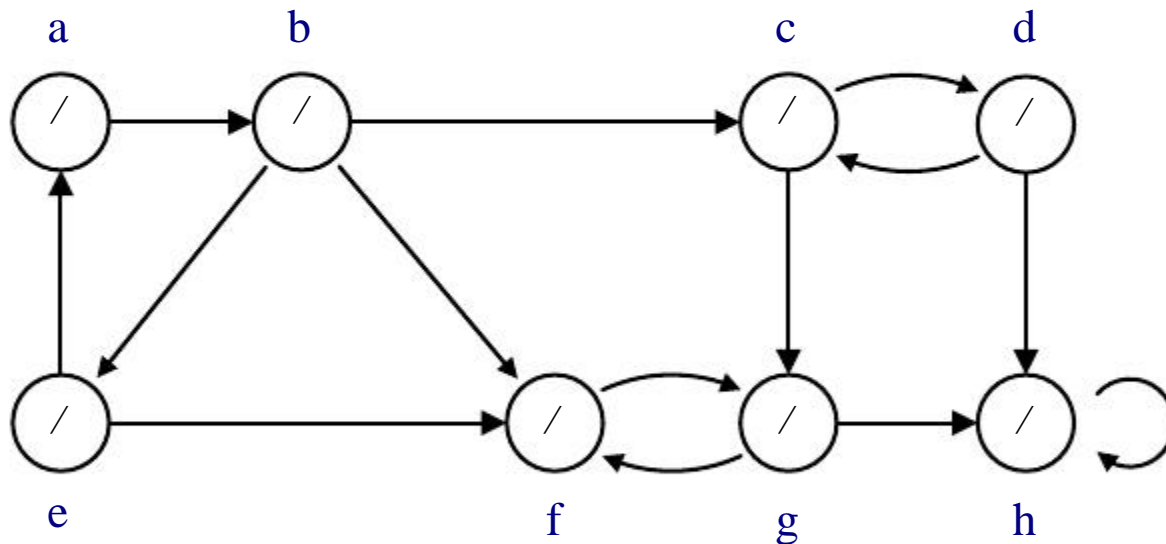
 4 para cada vértice  $u \in V[G]$

5 se  $cor[u] = BRANCO$

6  $DFS-VISIT(u)$

# Busca em Profundidade

- A cor do vértice **c** é BRANCA?



Lista: [c,a,b,d,e,f,g,h]



tempo = 0

*DFS(G)*

1 para cada vértice  $u \leftarrow V[G]$

2  $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

4 para cada vértice  $u \in V[G]$

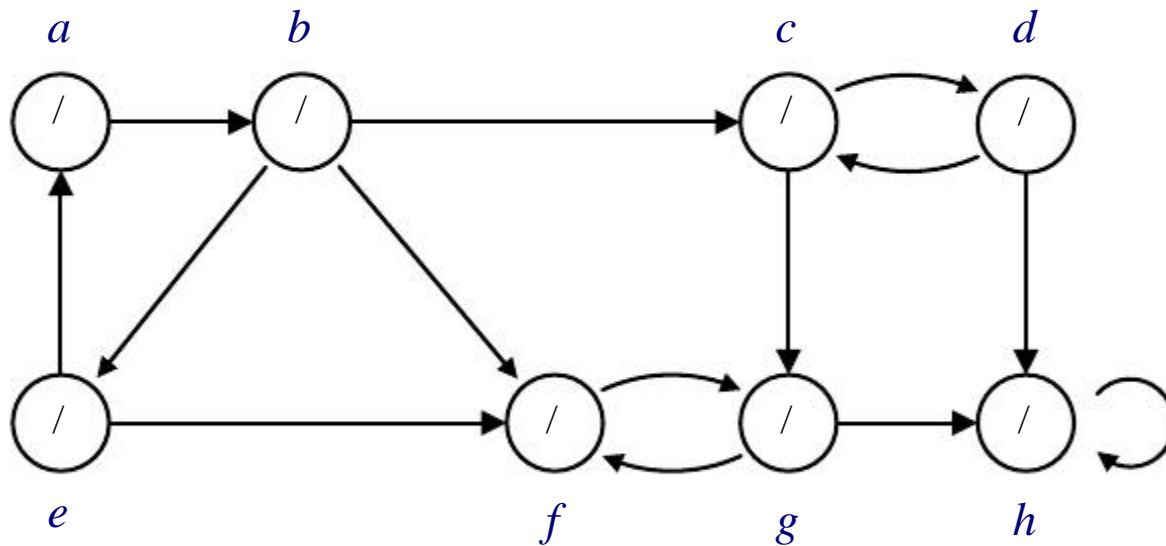
5 se  $cor[u] = BRANCO$

6  $DFS-VISIT(u)$



# Busca em Profundidade

- Chamada de função: **DFS\_VISIT(c)**;
- Vai empilhar a função DFS(G), com o  $CP = 4$ , e próximo  $u=a$ ;



Lista: [c,a,b,d,e,f,g,h]



tempo = 0

*DFS(G)*

1 para cada vértice  $u \leftarrow V[G]$

2  $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

4 para cada vértice  $u \in V[G]$

5 se  $cor[u] = BRANCO$

6  $DFS\_VISIT(u)$



# Busca em Profundidade

- Chamada de função: **DFS\_VISIT(c)**;



*DFS-VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

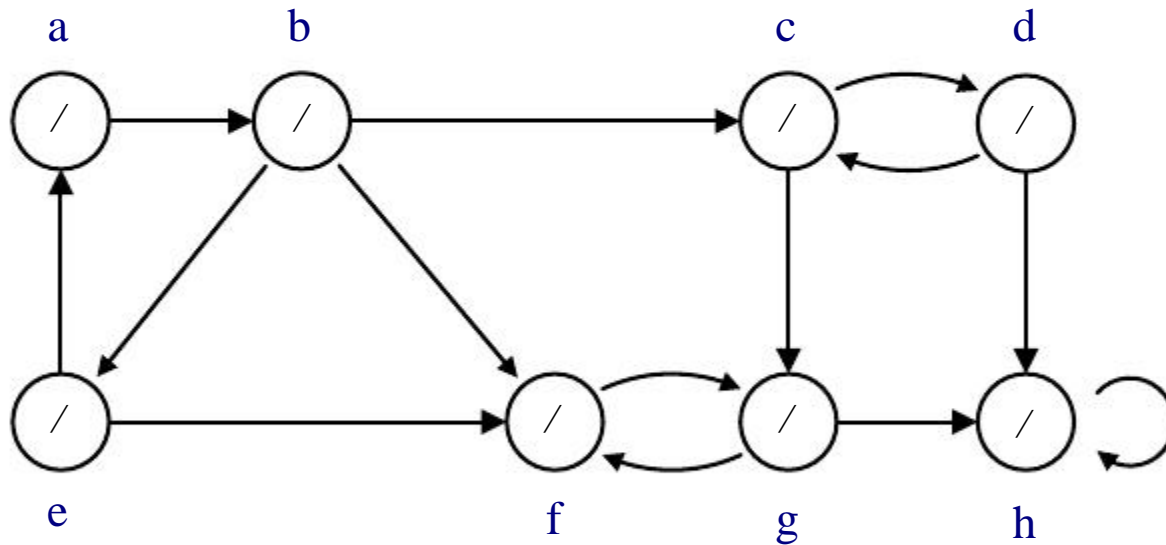
4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6  $DFS-VISIT(v)$

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$



Lista: [c,a,b,d,e,f,g,h]

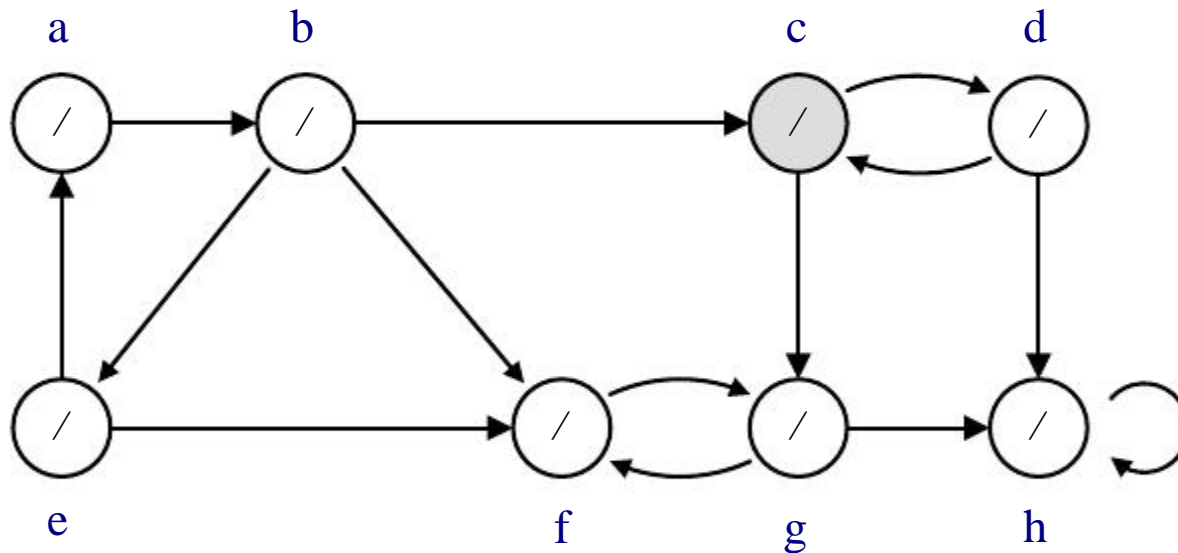
**Pilha de execução:**

DFS(G) - CP: linha 4 – próximo:  $u = a$

$tempo = 0$

# Busca em Profundidade

- Colore **c** de cinza;



Lista: [c,a,b,d,e,f,g,h]

tempo = 0

DFS -VISIT(u)



1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

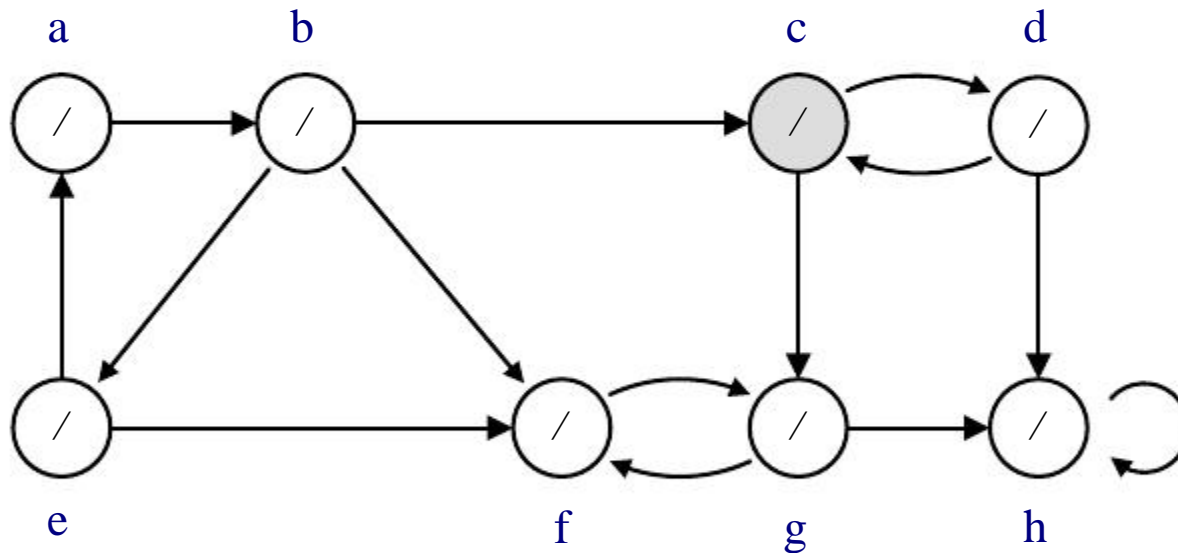
8 f[u] ← tempo ← (tempo + 1)

Pilha de execução:

DFS(G) - CP: linha 4 – próximo: u = a

# Busca em Profundidade

- Colore **c** de cinza;



Lista: [c,a,b,d,e,f,g,h]

DFS -VISIT(u)

1 cor[u] ← CINZA

➔ 2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

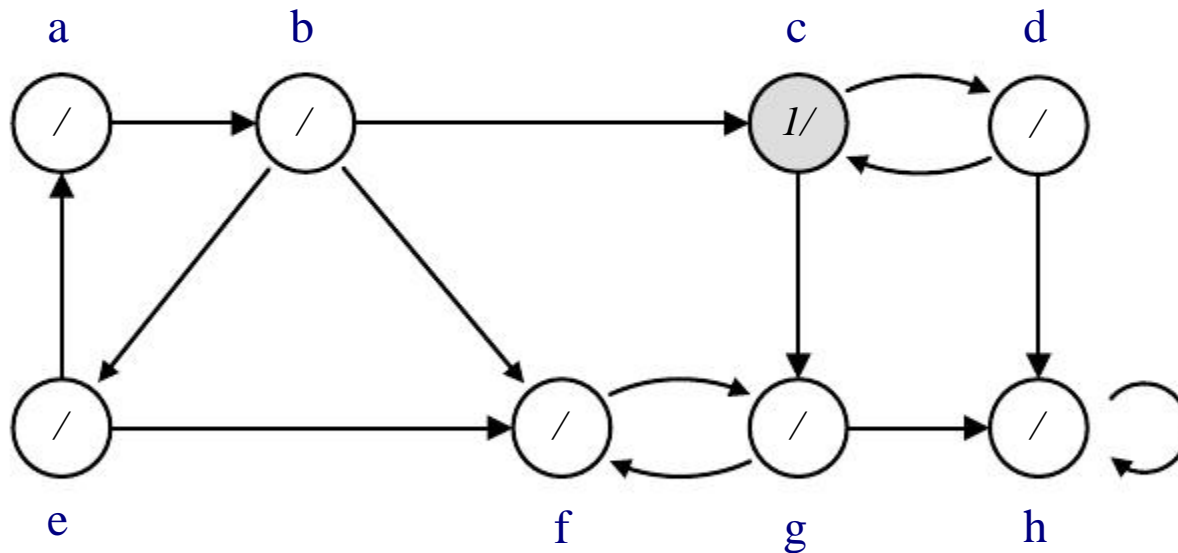
Pilha de execução:

DFS(G) - CP: linha 4 – próximo: u = a

tempo = 0 => 1

# Busca em Profundidade

- Colore **c** de cinza;



Lista: [c,a,b,d,e,f,g,h]

tempo = 1

*DFS-VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

➔ 3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6  $DFS-VISIT(v)$

7  $cor[u] \leftarrow PRETO$

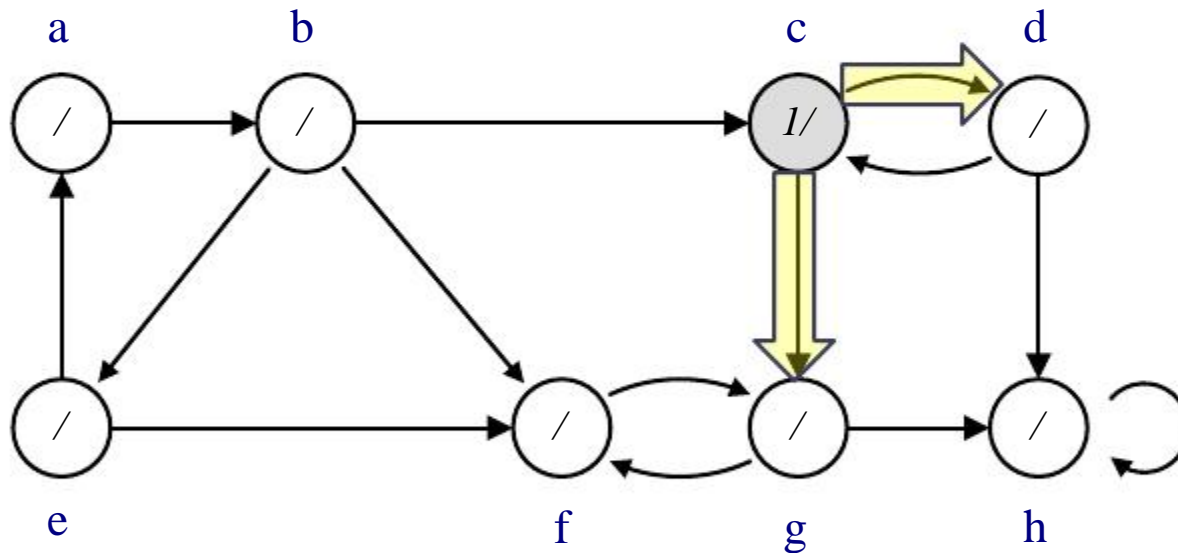
8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

DFS(G) - CP: linha 4 – próximo:  $u = a$

# Busca em Profundidade

- Como não sabemos qual a representação computacional utilizada, vamos considerar primeiro g, depois d.



Lista: [c,a,b,d,e,f,g,h]

tempo = 1

*DFS - VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6 *DFS - VISIT(v)*

7  $cor[u] \leftarrow PRETO$

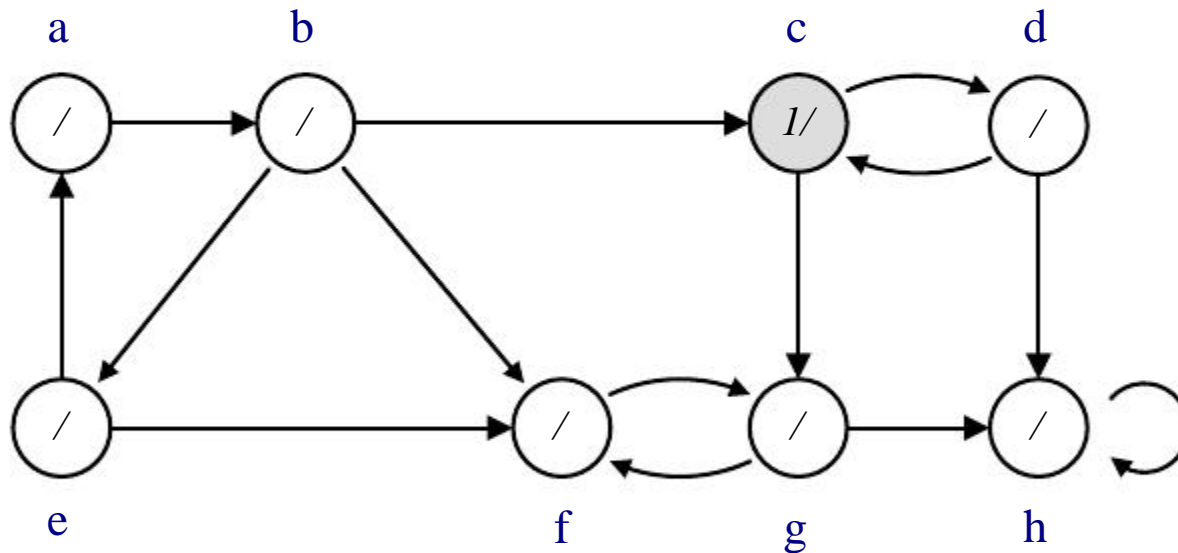
8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

DFS(G) - CP: linha 4 – próximo: u = a

# Busca em Profundidade

- A cor de **g** é BRANCA?



Lista: [c,a,b,d,e,f,g,h]

tempo = 1

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

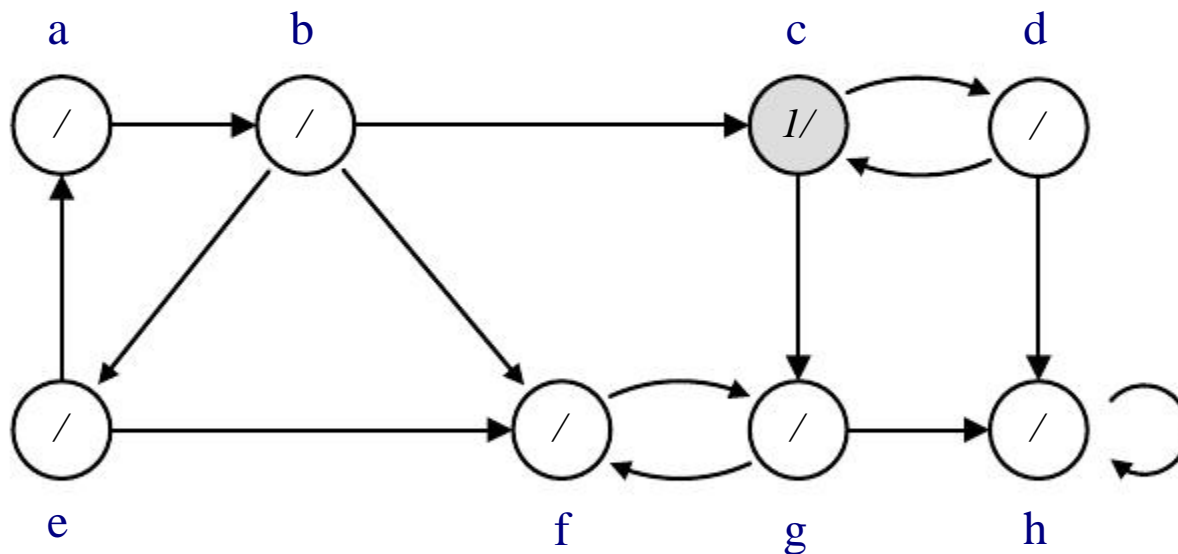
8 f[u] ← tempo ← (tempo + 1)

Pilha de execução:

DFS(G) - CP: linha 4 – próximo: u = a

# Busca em Profundidade

- Chama a função DFS\_VISIT(g)
- Vai empilhar DFS\_VISIT(c), CP = 4



Lista: [c,a,b,d,e,f,g,h]

tempo = 1

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

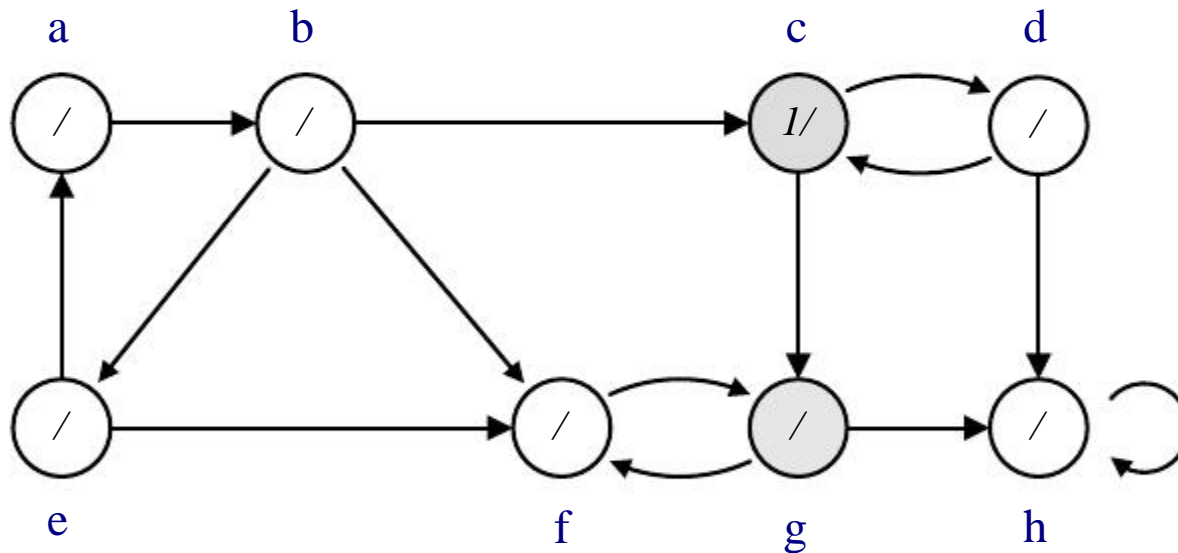
**Pilha de execução:**

DFS(G) - CP: linha 4 – próximo: u = a



# Busca em Profundidade

- Colore g de cinza



Lista: [c,a,b,d,e,f,g,h]

tempo = 1

DFS -VISIT(u)



1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

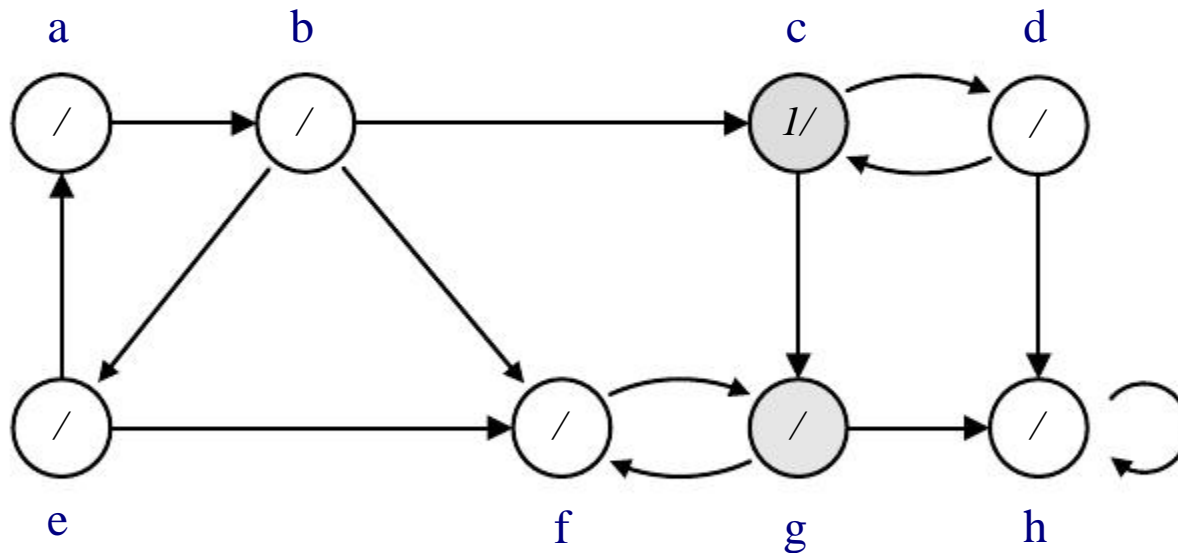
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Incremento no tempo: 2



Lista: [c,a,b,d,e,f,g,h]

DFS -VISIT(u)

1 cor[u] ← CINZA

➔ 2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

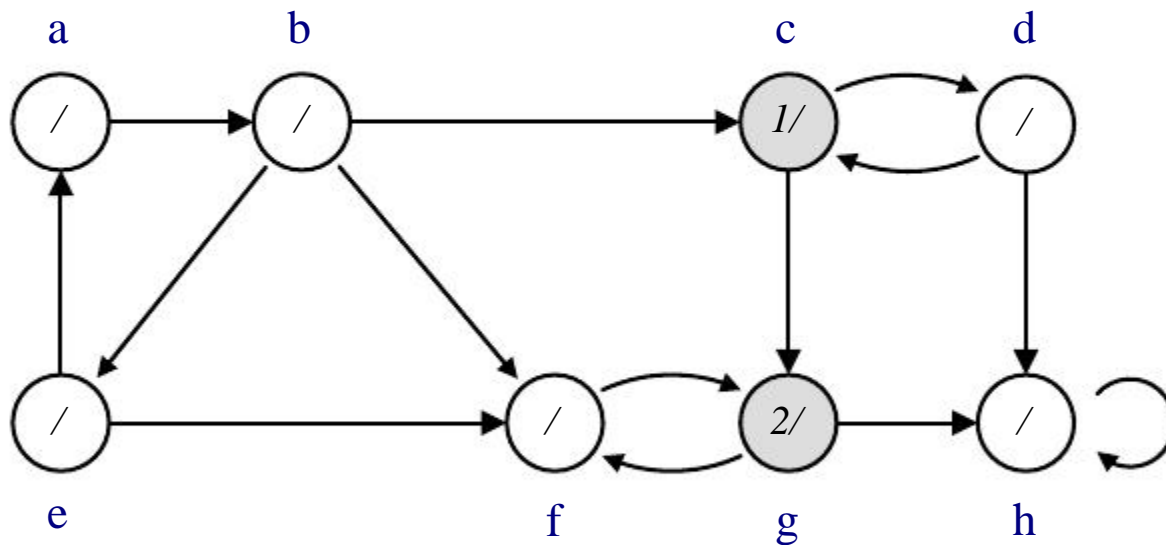
DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

tempo = 1 => 2

# Busca em Profundidade

- Indica o tempo de descoberta do vértice  $g$



Lista: [c,a,b,d,e,f,g,h]

tempo = 2

*DFS-VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

➔ 3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6 *DFS-VISIT(v)*

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

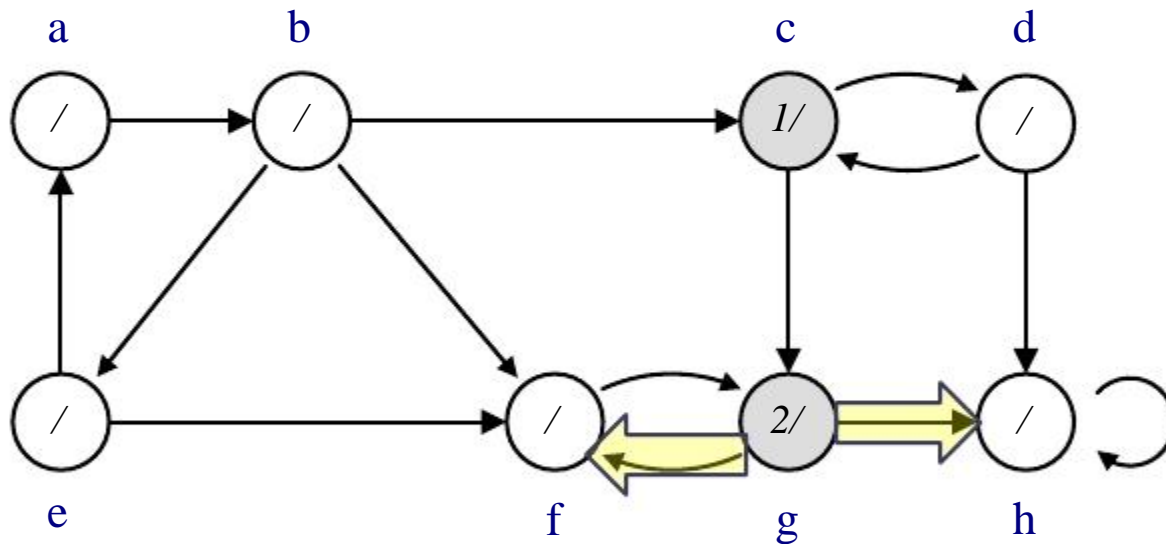
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo:  $u = a$

# Busca em Profundidade

- Para cada adjacente do vértice  $g = \{f, h\}$



Lista: [c,a,b,d,e,f,g,h]

tempo = 2

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

➔ 4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

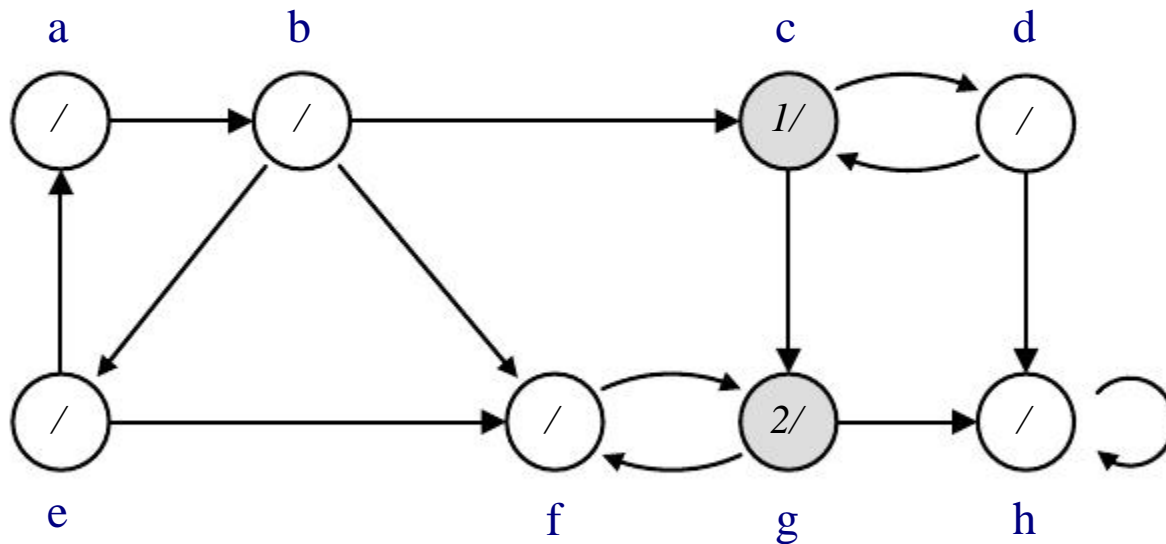
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(g) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- A cor do vértice f é BRANCA?
  - Então, invoca DFS\_VISIT(f);
  - Empilha DFS\_VISIT(g), CP: 4 – próximo: v=h



Lista: [c,a,b,d,e,f,g,h]

tempo = 2

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

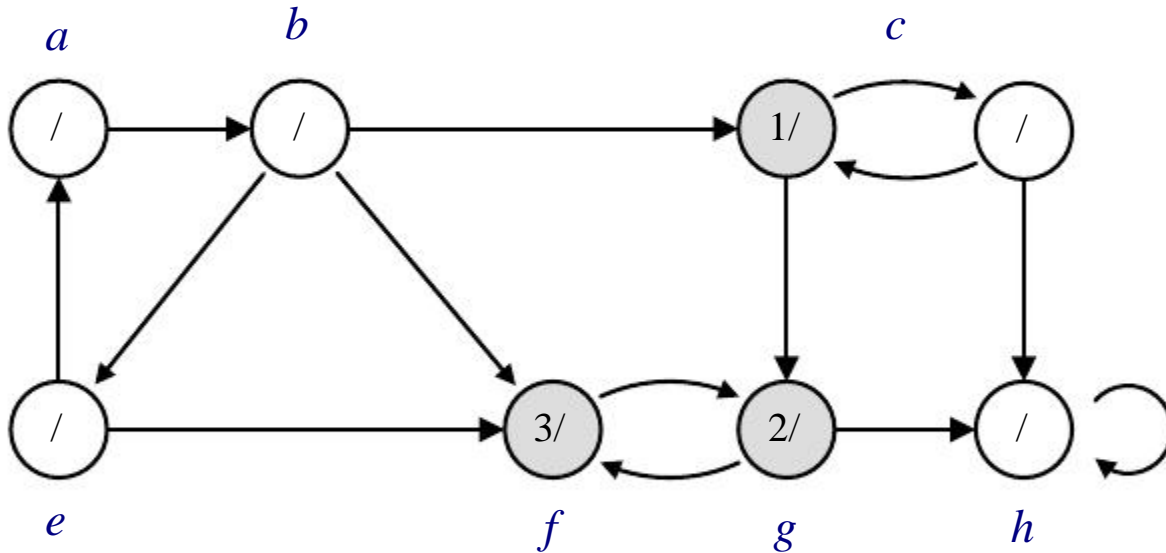
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 – próximo: u = a

# Busca em Profundidade

- Marca o vértice  $f$  como CINZA.  
Incrementa o Tempo.
- Indica o tempo de descoberta do vértice  $f$



*Lista: [c,a,b,d,e,f,g,h]*

tempo = 2 => 3

$$DFS-VISIT(u)$$
$$1 \text{ } cor[u] \leftarrow CINZA$$
$$2 \text{ tempo} \leftarrow \text{tempo} + 1$$

```
3 d[u] ← tempo
```

4 para cada vértice  $v \in Adj(u)$

5    *se cor[v]=BRANCO*

6      *DFS-VISIT*(*v*)
$$7 \text{ cor}[u] \leftarrow PRETO$$
$$8 \quad f[u] \leftarrow tempo \leftarrow (tempo+1)$$

## Pilha de execução:

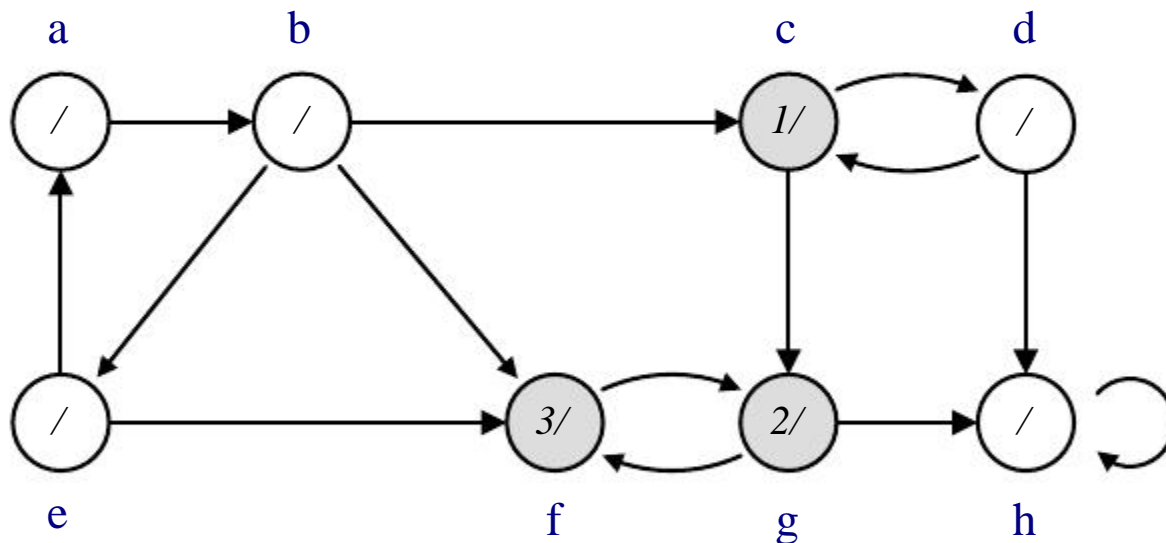
DFS\_VISIT(g), CP: linha 4

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo:  $u = a$

# Busca em Profundidade

- O vértice f possui apenas um adjacente: {g}
- g não é BRANCO;



Lista: [c,a,b,d,e,f,g,h]

tempo = 3

*DFS-VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6 *DFS-VISIT(v)*

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

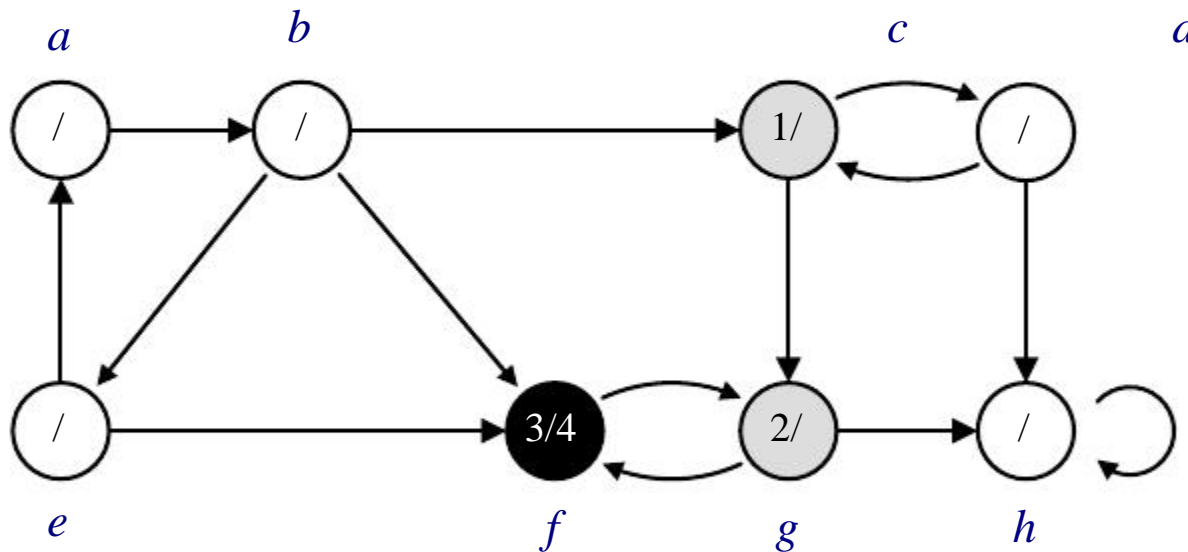
DFS\_VISIT(g), CP: linha 4

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- O laço termina, f recebe a cor preta; Incrementa o tempo; É indicado o tempo de finalização de f;
- A função termina... Agora, e a última chamada é desempilhada.



Lista: [c,a,b,d,e,f,g,h]

tempo = 3 => 4

*DFS – VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6  $DFS\_VISIT(v)$

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

DFS\_VISIT(g), CP: linha 4

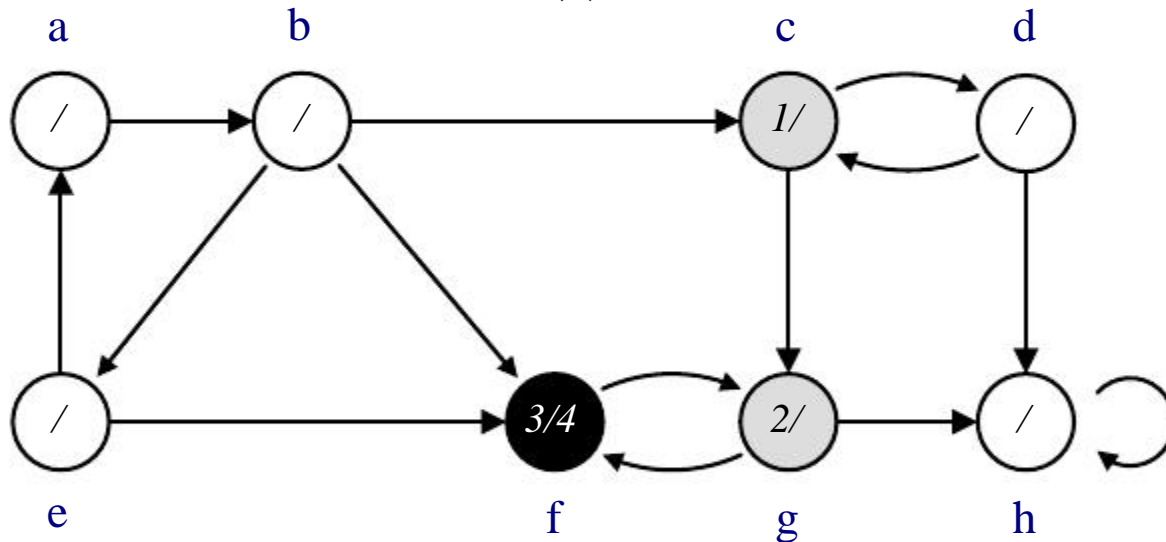
DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 – próximo: u = a



# Busca em Profundidade

- Desempilhou: **DFS\_VISIT(g)**, CP: linha 4
- O próximo adjacente de g é h, e ele é BRANCO...
- Assim, empilha novamente DFS\_VISIT(g), CP: linha 4
- E chama DFS\_VISIT(h)



Lista: [c,a,b,d,e,f,g,h]

tempo = 4

*DFS - VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6  $DFS\_VISIT(v)$

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Colore o vértice  $h$  de cinza, incrementa uma unidade de tempo, e indica o tempo que o vértice  $h$  foi localizado...

DFS –VISIT( $u$ )

1  $cor[u] \leftarrow \text{CINZA}$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

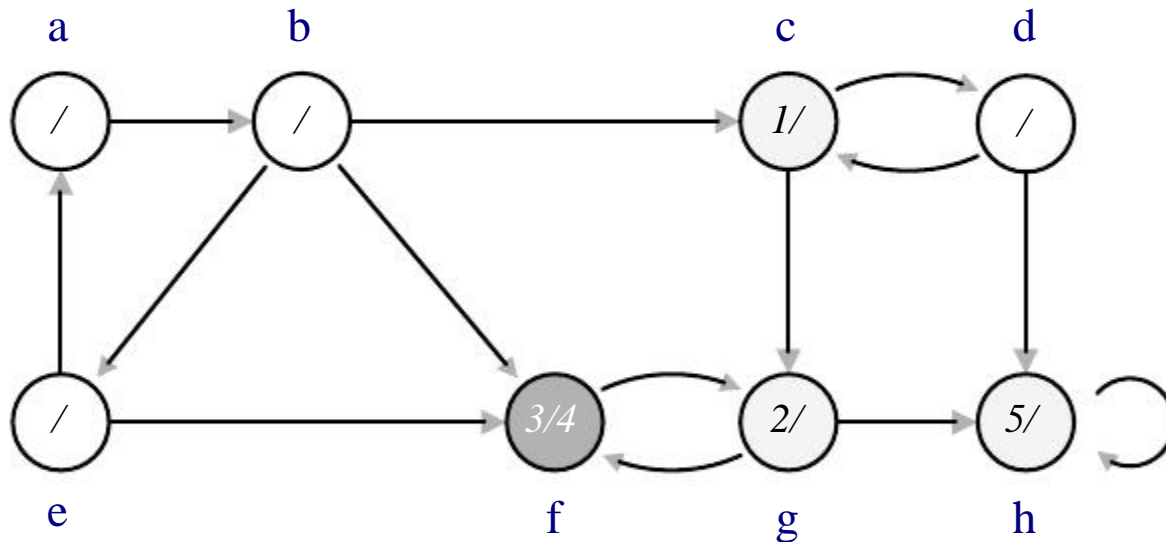
4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = \text{BRANCO}$

6 DFS –VISIT( $v$ )

7  $cor[u] \leftarrow \text{PRETO}$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$



Lista: [c,a,b,d,e,f,g,h]

**Pilha de execução:**

DFS\_VISIT( $g$ ), CP: linha 4

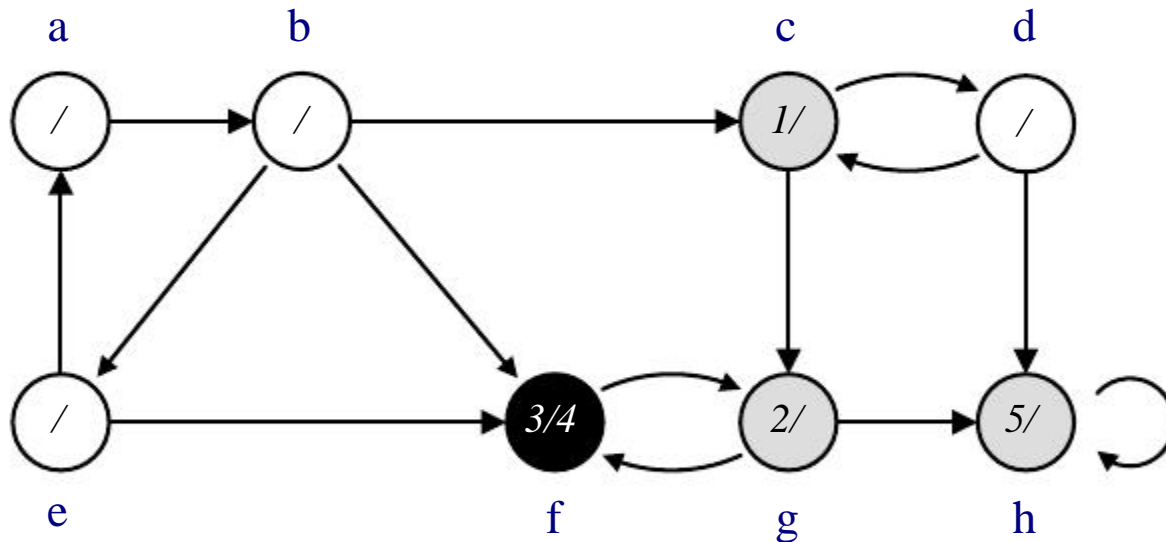
DFS\_VISIT( $c$ ), CP: linha 4

DFS( $G$ ) - CP: linha 4 – próximo:  $u = a$

$tempo = 4 \Rightarrow 5$

# Busca em Profundidade

- Para cada adjacente de h: {h}
- Mas o vértice h é CINZA...
- Então a busca sobre h será finalizada...



Lista: [c,a,b,d,e,f,g,h]

tempo = 5

DFS - VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS - VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

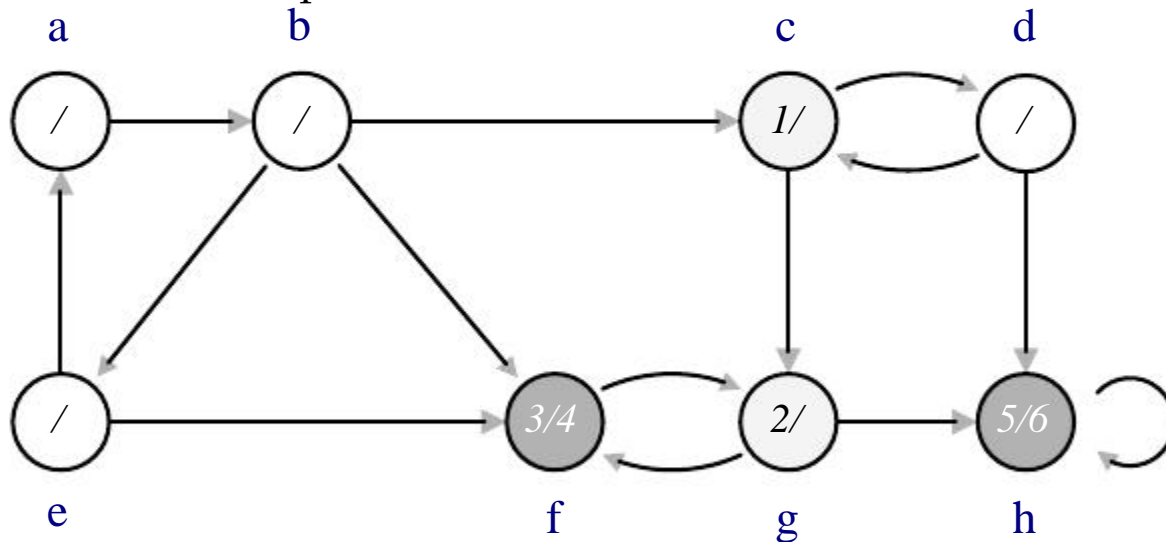
DFS\_VISIT(g), CP: linha 4

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Marca h de PRETO;
- Incrementa o tempo em uma unidade;
- Indica o tempo de finalização de h;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

tempo = 5 => 6

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

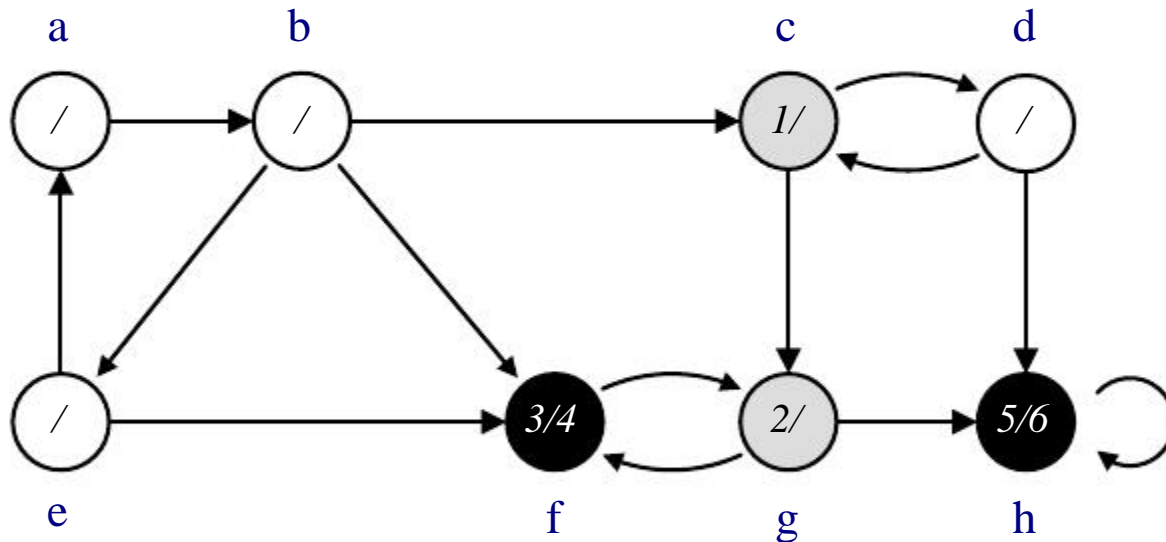
DFS\_VISIT(g), CP: linha 4

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Desempilhou : DFS\_VISIT(g), CP: linha 4
- O vértice g não possui mais adjacentes não visitados.
- Assim, a busca em g termina...



Lista: [c,a,b,d,e,f,g,h]

tempo = 6

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

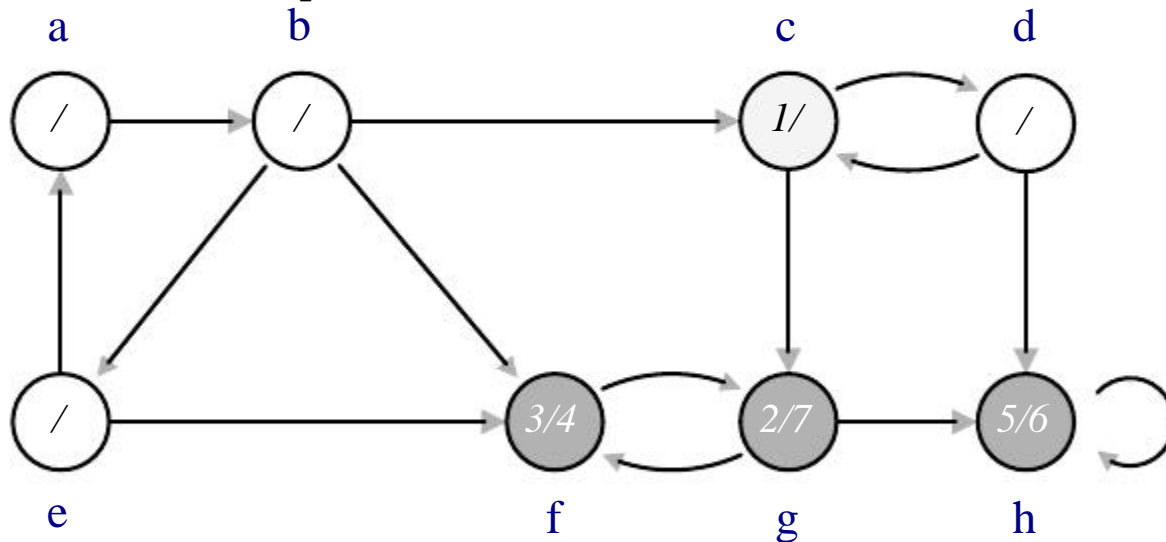
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Marca g de PRETO;
- Incrementa o tempo em uma unidade;
- Indica o tempo de finalização de g;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

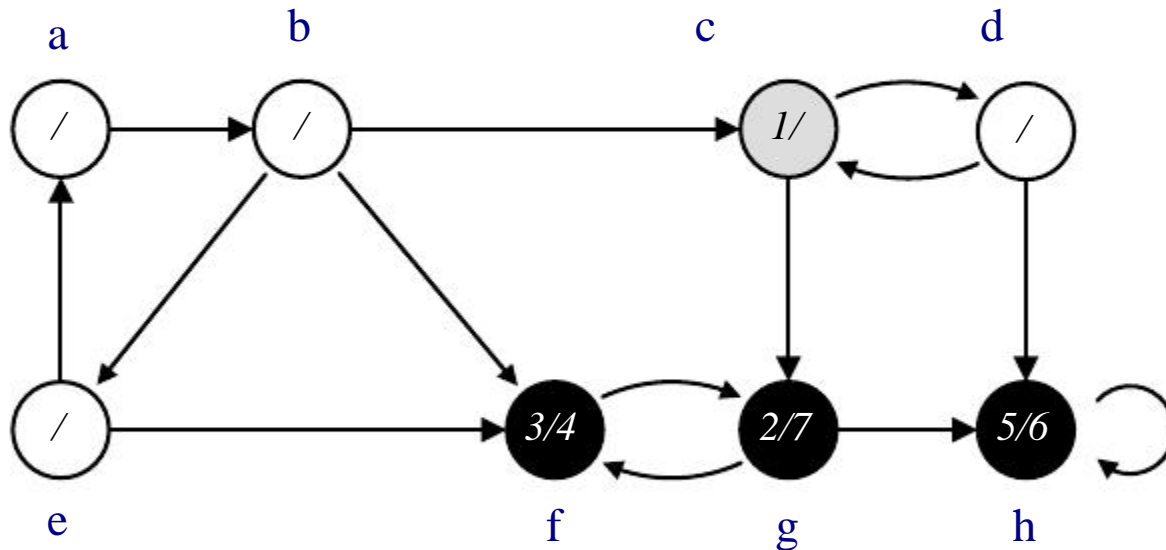
DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 – próximo: u = a

tempo = 6 => 7

# Busca em Profundidade

- Desempilhou DFS\_VISIT(c), CP: linha 4
- O próximo adjacente do vértice c é o vértice d, que é BRANCO, assim, invoca DFS\_VISIT(d) e empilha [DFS\_VISIT(c), CP: linha 4] novamente



Lista: [c,a,b,d,e,f,g,h]

Pilha de execução:

DFS(G) - CP: linha 4 - próximo: u = a

tempo = 7

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice v ∈ Adj(u)

5 se cor[v] = BRANCO

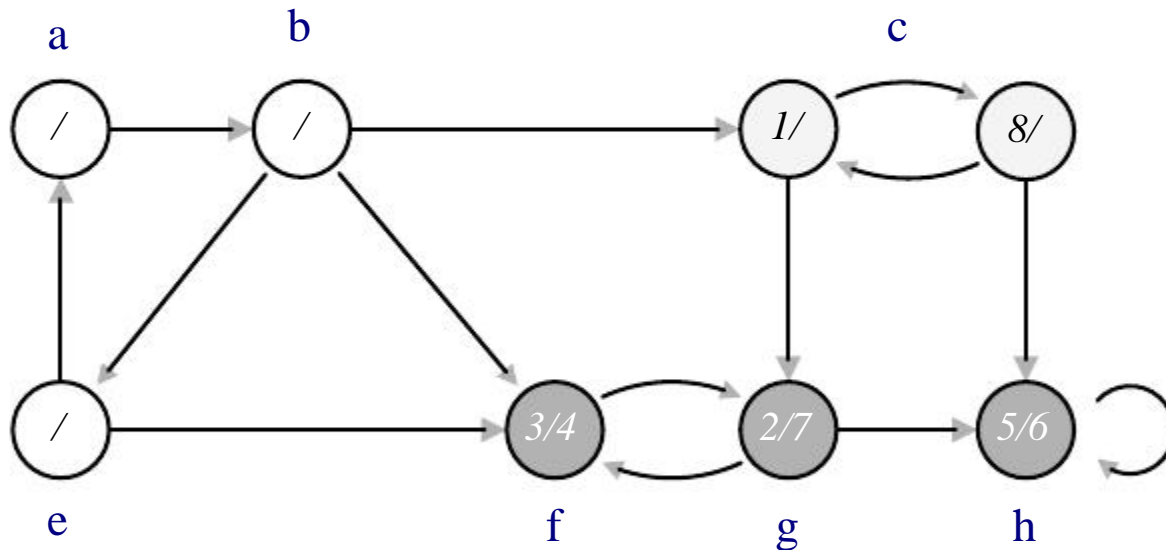
6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

# Busca em Profundidade

- Desempilhou DFS\_VISIT(c), CP: linha 4
- O próximo adjacente do vértice c é o vértice d, que é BRANCO, assim, invoca DFS\_VISIT(d) e empilha [DFS\_VISIT(c), CP: linha 4] novamente



Lista: [c,a,b,d,e,f,g,h]

*DFS - VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6  $DFS - VISIT(v)$

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

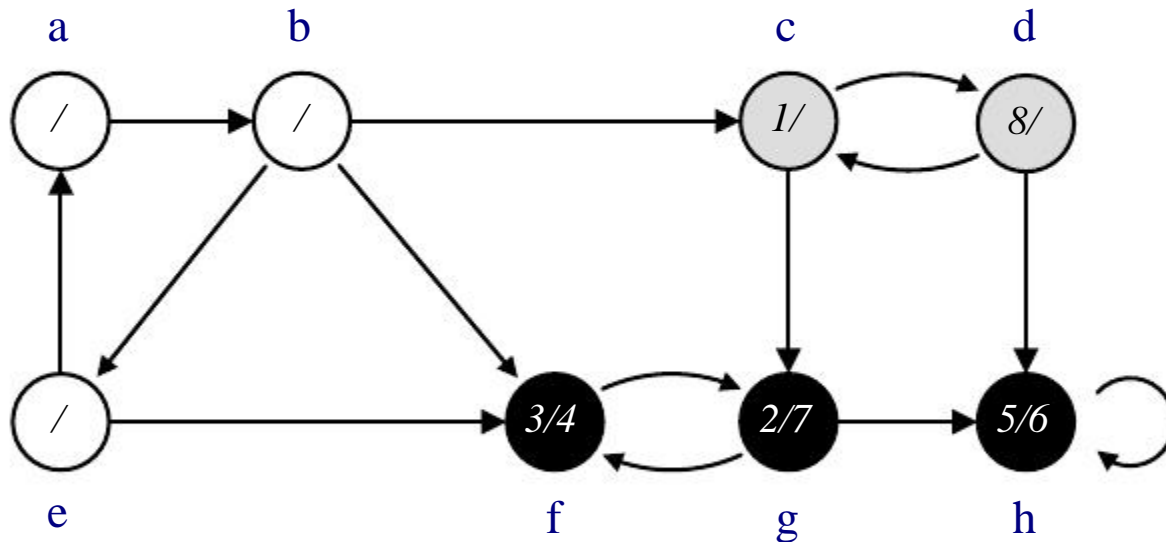
DFS(G) - CP: linha 4 - próximo: u = a

tempo = 7 => 8



# Busca em Profundidade

- O vértice d possui apenas um adjacente, que não é BRANCO, assim a busca sobre d termina...



Lista: [c,a,b,d,e,f,g,h]

tempo = 8

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

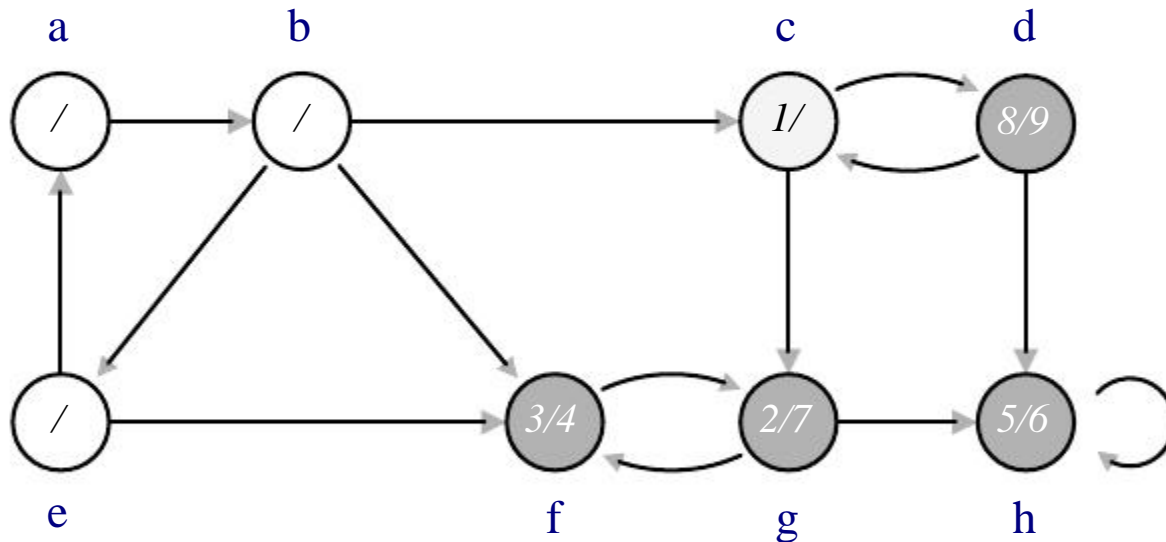
**Pilha de execução:**

DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = a

# Busca em Profundidade

- Marca d de PRETO; Incrementa o tempo;
- Atribui o tempo de finalização de d;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

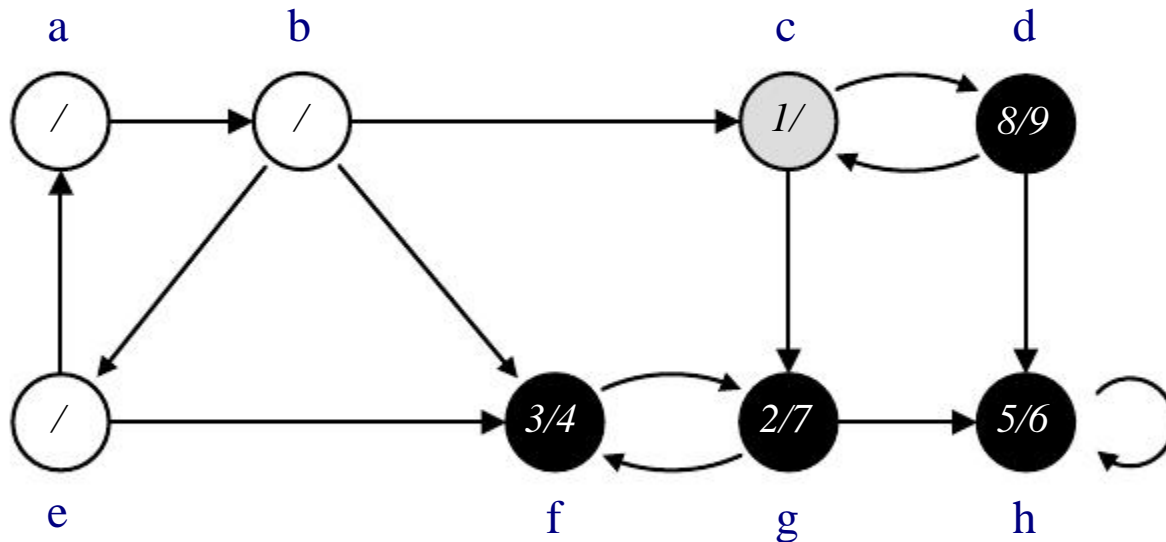
DFS\_VISIT(c), CP: linha 4

DFS(G) - CP: linha 4 – próximo: u = a

tempo = 8 => 9

# Busca em Profundidade

- Desempilhou DFS\_VISIT(c), CP: linha 4
- Não possui mais adjacentes, então a busca sobre c será finalizada...



Lista: [c,a,b,d,e,f,g,h]

**Pilha de execução:**

DFS(G) - CP: linha 4 – próximo: u = a

tempo = 9

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice v ∈ Adj(u)

5 se cor[v] = BRANCO

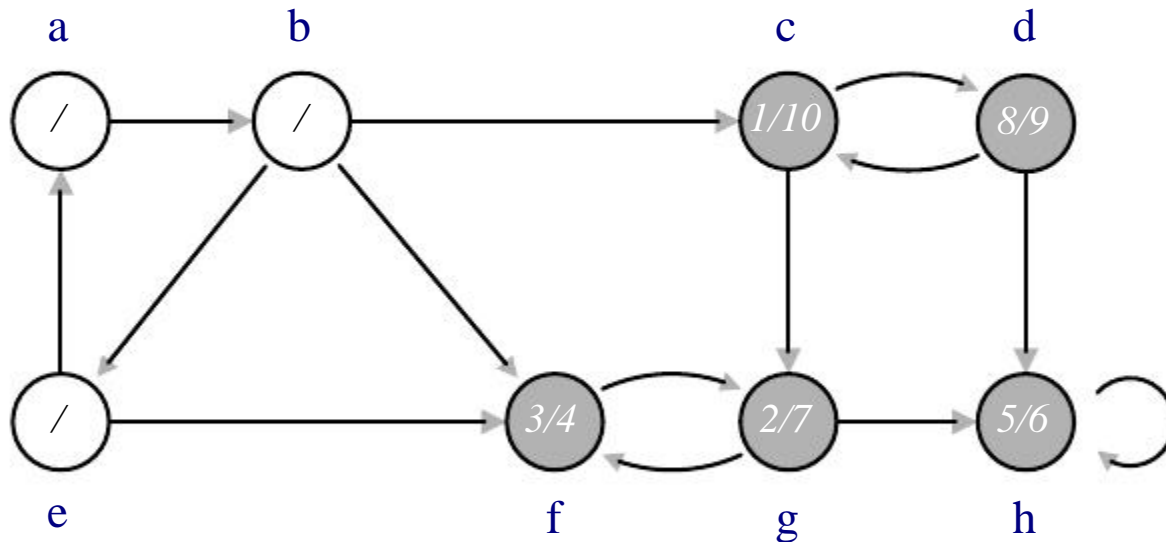
6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

# Busca em Profundidade

- Marca c de PRETO; Incrementa o tempo;
- Atribui o tempo de finalização de c;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

Pilha de execução:

DFS(G) - CP: linha 4 – próximo: u = a

tempo = 9 => 10

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice v ∈ Adj(u)

5 se cor[v] = BRANCO

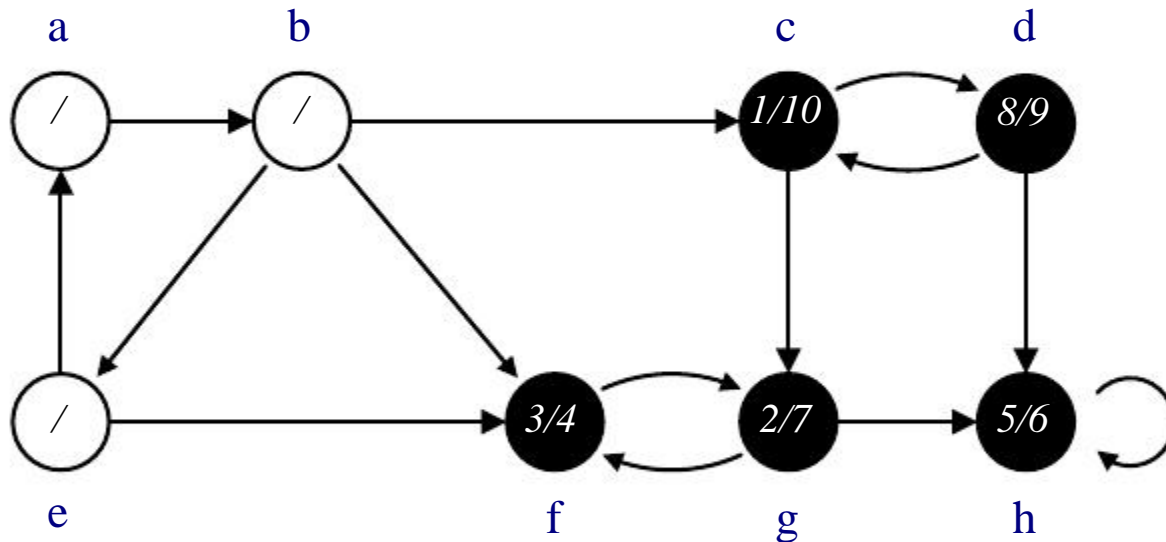
6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

# Busca em Profundidade

- Desempilhou: DFS(G) - CP: linha 4 - próximo:  $u = a$
- O vértice  $a$  é BRANCO, então chama: DFS\_VISIT( $a$ );
- Empilha DFS(G) - CP: linha 4 - próximo:  $u = b$



Lista: [c,a,b,d,e,f,g,h]



tempo = 10

DFS(G)

1 para cada vértice  $u \leftarrow V[G]$

2 cor[u] ← BRANCO

3 tempo ← 0

4 para cada vértice  $u \in V[G]$

5 se cor[u] = BRANCO

6 DFS-VISIT(u)

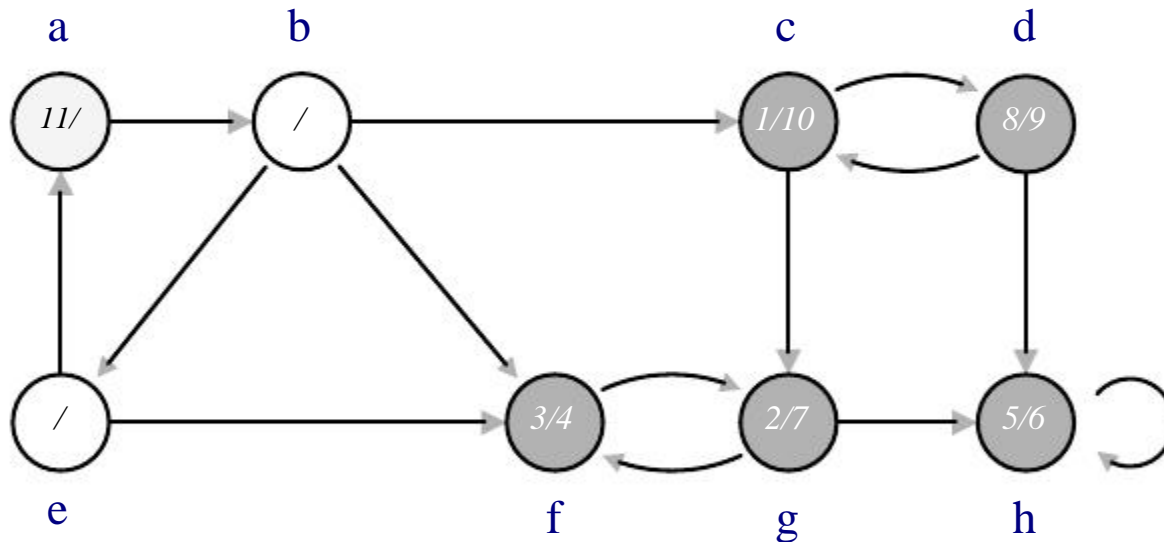
Pilha de execução:  
VAZIA

# Busca em Profundidade

- Marca a de CINZA;
- Incrementa o tempo;
- Atribui o tempo de descoberta de a...

*DFS - VISIT(u)*

- 1  $cor[u] \leftarrow CINZA$
- 2  $tempo \leftarrow tempo + 1$
- 3  $d[u] \leftarrow tempo$
- 4 para cada vértice  $v \in Adj(u)$
- 5     se  $cor[v] = BRANCO$
- 6         *DFS - VISIT(v)*
- 7  $cor[u] \leftarrow PRETO$
- 8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$



Lista: [c,a,b,d,e,f,g,h]

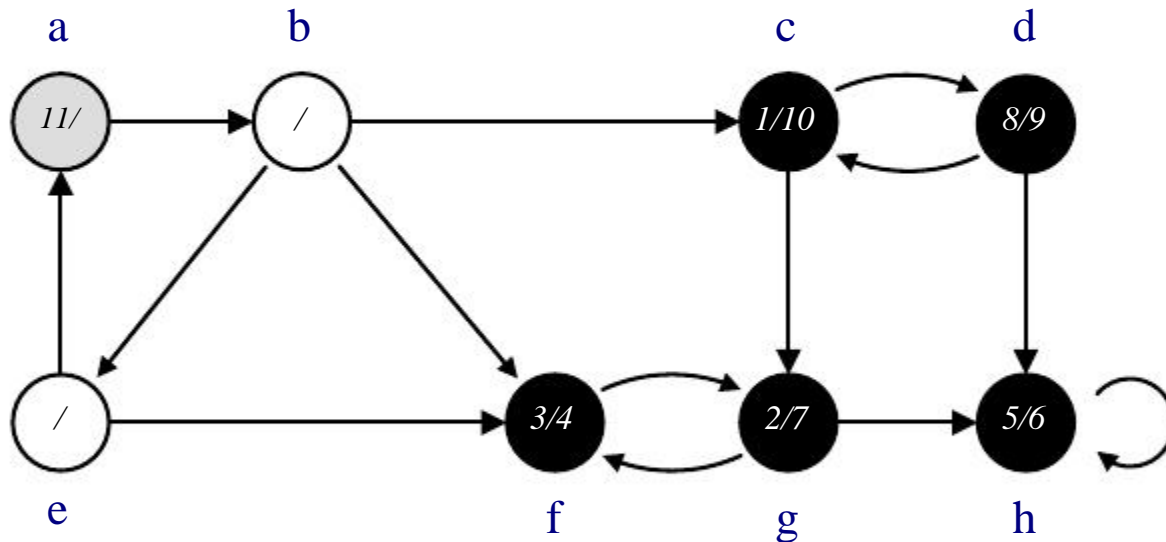
Pilha de execução:

DFS(G) - CP: linha 4 - próximo: u = b

tempo = 10 => 11

# Busca em Profundidade

- Para todos os adjacentes do vértice  $a = \{b\}$
- Se cor de  $b$  for BRANCA, então,  $\text{DFS\_VISIT}(b)$
- Empilha  $\text{DFS\_VISIT}(a)$ : CP=4.



Lista: [c,a,b,d,e,f,g,h]

Pilha de execução:

DFS(G) - CP: linha 4 - próximo:  $u = b$

*DFS-VISIT(u)*

1  $\text{cor}[u] \leftarrow \text{CINZA}$

2  $\text{tempo} \leftarrow \text{tempo} + 1$

3  $d[u] \leftarrow \text{tempo}$

4 para cada vértice  $v \in \text{Adj}(u)$

5 se  $\text{cor}[v] = \text{BRANCO}$

6  $\text{DFS-VISIT}(v)$

7  $\text{cor}[u] \leftarrow \text{PRETO}$

8  $f[u] \leftarrow \text{tempo} \leftarrow (\text{tempo} + 1)$

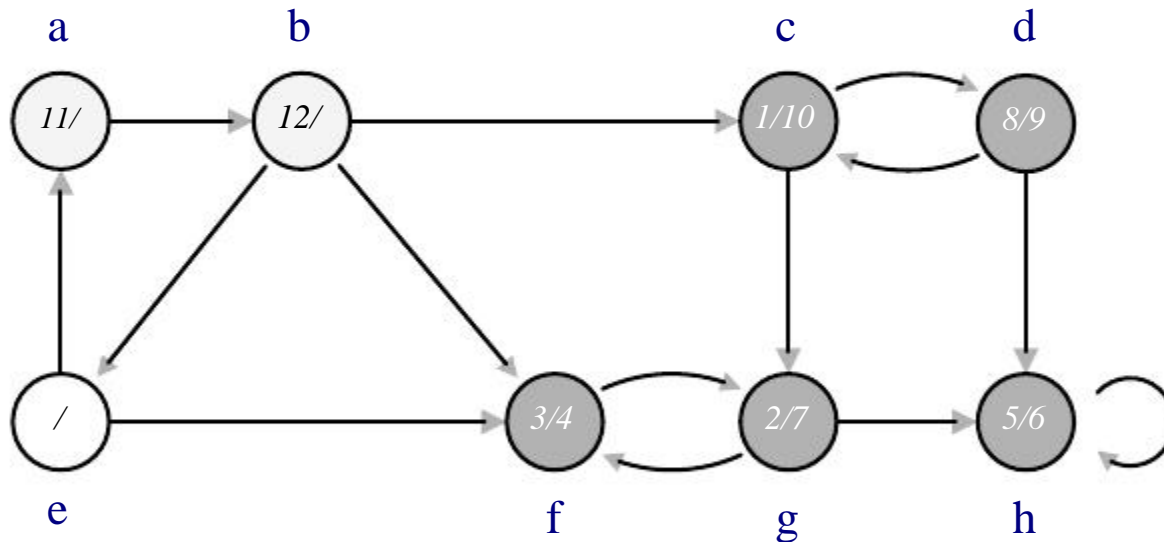
$\text{tempo} = 11$

# Busca em Profundidade

- Marca b de CINZA;
- Incrementa o tempo;
- Atribui o tempo de descoberta de b...

DFS - VISIT(u)

- 1 cor[u] ← CINZA
- 2 tempo ← tempo + 1
- 3 d[u] ← tempo
- 4 para cada vértice  $v \in \text{Adj}(u)$
- 5     se cor[v] = BRANCO
- 6         DFS - VISIT(v)
- 7 cor[u] ← PRETO
- 8 f[u] ← tempo ← (tempo + 1)



Lista: [c,a,b,d,e,f,g,h]

Pilha de execução:

DFS\_VISIT(a) - CP: linha 4

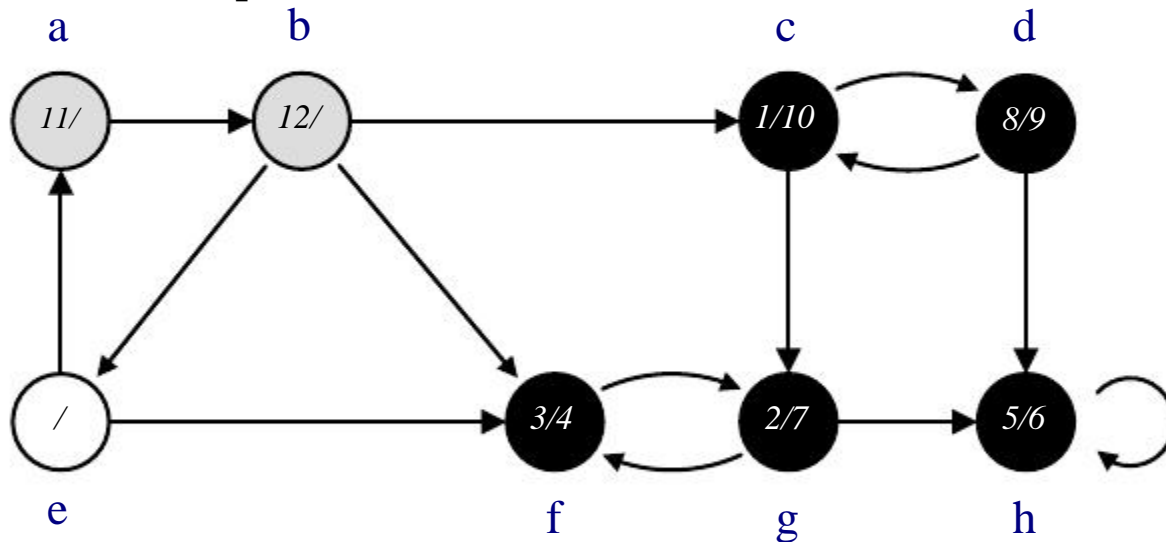
DFS(G) - CP: linha 4 - próximo: u = b

tempo = 11 => 12



# Busca em Profundidade

- Para todos os adjacentes do vértice  $b = \{c, e, f\}$
- A cor de  $c$  e de  $f$  é PRETA, então pula!
- Como a cor de  $e$  é BRANCA, então, DFS\_VISIT( $e$ )
- Empilha DFS\_VISIT( $b$ ): CP=4.



Lista: [c,a,b,d,e,f,g,h]

DFS -VISIT( $u$ )

1 cor[ $u$ ] ← CINZA

2 tempo ← tempo + 1

3 d[ $u$ ] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[ $v$ ] = BRANCO

6 DFS -VISIT( $v$ )

7 cor[ $u$ ] ← PRETO

8 f[ $u$ ] ← tempo ← (tempo + 1)

Pilha de execução:

DFS\_VISIT( $a$ ) - CP: linha 4

DFS( $G$ ) - CP: linha 4 - próximo:  $u = b$

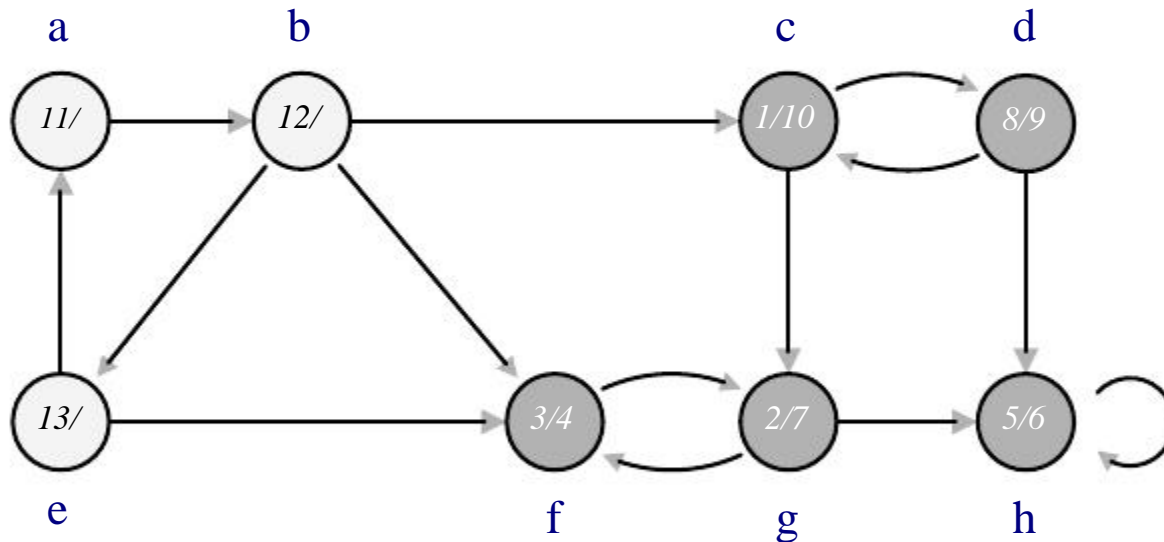
tempo = 12

# Busca em Profundidade

- Marca e de CINZA;
- Incrementa o tempo;
- Atribui o tempo de descoberta de e...

DFS - VISIT(u)

- 1 cor[u] ← CINZA
- 2 tempo ← tempo + 1
- 3 d[u] ← tempo
- 4 para cada vértice  $v \in \text{Adj}(u)$
- 5    se cor[v] = BRANCO
- 6        DFS - VISIT(v)
- 7 cor[u] ← PRETO
- 8 f[u] ← tempo ← (tempo + 1)



Lista: [c,a,b,d,e,f,g,h]

tempo = 12 => 13

**Pilha de execução:**

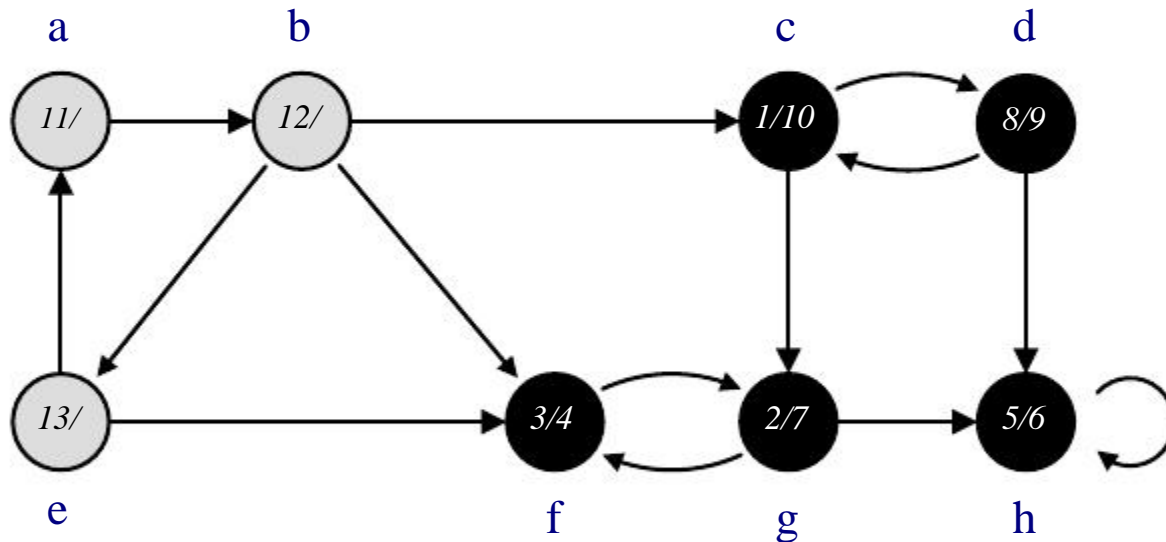
DFS\_VISIT(b) - CP: linha 4

DFS\_VISIT(a) - CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = b

# Busca em Profundidade

- Avalia o único adjacente do vértice  $e = \{f\}$ ;
- O vértice  $f$  não é BRANCO, então, finaliza a busca sobre o vértice  $e$ .



Lista: [c,a,b,d,e,f,g,h]

tempo = 13

DFS - VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS - VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

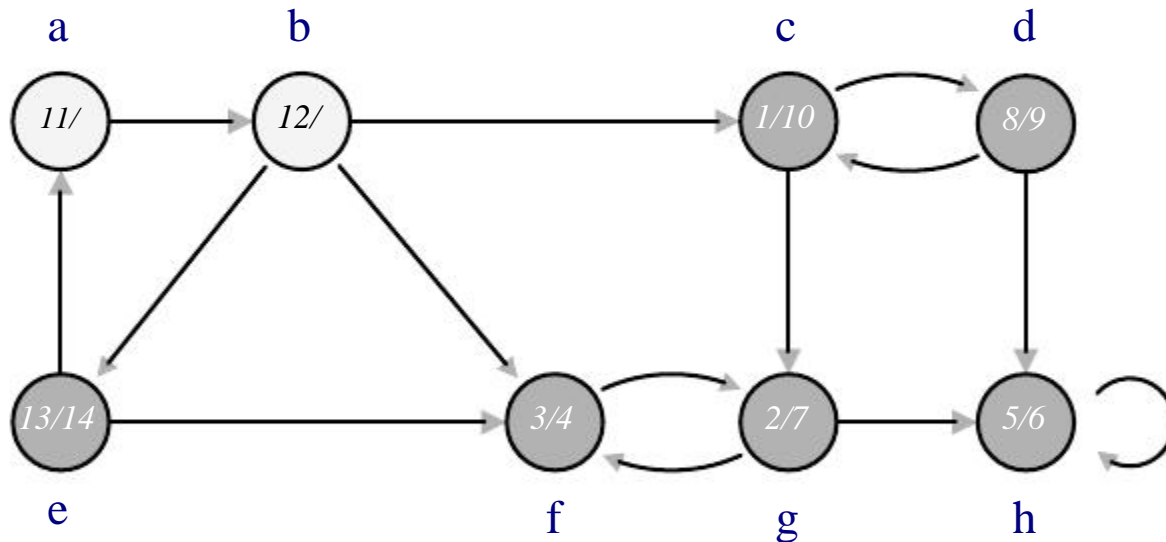
DFS\_VISIT(b) - CP: linha 4

DFS\_VISIT(a) - CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = b

# Busca em Profundidade

- Marca e de PRETO; Incrementa o tempo;
- Atribui o tempo de finalização de e;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

tempo = 13 => 14

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

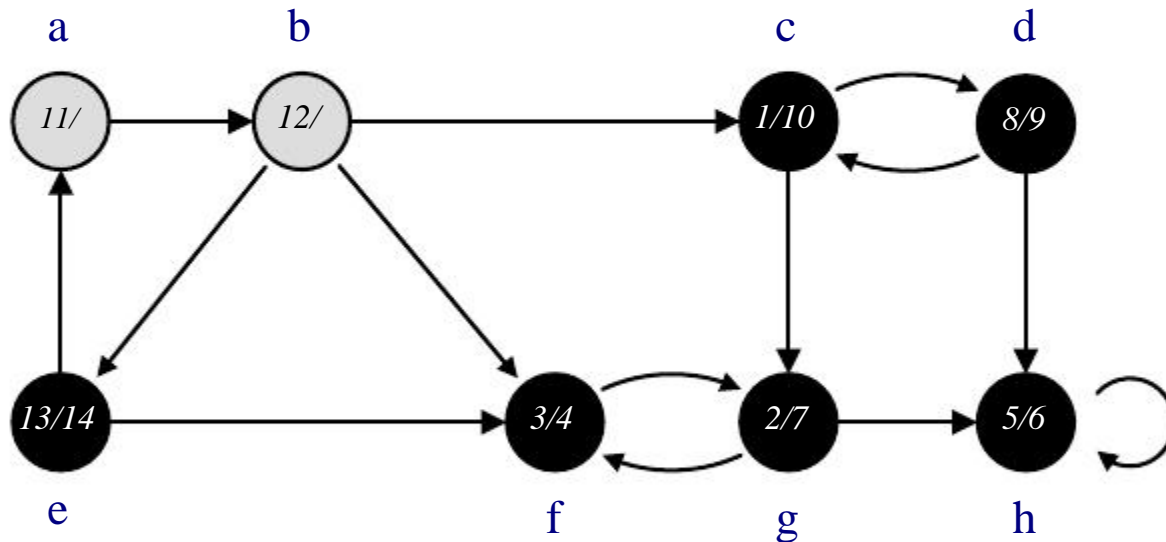
DFS\_VISIT(b) - CP: linha 4

DFS\_VISIT(a) - CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = b

# Busca em Profundidade

- Não possui mais adjacentes;
- Assim finaliza busca em b...



Lista: [c,a,b,d,e,f,g,h]

tempo = 14

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5     se cor[v] = BRANCO

6         DFS -VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

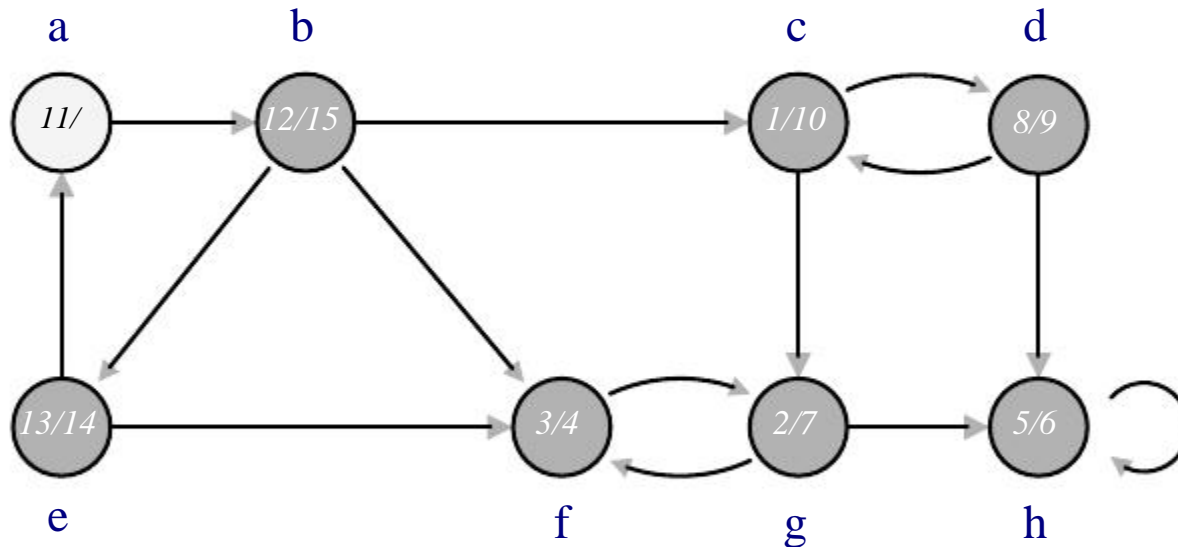
Pilha de execução:

DFS\_VISIT(a) - CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = b

# Busca em Profundidade

- Marca b de PRETO; Incrementa o tempo;
- Atribui o tempo de finalização de b;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

*DFS - VISIT(u)*

1  $cor[u] \leftarrow CINZA$

2  $tempo \leftarrow tempo + 1$

3  $d[u] \leftarrow tempo$

4 para cada vértice  $v \in Adj(u)$

5 se  $cor[v] = BRANCO$

6 *DFS - VISIT(v)*

7  $cor[u] \leftarrow PRETO$

8  $f[u] \leftarrow tempo \leftarrow (tempo + 1)$

**Pilha de execução:**

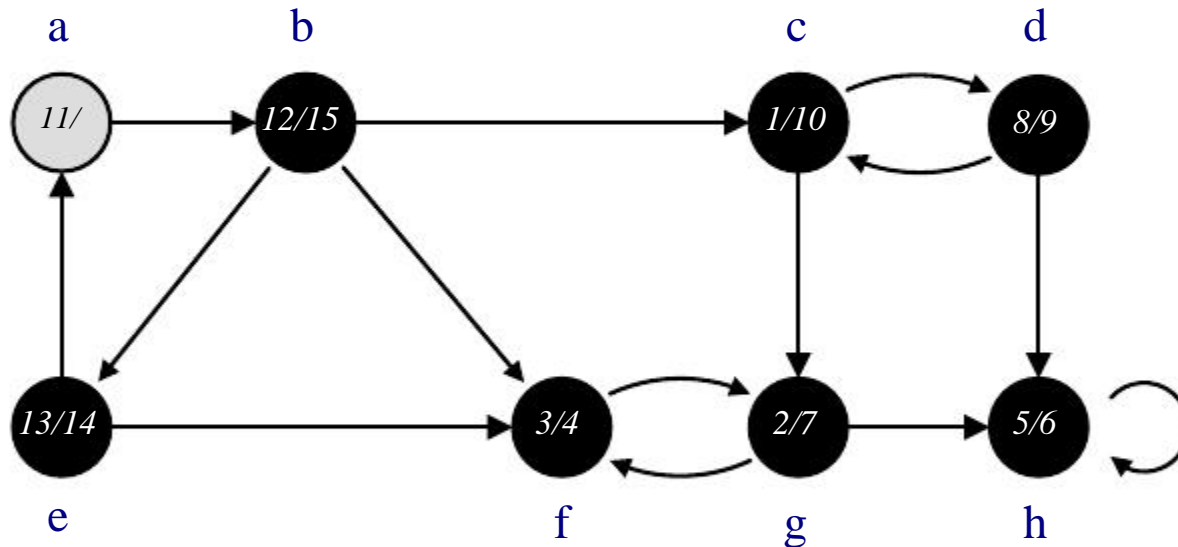
DFS\_VISIT(a) - CP: linha 4

DFS(G) - CP: linha 4 - próximo: u = b

tempo = 14 => 15

# Busca em Profundidade

- Desempilhou: DFS\_VISIT(a) - CP: linha 4
- Mas o vértice a não possui mais adjacentes BRANCOS;
- Assim, finaliza a busca sobre a...



Lista: [c,a,b,d,e,f,g,h]

tempo = 15

DFS -VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS -VISIT(v)

7 cor[u] ← PRETO

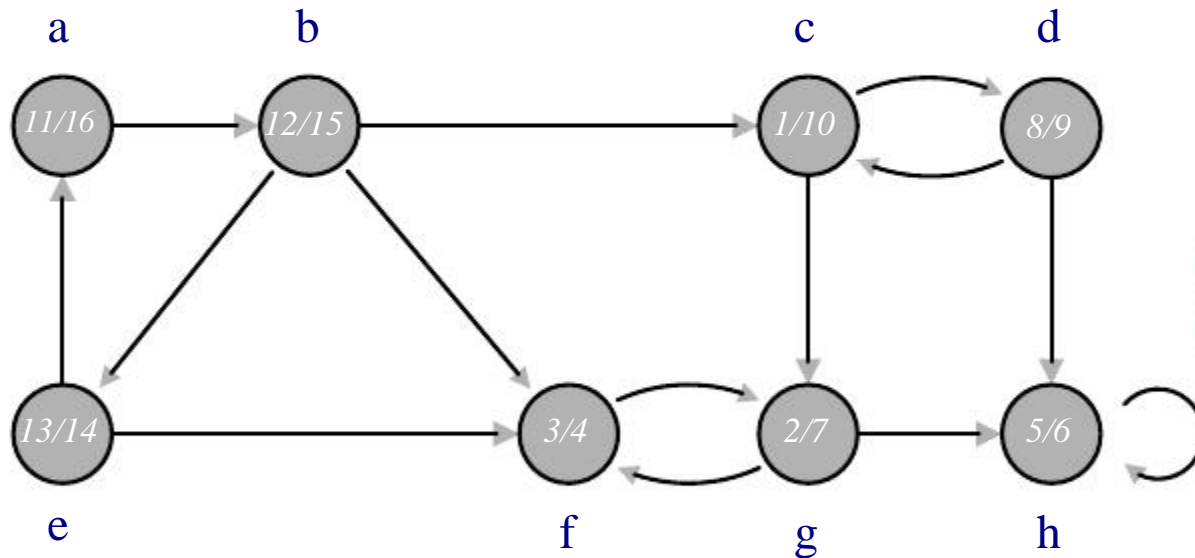
8 f[u] ← tempo ← (tempo + 1)

Pilha de execução:

DFS(G) - CP: linha 4 - próximo: u = b

# Busca em Profundidade

- Marca a de PRETO; Incrementa o tempo;
- Atribui o tempo de finalização de a;
- Desempilha...



Lista: [c,a,b,d,e,f,g,h]

DFS –VISIT(u)

1 cor[u] ← CINZA

2 tempo ← tempo + 1

3 d[u] ← tempo

4 para cada vértice  $v \in \text{Adj}(u)$

5 se cor[v] = BRANCO

6 DFS –VISIT(v)

7 cor[u] ← PRETO

8 f[u] ← tempo ← (tempo + 1)

**Pilha de execução:**

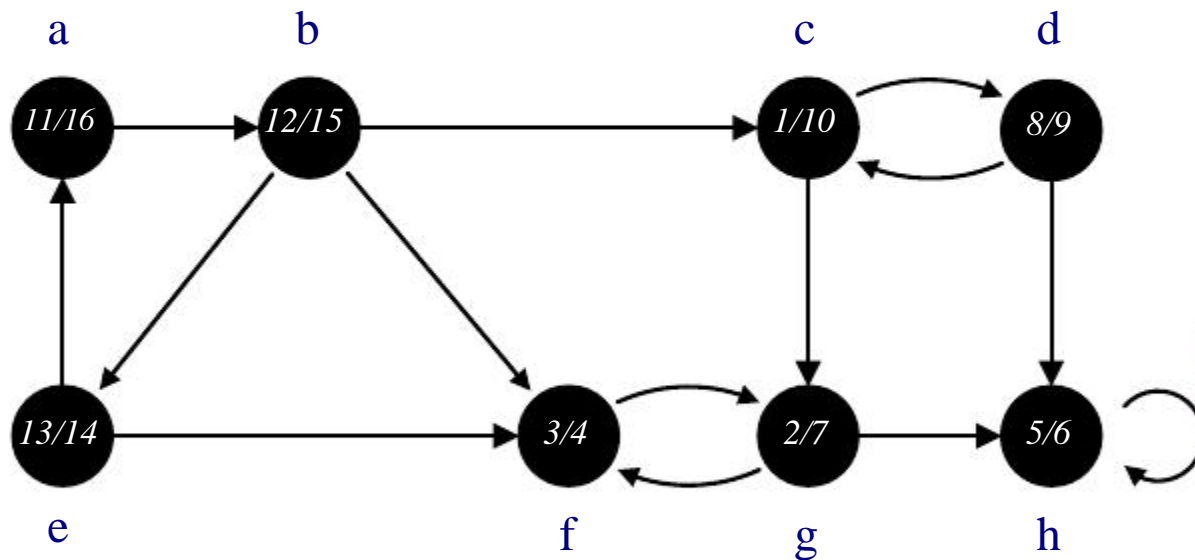
DFS(G) - CP: linha 4 – próximo: u = b

tempo = 15 => 16



# Busca em Profundidade

- Desempilhou: DFS(G) - CP: linha 4 – próximo:  $u = b$
- Mas todos os vértices não são mais BRANCOS;
- Assim, a DFS(G) termina verificando a cor de todos os vértices restantes...



Lista: [c,a,b,d,e,f,g,h]



tempo = 16

*DFS(G)*

1 para cada vértice  $u \leftarrow V[G]$

2  $cor[u] \leftarrow BRANCO$

3  $tempo \leftarrow 0$

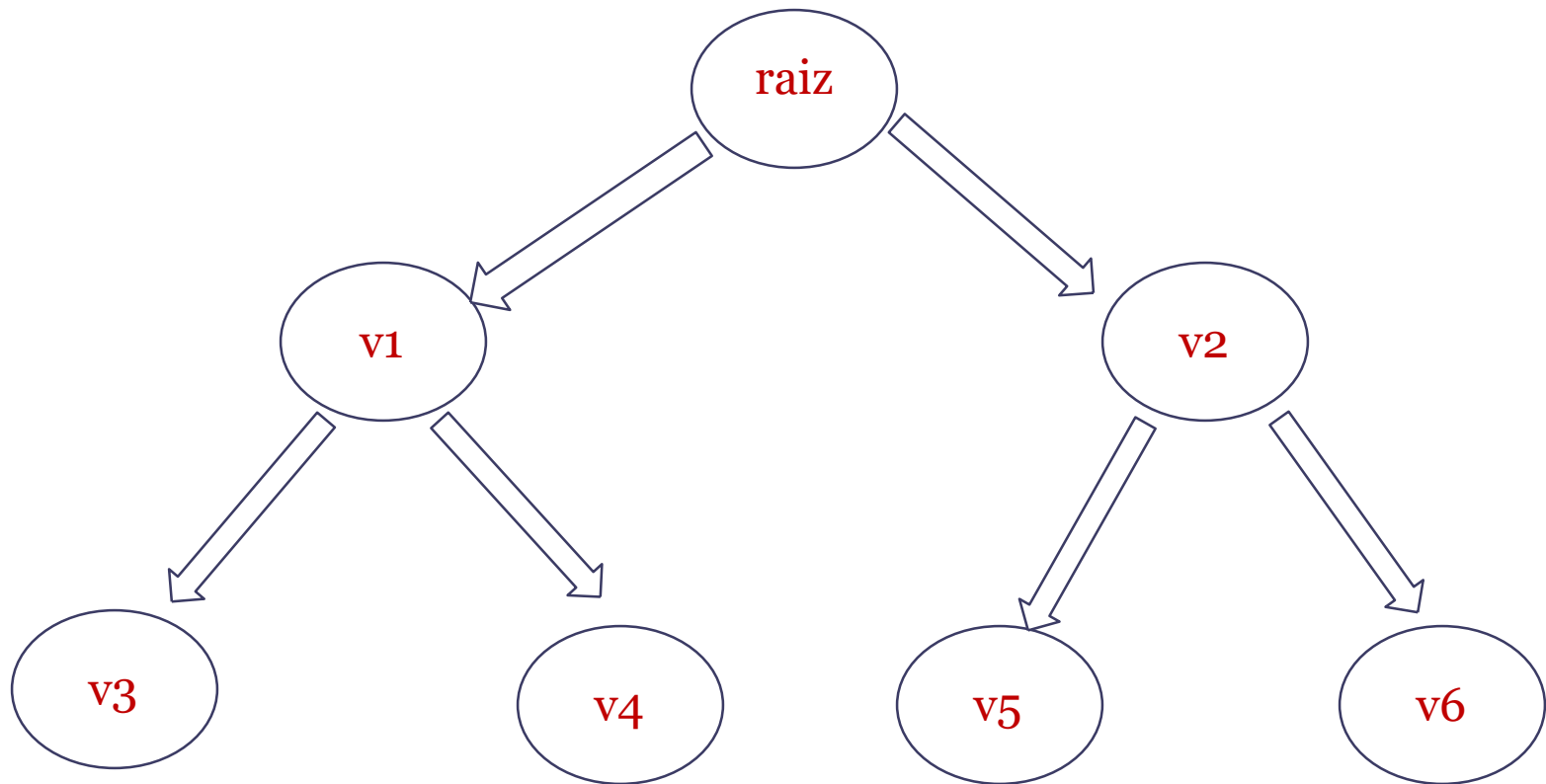
4 para cada vértice  $u \in V[G]$

5 se  $cor[u] = BRANCO$

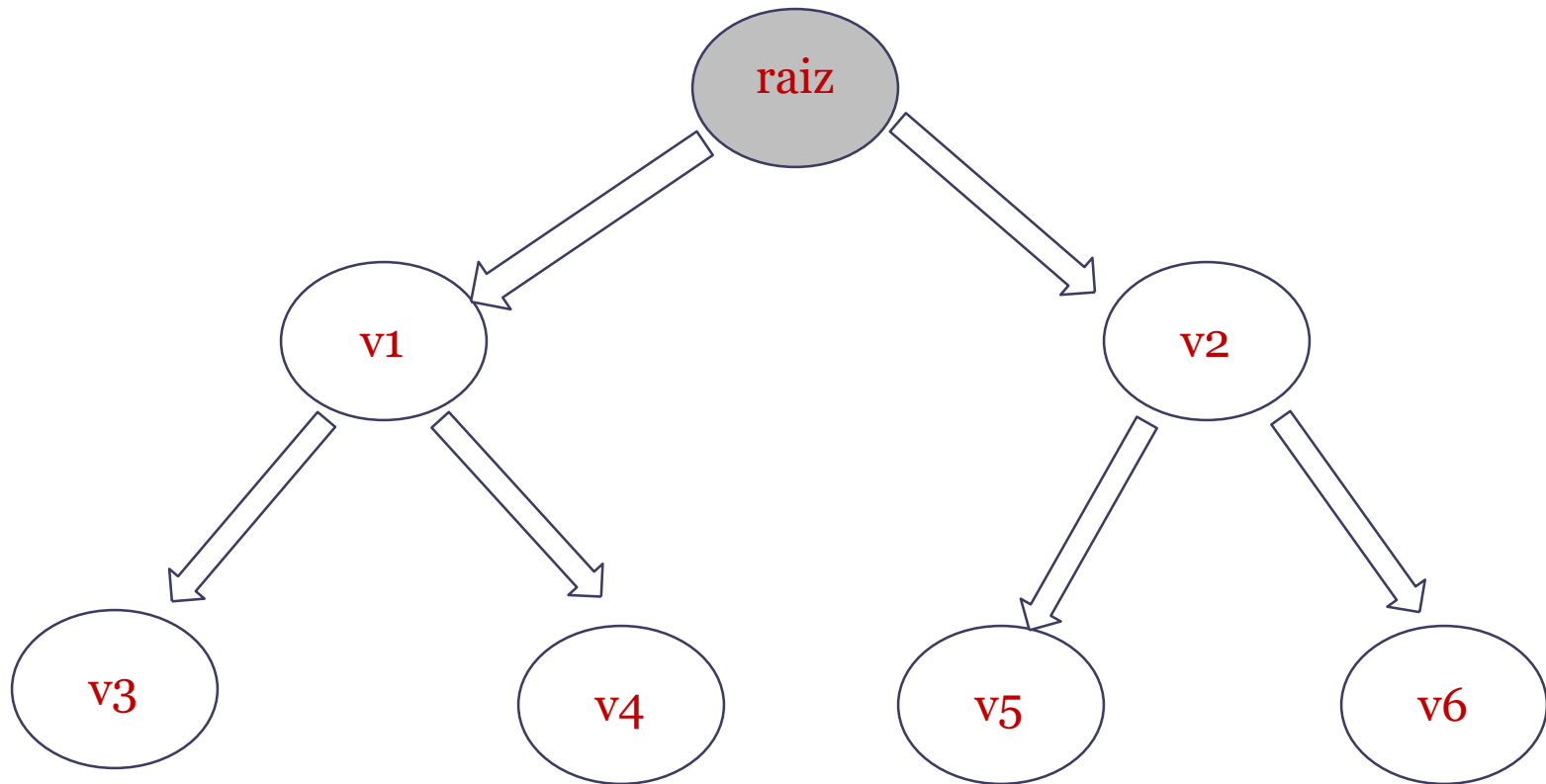
6  $DFS-VISIT(u)$

Pilha de execução:  
VAZIA

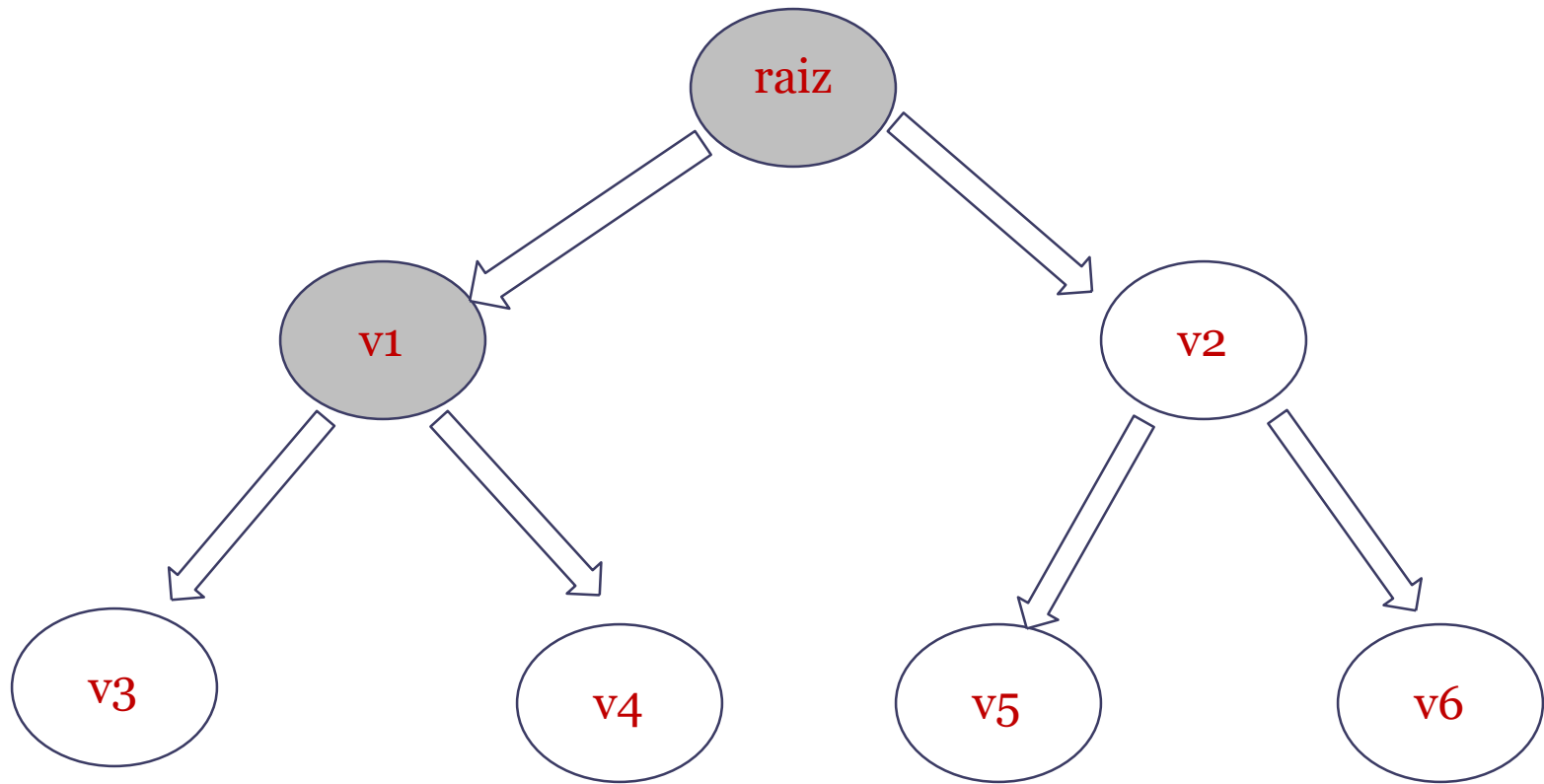
# Aplicando Busca em Profundidade em uma Árvore



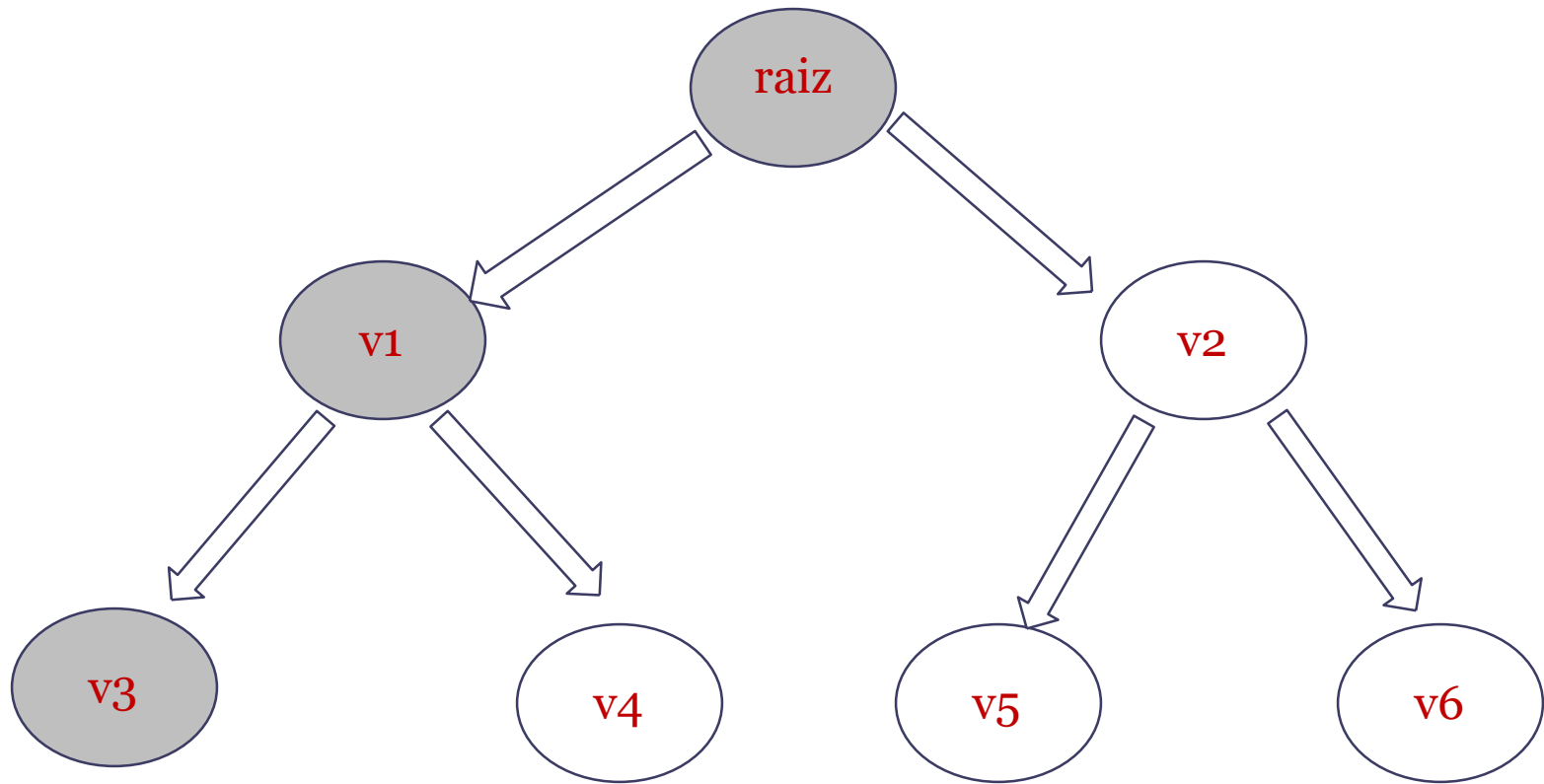
# Aplicando Busca em Profundidade em uma Árvore



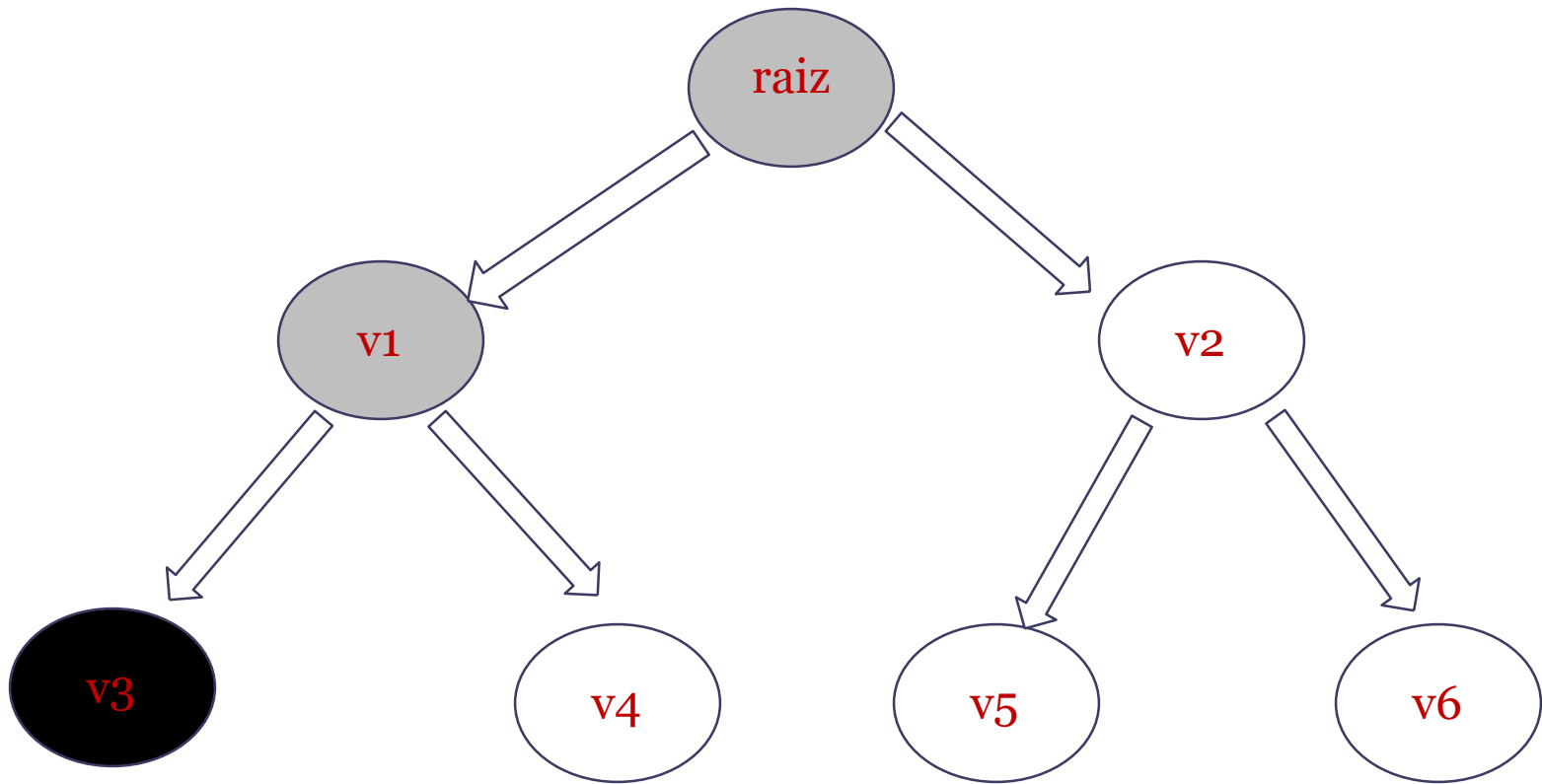
# Aplicando Busca em Profundidade em uma Árvore



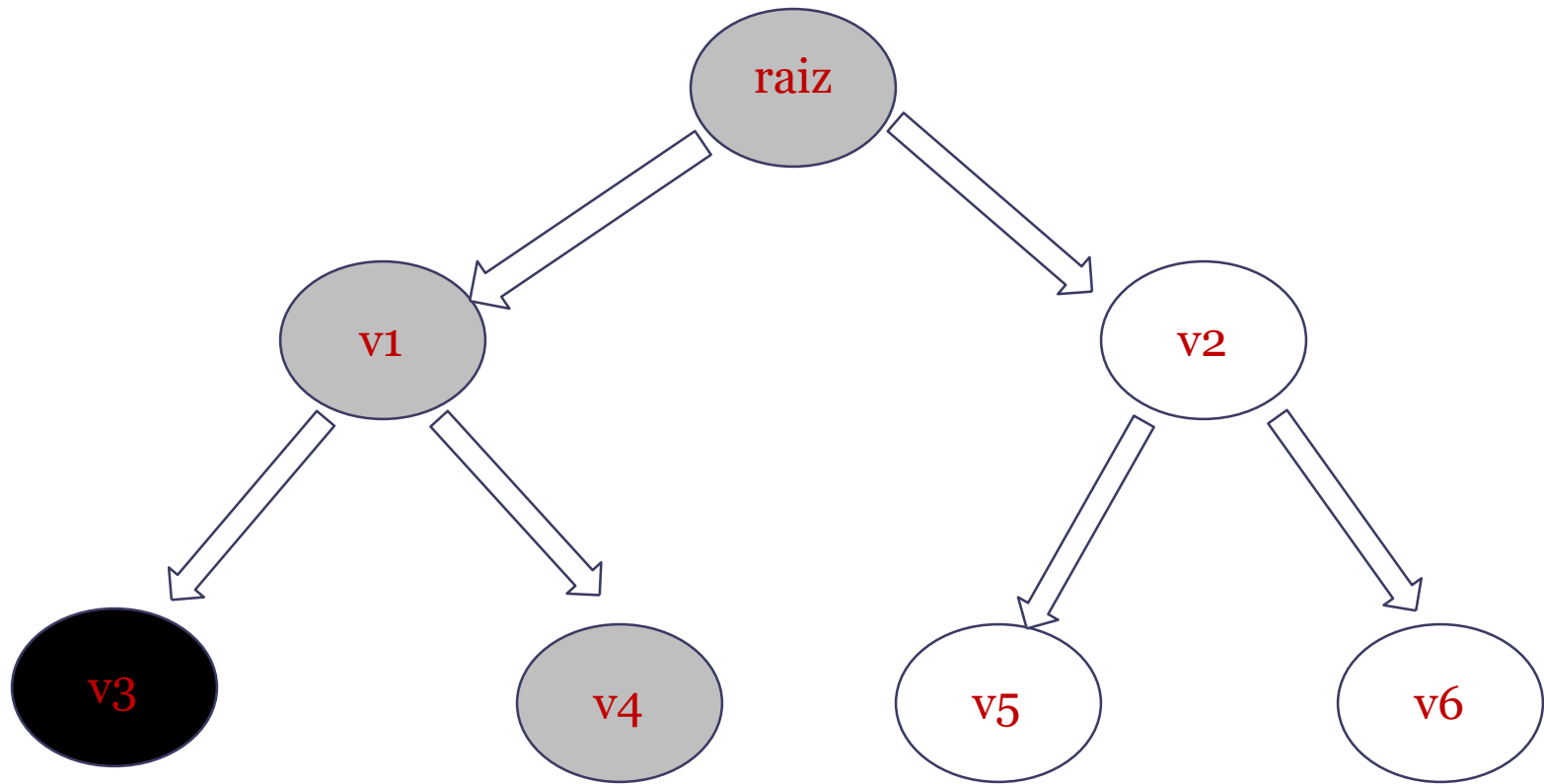
# Aplicando Busca em Profundidade em uma Árvore



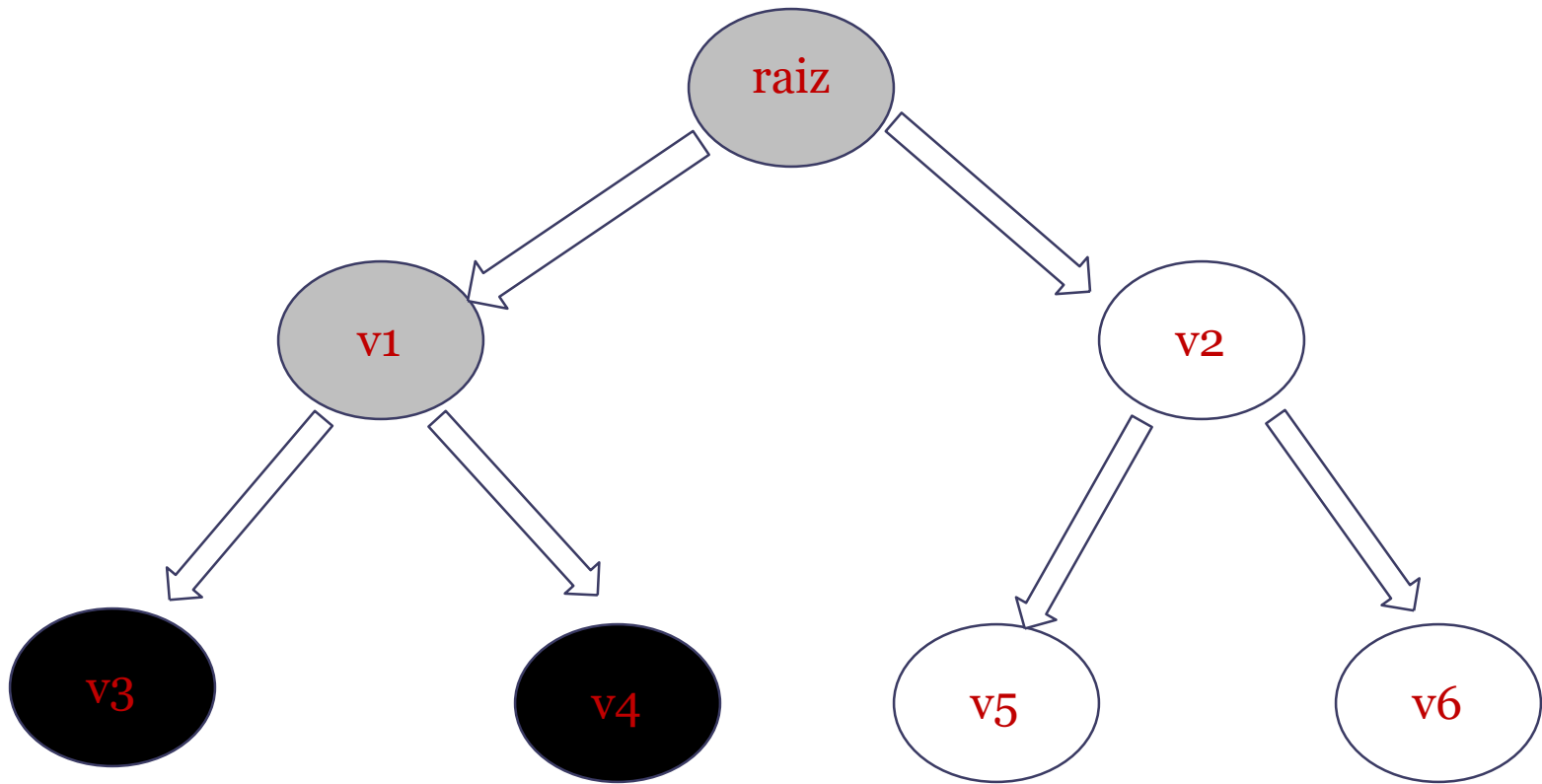
# Aplicando Busca em Profundidade em uma Árvore



# Aplicando Busca em Profundidade em uma Árvore

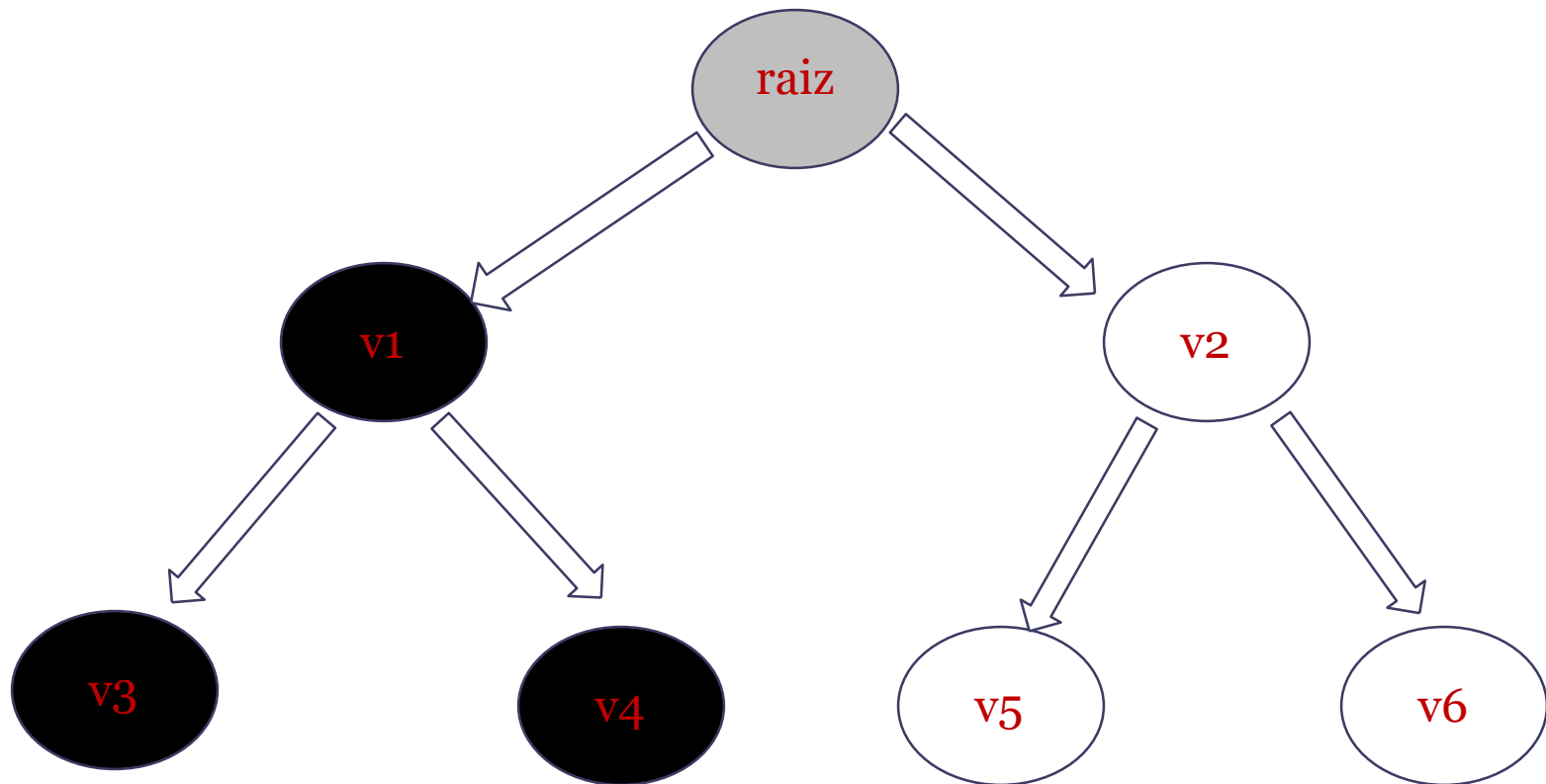


# Aplicando Busca em Profundidade em uma Árvore

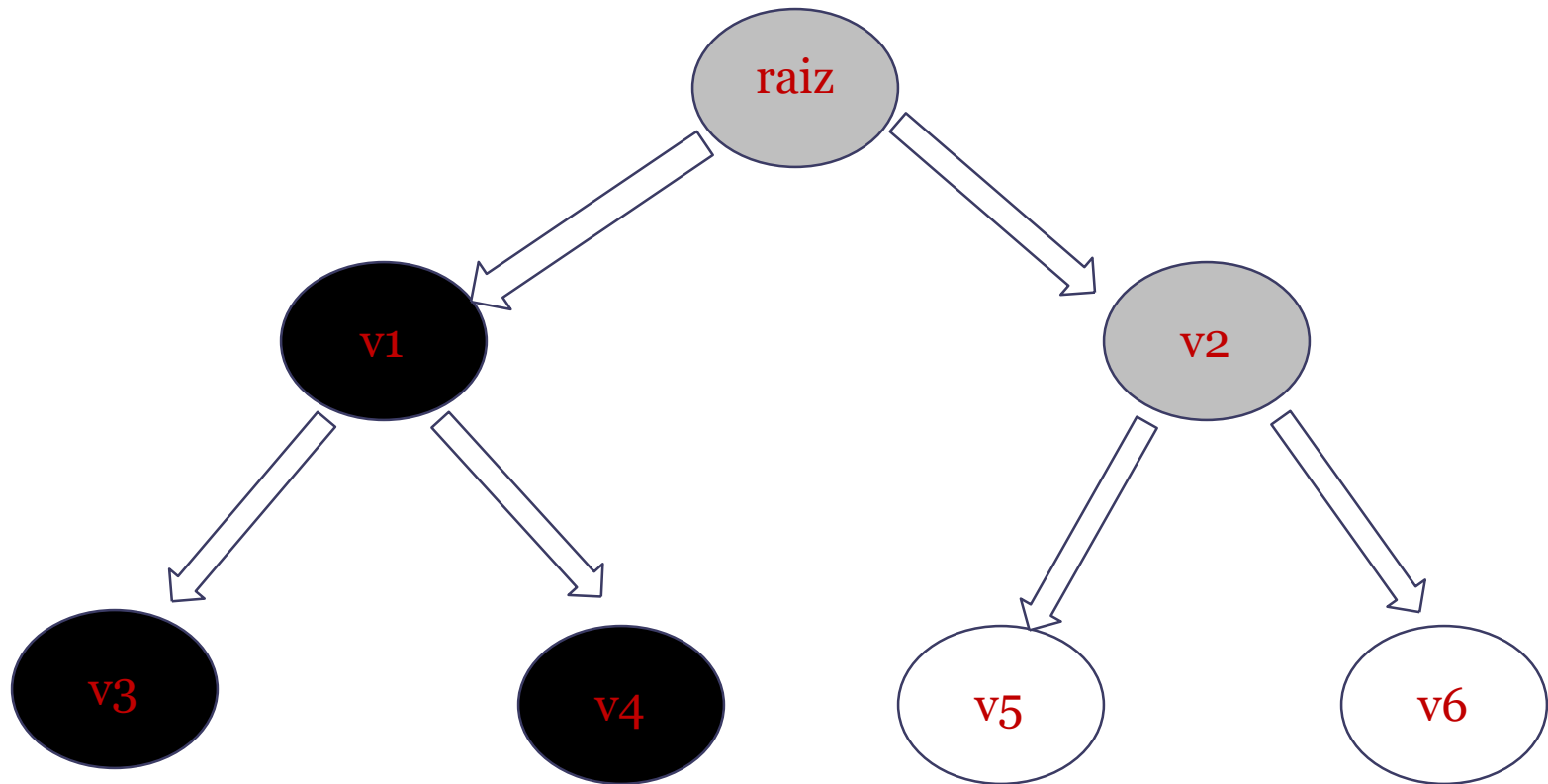




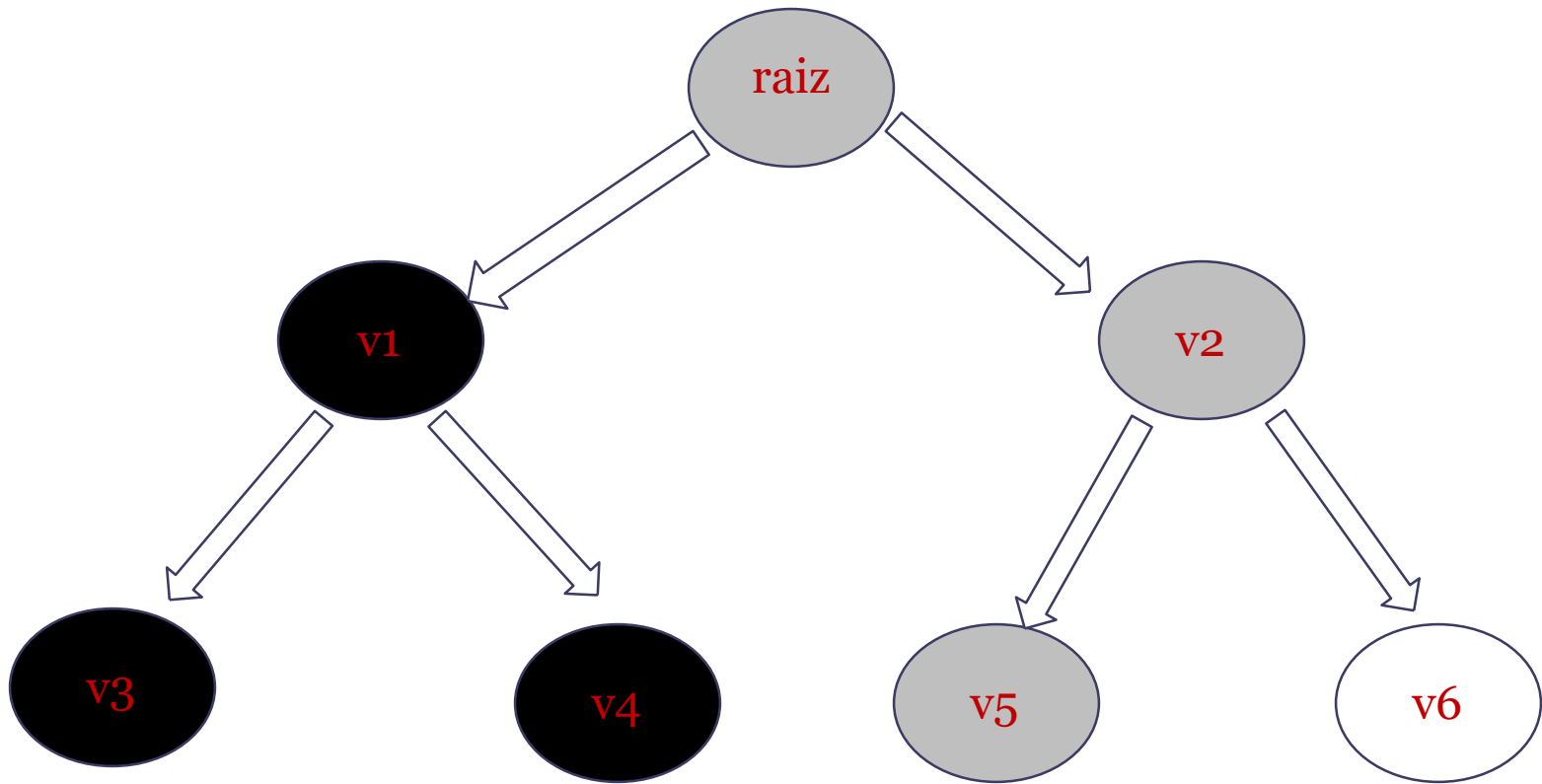
# Aplicando Busca em Profundidade em uma Árvore



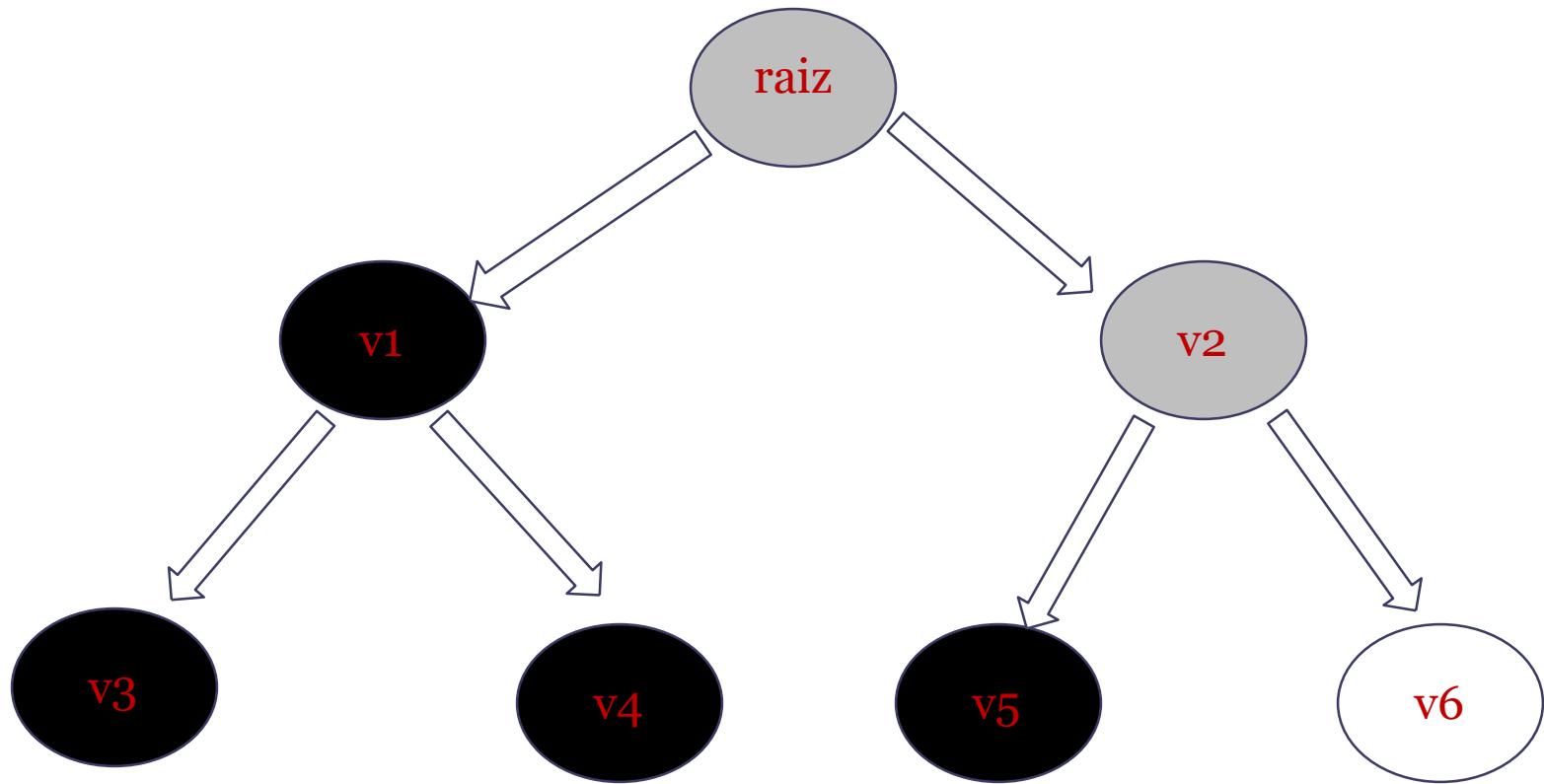
# Aplicando Busca em Profundidade em uma Árvore



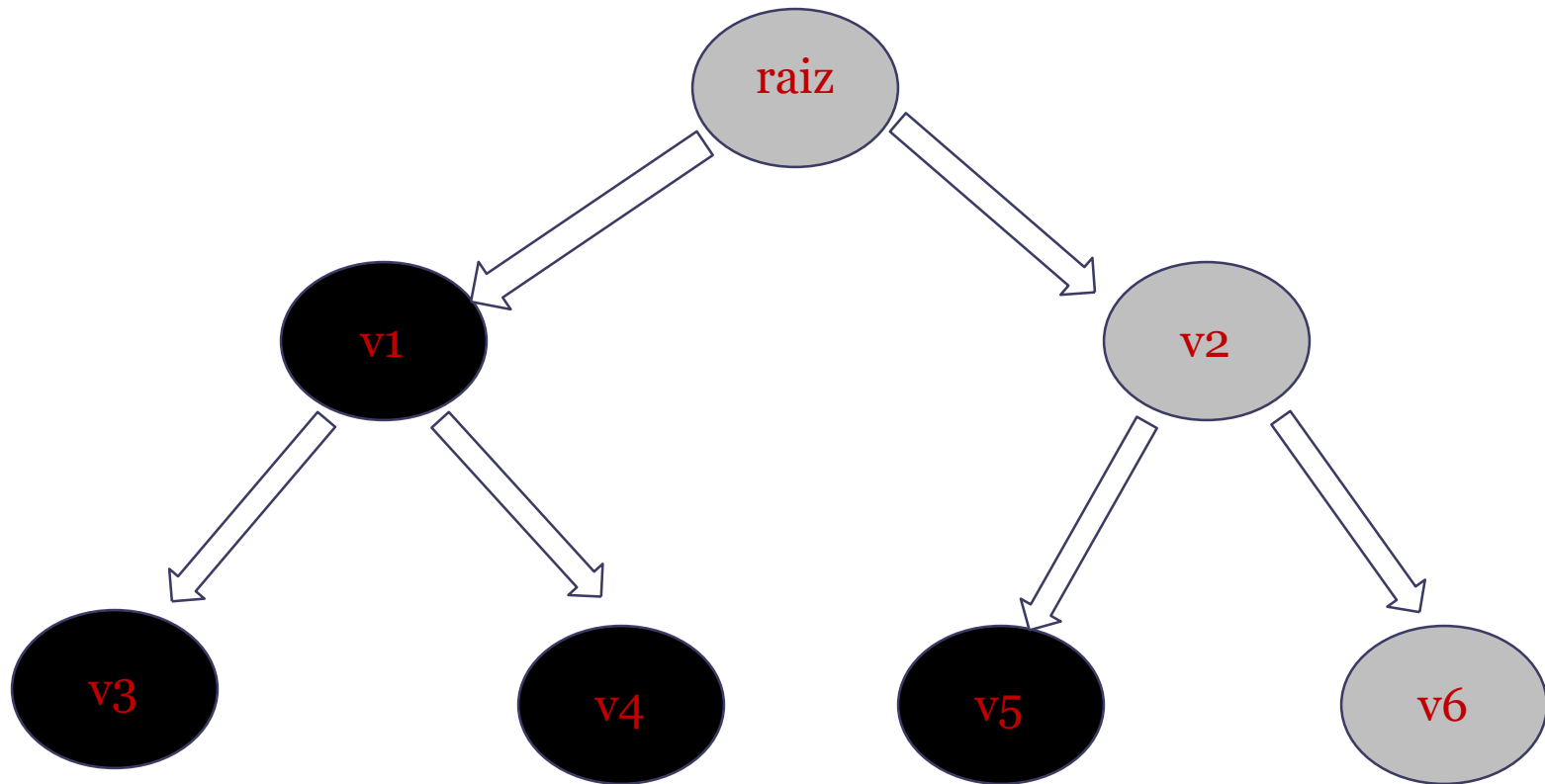
# Aplicando Busca em Profundidade em uma Árvore



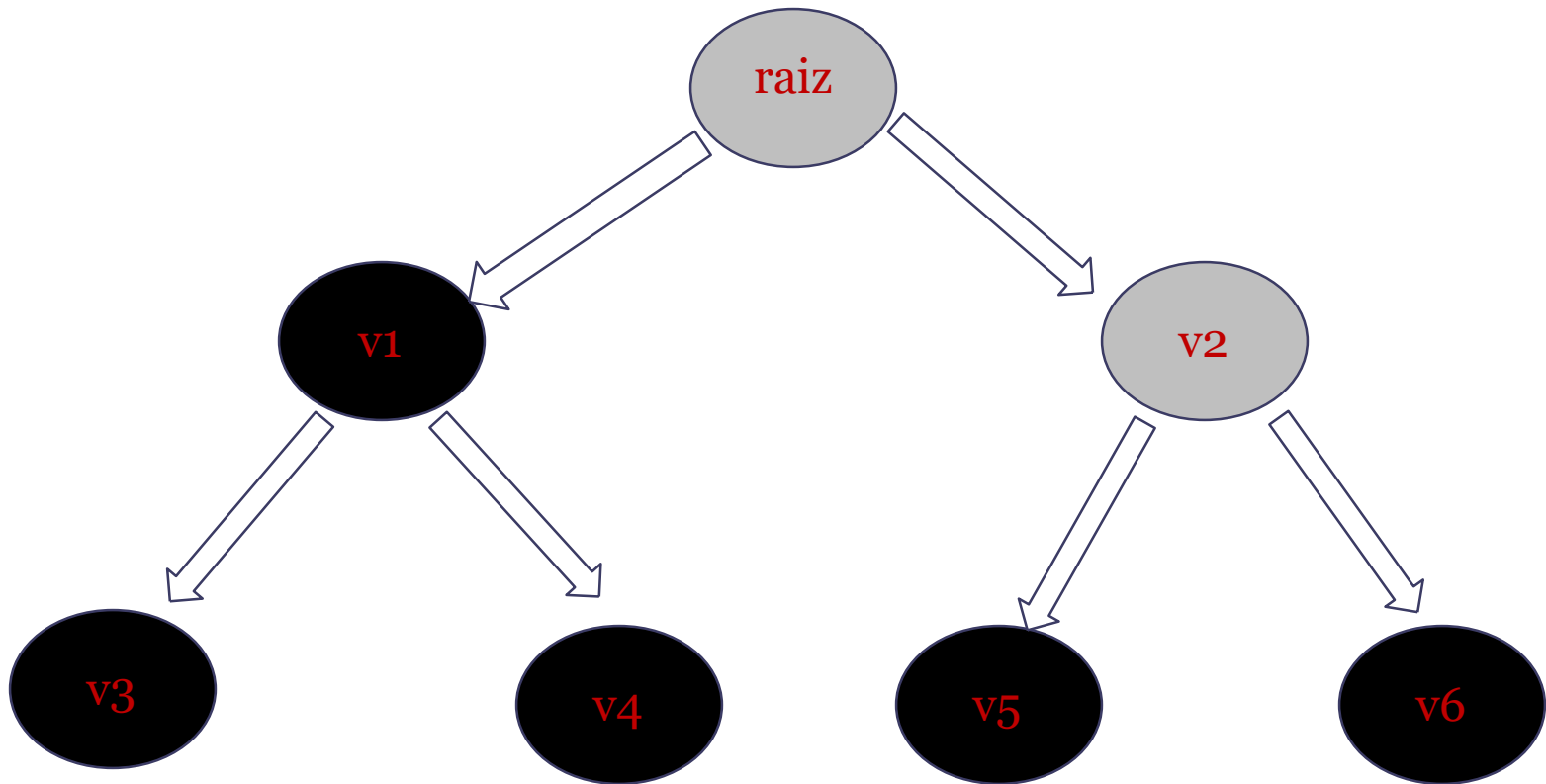
# Aplicando Busca em Profundidade em uma Árvore



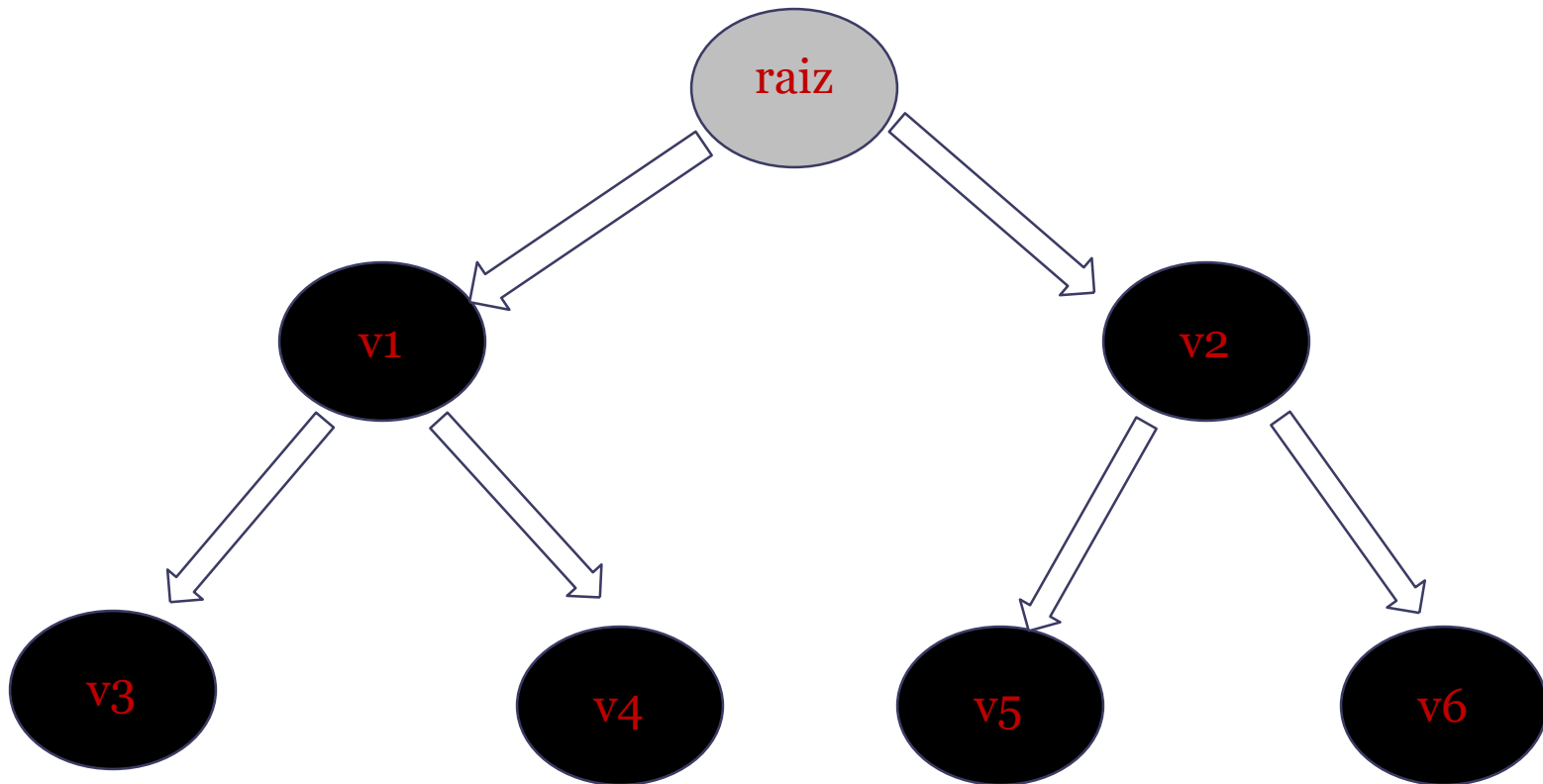
# Aplicando Busca em Profundidade em uma Árvore



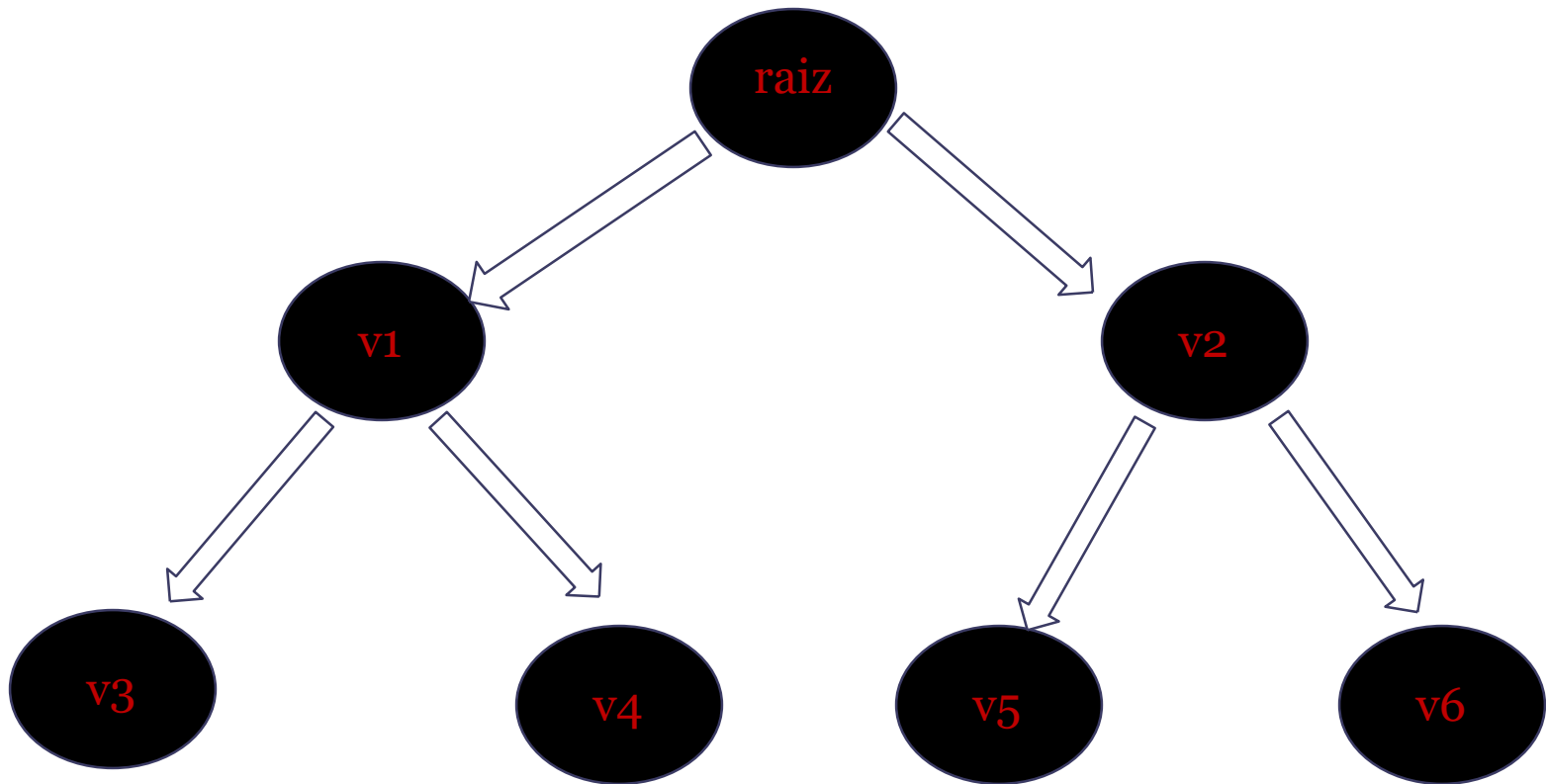
# Aplicando Busca em Profundidade em uma Árvore



# Aplicando Busca em Profundidade em uma Árvore



# Aplicando Busca em Profundidade em uma Árvore



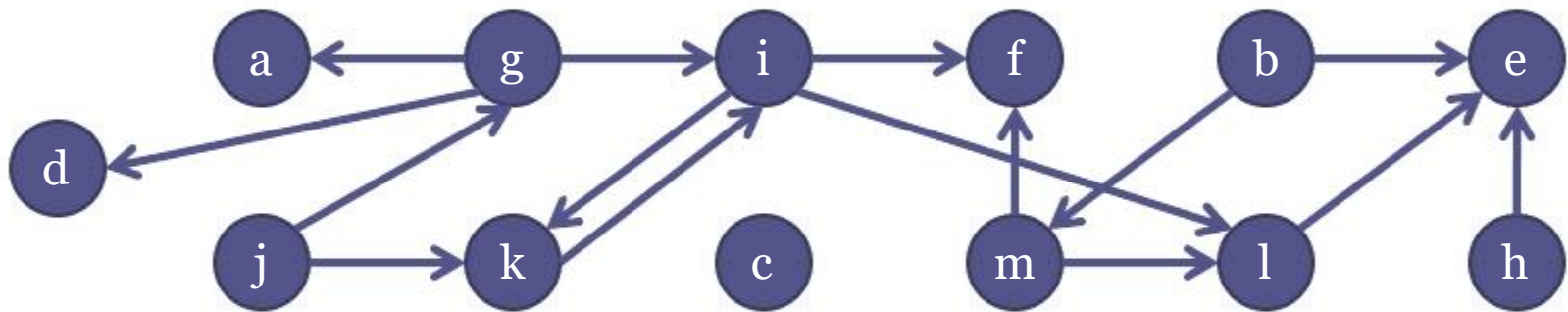


# Exercícios



# Exercício 01

- Realizar a busca em profundidade sobre o seguinte grafo:



- Considerar a ordem alfabética para qualquer decisão de ordem de elementos de conjuntos em qualquer parte do algoritmo.
- Ao final, indique o tempo de descoberta e de finalização de cada vértice do grafo

# Bibliografia

- CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; (2002). Algoritmos – Teoria e Prática. Tradução da 2ª edição americana. Rio de Janeiro. Editora Campus.
  - Capítulo 22.3
- ZIVIANI, N. (2007). Projeto e Algoritmos com implementações em Java e C++. São Paulo. Editora Thomson;
  - Capítulo 7.3

