Grafos

Árvores: Diâmetro

Prof. Edson Alves

Faculdade UnB Gama

Definição de diâmetro

Definição de diâmetro

Seja $G(V, E)$ um grafo. O **diâmetro** de G é igual ao maior dentre todos os tamanhos dos caminhos entre os pares de vértices $u, v \in V$.

Características do diâmetro

Características do diâmetro

- ⋆ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

Características do diâmetro

$\star$ **O caminho cujo tamanho determina o diâmetro não é, necessariamente, único**

$\star$ **Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente**

Características do diâmetro

- $\star$ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

- $\star$ Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente

- $\star$ Porém, é possível determinar o diâmetro usando programação dinâmica ou duas DFS

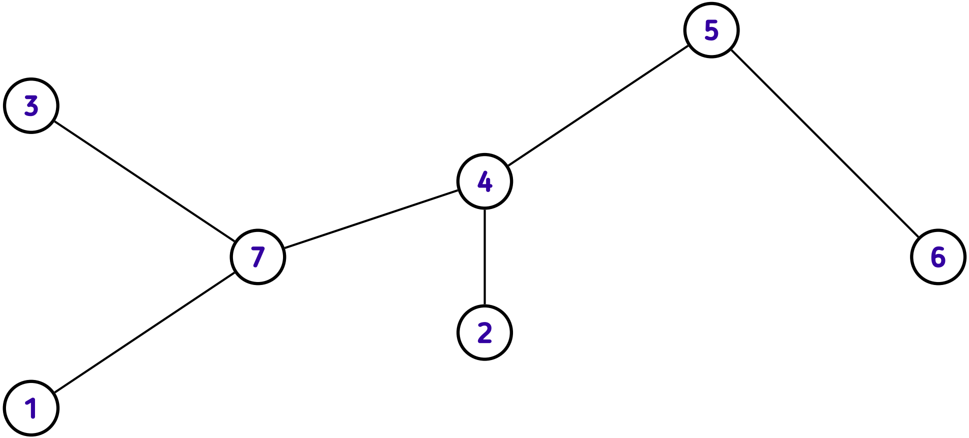
Características do diâmetro

$\star$ O caminho cujo tamanho determina o diâmetro não é, necessariamente, único

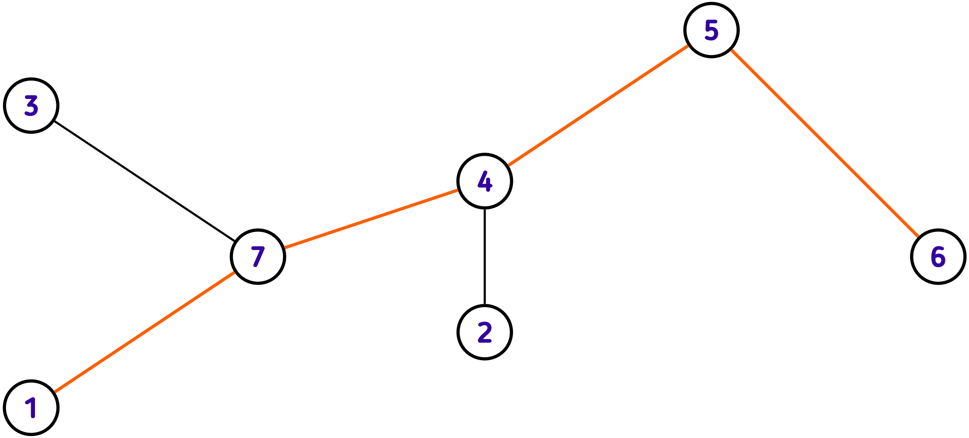
$\star$ Computar todos os tamanho com Floyd-Warshall em $O(V^3)$ e determinar o maior dentre eles em $O(V^2)$ determina o diâmetro corretamente

$\star$ Porém, é possível determinar o diâmetro usando programação dinâmica ou duas DFS

$\star$ Em ambos casos, a complexidade é $O(V)$


5
3
4
7
6
2
1

Diâmetro: 4



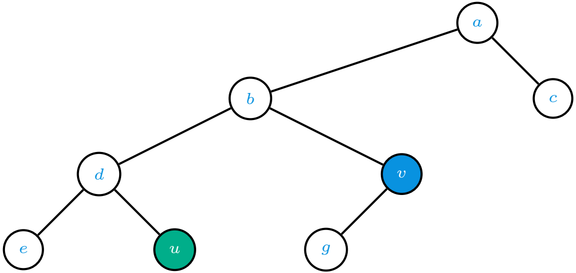
Definição de pico de um caminho

Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível baixo em T.

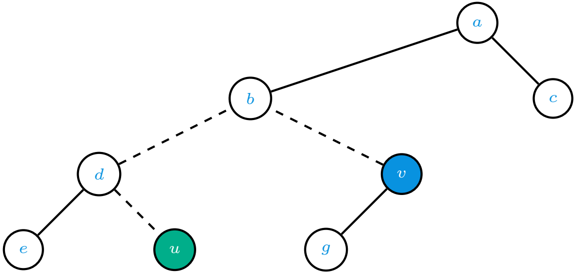
Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível baixo em T.



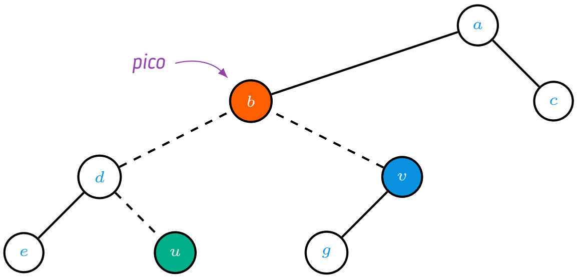
Definição de pico de um caminho

Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível baixo em T.



Definição de pico de um caminho

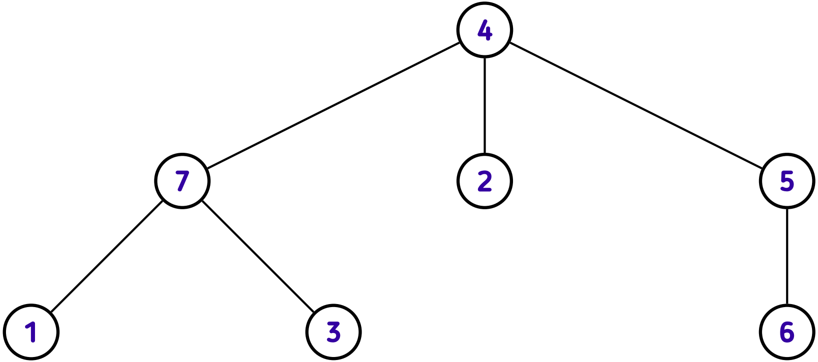
Seja T uma árvore enraizada e considere dois vértices u e v de T. O pico do caminho de u a v é o nó que ocupa o nível baixo em T.



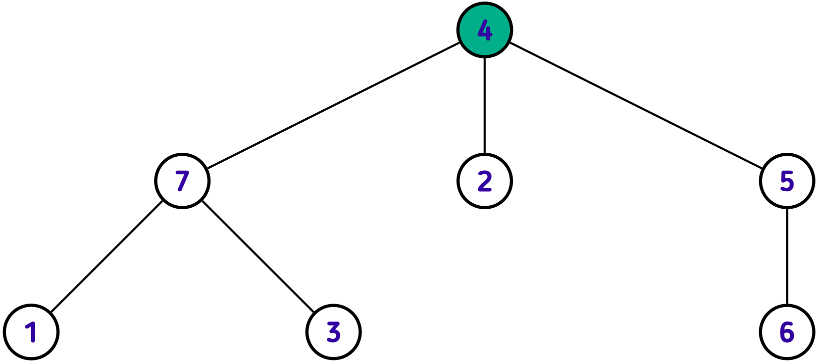
Diâmetro de uma árvore com programação dinâmica

Diâmetro de uma árvore com programação dinâmica

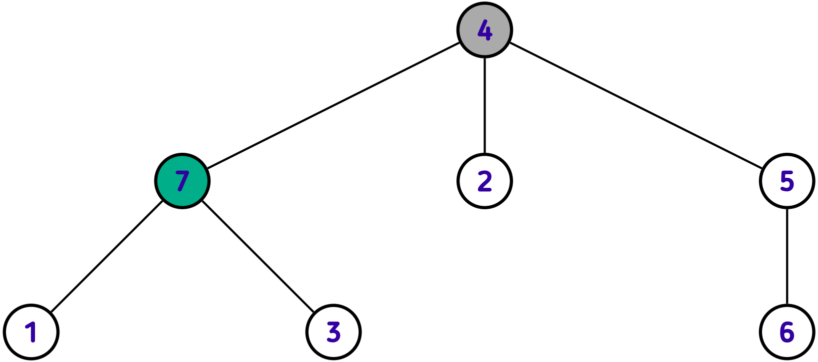
- Em uma árvore enraizada, todo caminho tem um pico
- O pico de um caminho é o nó que ocupa o nível baixo na árvore



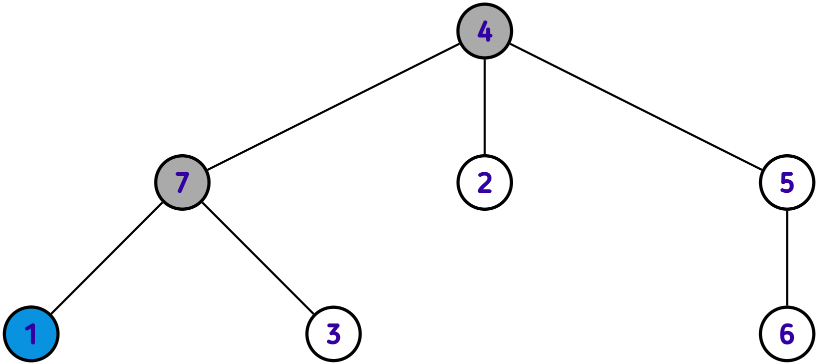
	1	2	3	4	5	6	7
$\text{nodes}[u] =$	-	-	-	-	-	-	-



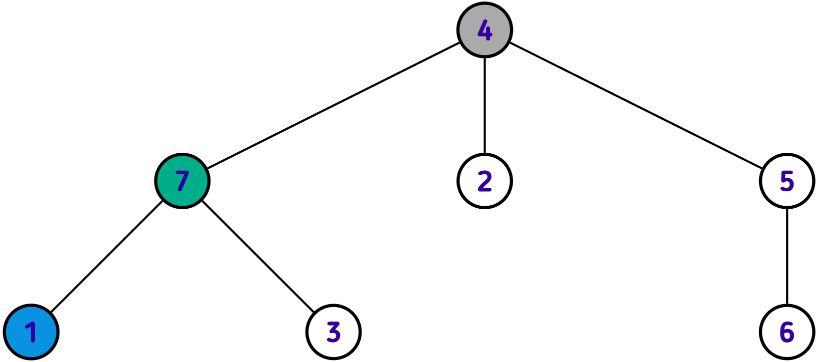
	1	2	3	4	5	6	7
$\text{nodes}[u] =$	-	-	-	1	-	-	-



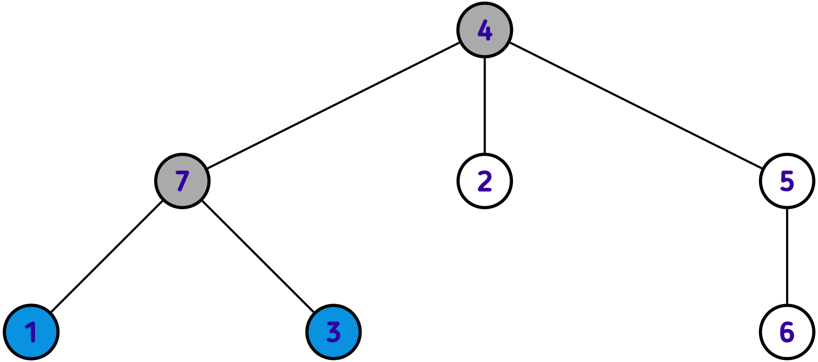
	1	2	3	4	5	6	7
$\text{nodes}[u] =$	-	-	-	1	-	-	1



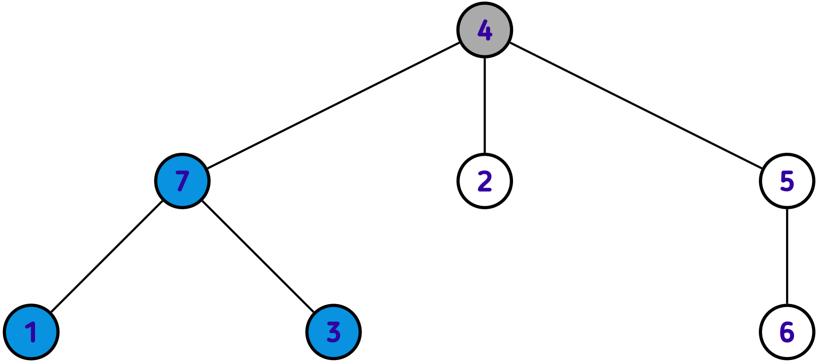
	1	2	3	4	5	6	7
$\text{nodes}[u] =$	1	-	-	1	-	-	1



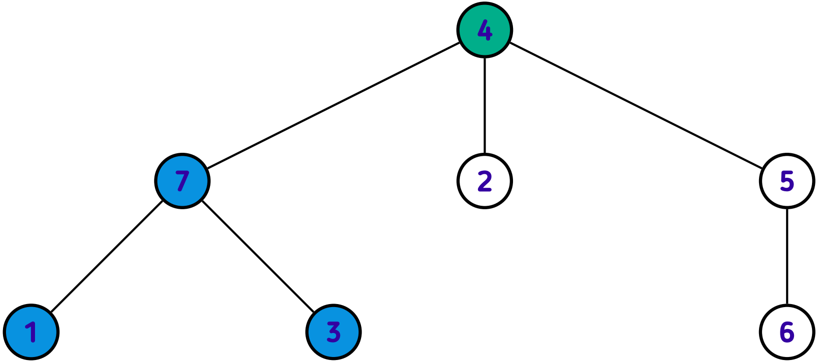
	1	2	3	4	5	6	7
nodes$[u]$ $=$	1	-	-	1	-	-	2



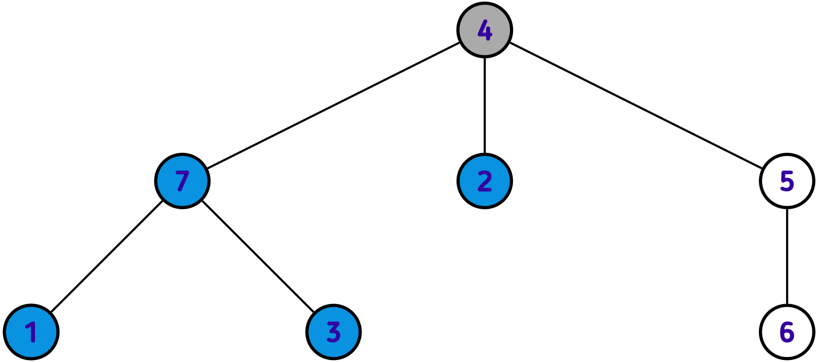
	1	2	3	4	5	6	7
$\mathsf{nodes}[u] =$	1	-	**1**	1	-	-	2



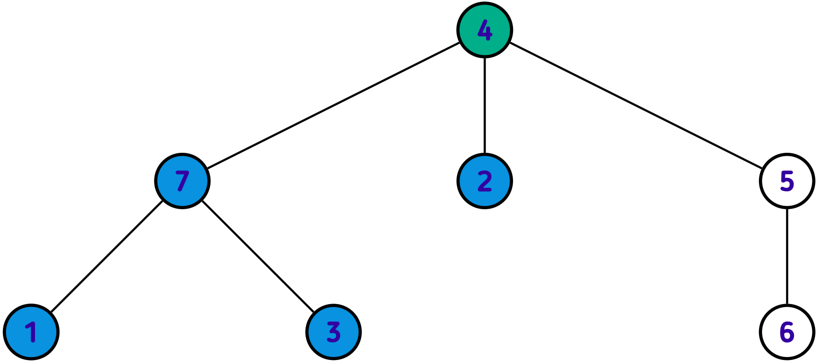
	1	2	3	4	5	6	7
$\mathsf{nodes}[u] =$	1	-	1	1	-	-	3



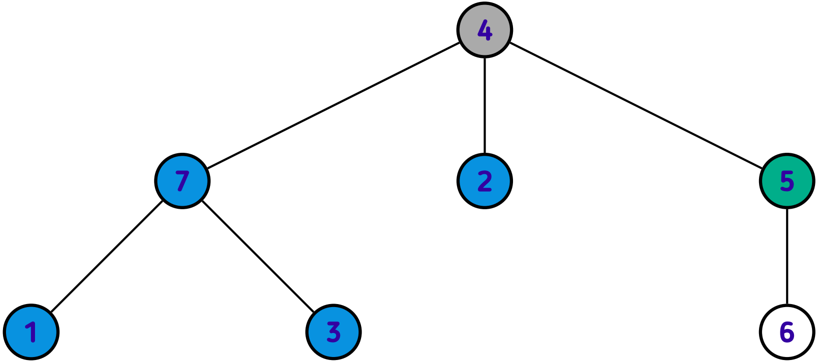
	1	2	3	4	5	6	7
$\text{nodes}[u] =$	1	-	1	4	-	-	3



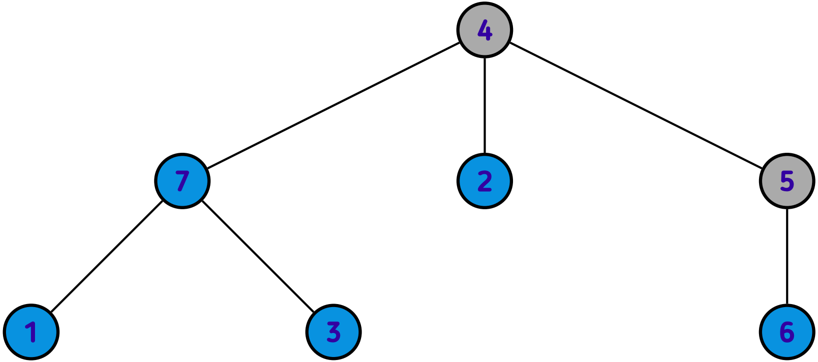
	1	2	3	4	5	6	7
$\mathsf{nodes}[u] =$	1	**1**	1	4	-	-	3



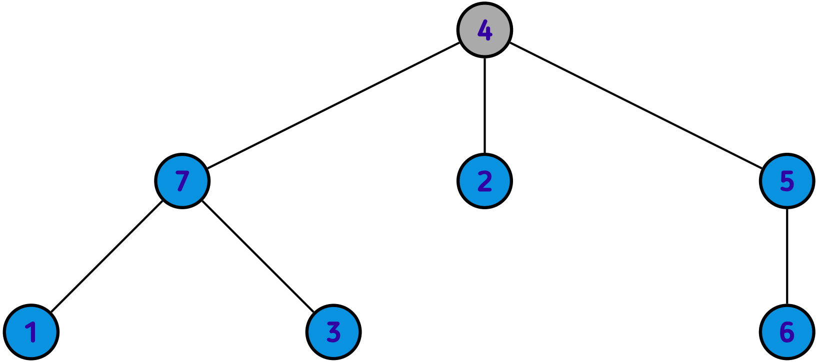
	1	2	3	4	5	6	7
nodes$[u]$ =	1	1	1	**5**	-	-	3



	1	2	3	4	5	6	7
$\text{nodes}[u] =$	1	1	1	5	**1**	-	3

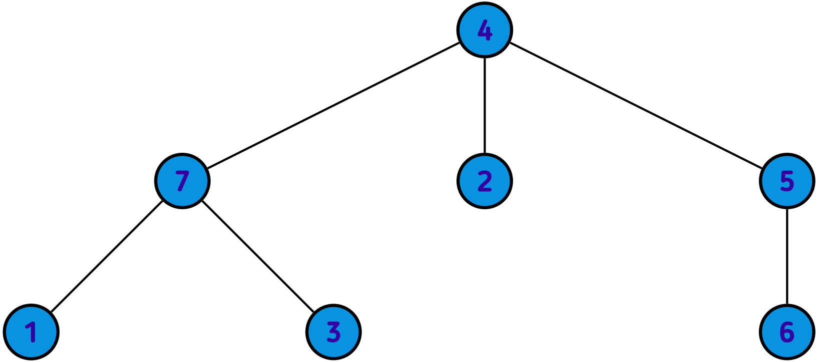


	1	2	3	4	5	6	7
$\text{nodes}[u] =$	1	1	1	5	1	1	3

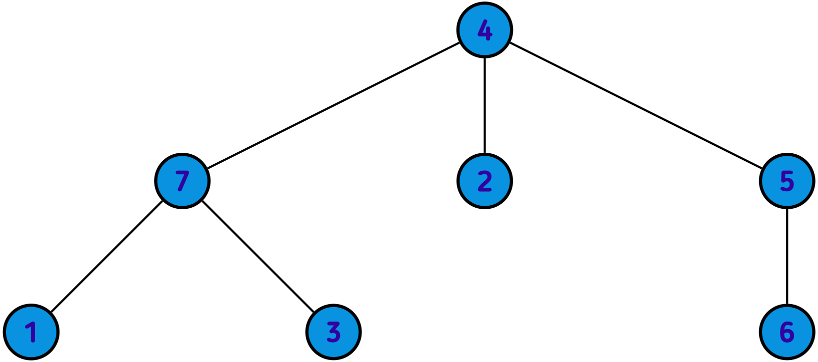


$\mathsf{nodes}[u] =$

1	2	3	4	5	6	7
1	1	1	5	**2**	1	3



	1	2	3	4	5	6	7
nodes$[u] =$	1	1	1	**7**	2	1	3



$\mathsf{nodes}[u] =$

u	1	2	3	4	5	6	7
$\mathsf{nodes}[u]$	1	1	1	7	2	1	3

Maior caminho até uma folha

Maior caminho até uma folha

- ⋆ **Outra aplicação de DFS com DP é o cálculo do tamanho, em arestas, do maior caminho** toLeaf$[u]$ **de** u **até uma folha**

Maior caminho até uma folha

- $\star$ **Outra aplicação de DFS com DP é o cálculo do tamanho, em arestas, do maior caminho** $\text{toLeaf}[u]$ **de** u **até uma folha**

- $\star$ **Se** u **for uma folha, então** $\text{toLeaf}[u] = 0$

Maior caminho até uma folha

- **Outra aplicação de DFS com DP é o cálculo do tamanho, em arestas, do maior caminho** $\textsf{toLeaf}[u]$ **de** u **até uma folha**

- **Se** u **for uma folha, então** $\textsf{toLeaf}[u] = 0$

- **Caso contrário,**

$$\textsf{toLeaf}[u] = 1 + \max_{v \in \textsf{adj[u]}} \{\ \textsf{toLeaf}[v]\ \}$$

Maior caminho até uma folha

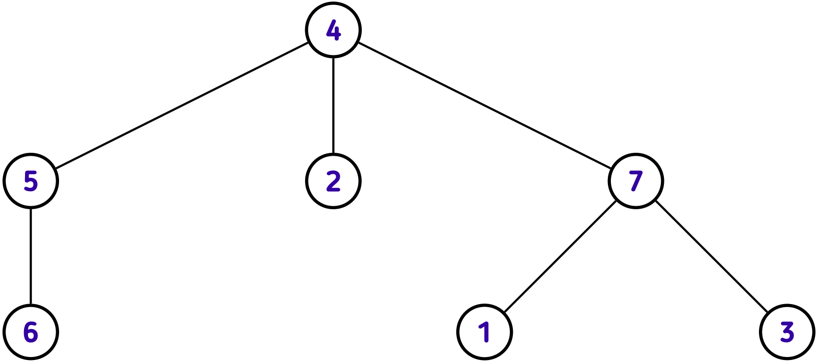
- $\star$ **Outra aplicação de DFS com DP é o cálculo do tamanho, em arestas, do maior caminho** $\textsf{toLeaf}[u]$ **de** u **até uma folha**

- $\star$ **Se** u **for uma folha, então** $\textsf{toLeaf}[u] = 0$

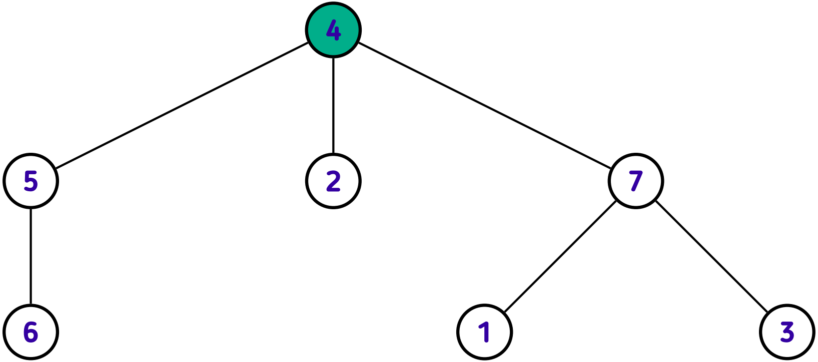
- $\star$ **Caso contrário,**

$$\textsf{toLeaf}[u] = 1 + \max_{v \in \textsf{adj[u]}} \{\ \textsf{toLeaf}[v]\ \}$$

- $\star$ **Este algoritmo pode ser adaptado para computar o tamanho como soma dos pesos das arestas do caminho que vai de** u **até a folha**

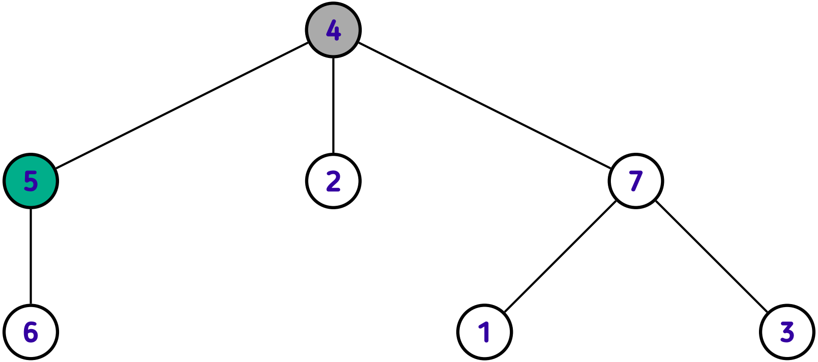


	1	2	3	4	5	6	7
$\text{toLeaf}[u] =$	-	-	-	-	-	-	-



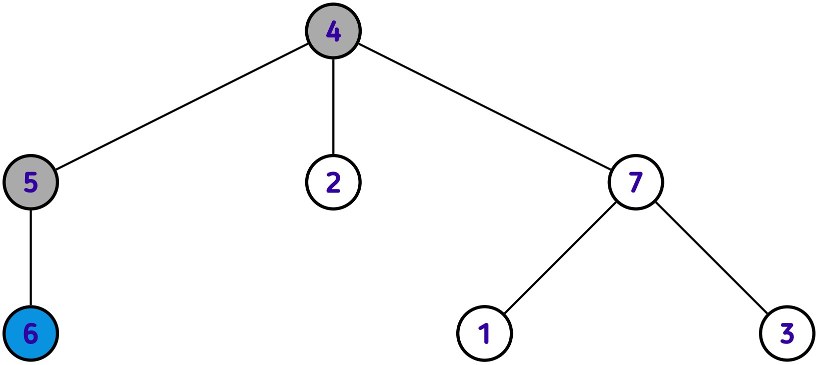
$\text{toLeaf}[u] =$

1	2	3	4	5	6	7
-	-	-	-	-	-	-

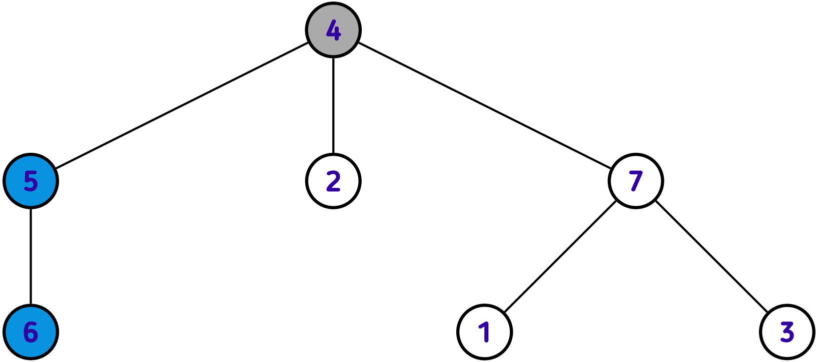


$\text{toLeaf}[u] =$

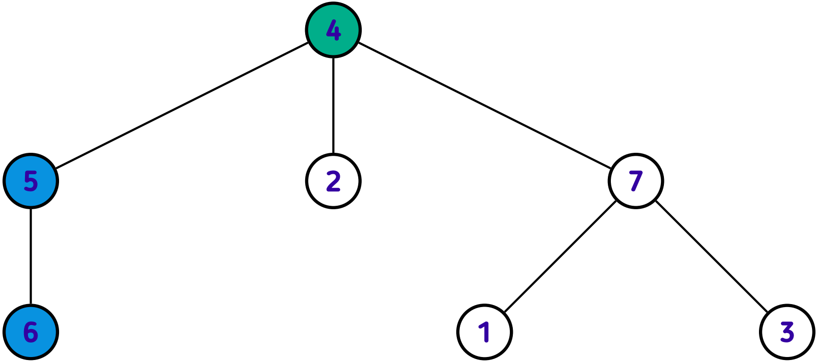
1	2	3	4	5	6	7
-	-	-	-	-	-	-



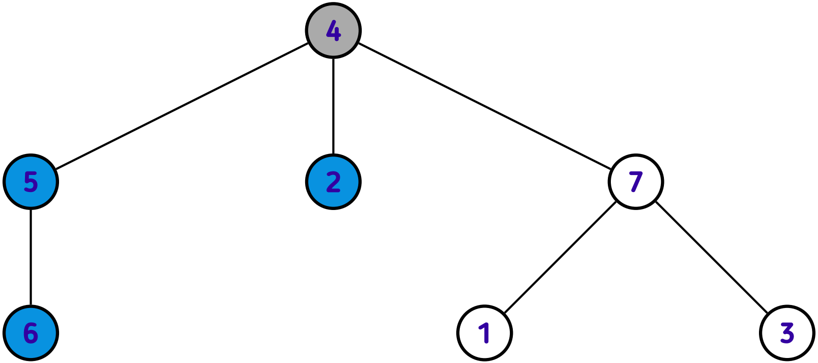
	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	-	-	-	-	0	-



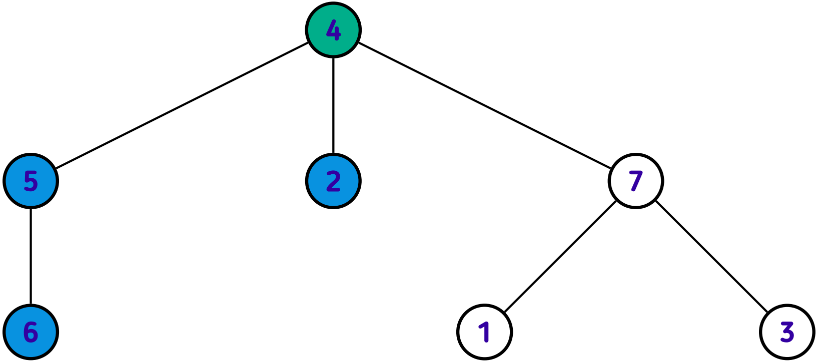
	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	-	-	-	1	0	-



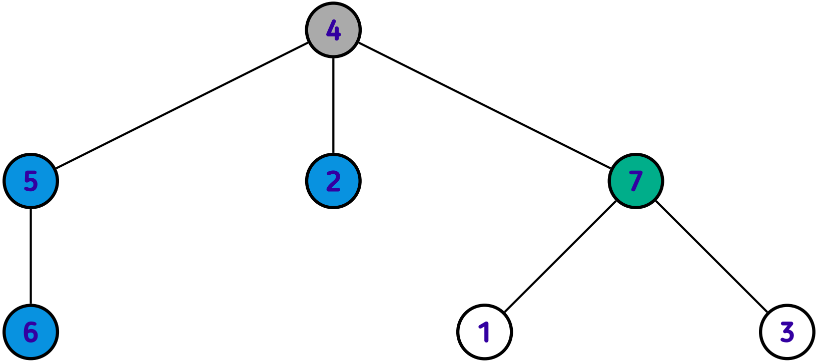
	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	-	-	2	1	0	-



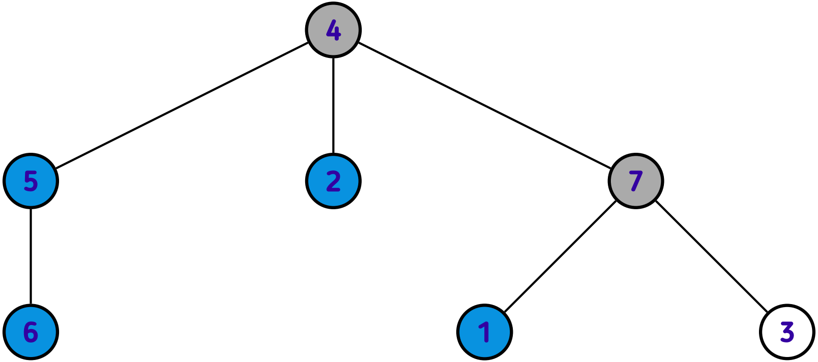
	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	**0**	-	2	1	0	-



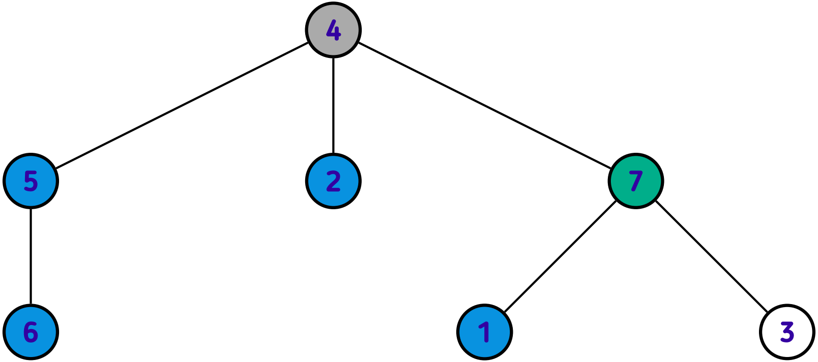
	1	2	3	4	5	6	7
$\text{toLeaf}[u] =$	-	0	-	2	1	0	-



	1	2	3	4	5	6	7
toLeaf$[u]$ =	-	0	-	2	1	0	-

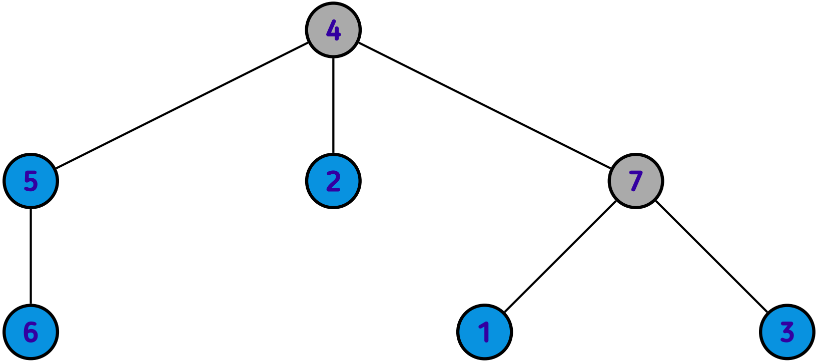


	1	2	3	4	5	6	7
toLeaf$[u]$ =	**0**	0	-	2	1	0	-

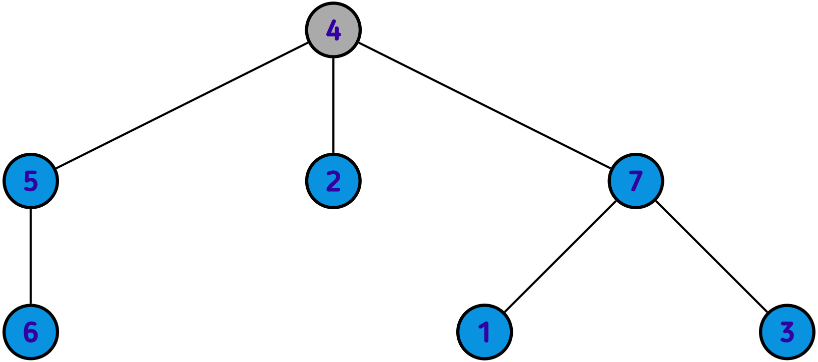


toLeaf$[u] =$

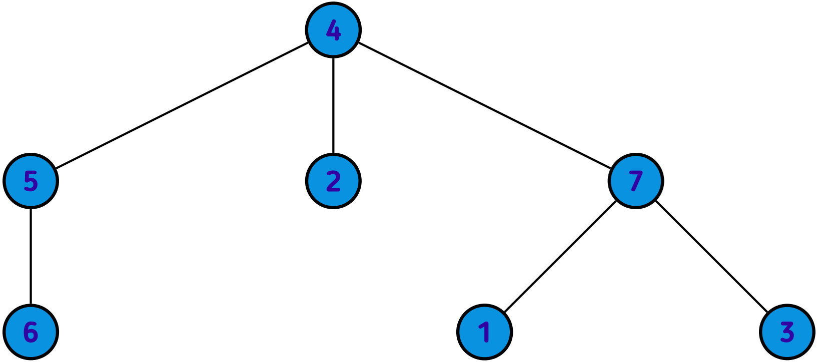
1	2	3	4	5	6	7
0	0	-	2	1	0	**0**



$\text{toLeaf}[u] =$	1	2	3	4	5	6	7
	0	0	**0**	2	1	0	0



	1	2	3	4	5	6	7
toLeaf$[u]$ =	0	0	0	2	1	0	**1**



	1	2	3	4	5	6	7
$\mathsf{toLeaf}[u] =$	0	0	0	2	1	0	1

Problemas sugeridos

1. AtCoder Beginner Contest 126 – Problem D: Even Relation
2. Codeforces Beta Round #87 (Div. 1 Only) – Problem A: Party
3. Codeforces Round #660 (Div. 2) – Problem C: Uncle Bogdan and Country Happiness
4. OJ 10459 – The Tree Root

Referências

1. **DROZDEK**, **Adam**. *Algoritmos e Estruturas de Dados em C++*, **2002.**
2. **HALIM**, **Felix**; **HALIM**, **Steve**. *Competitive Programming 3*, **2010.**
3. **LAAKSONEN**, **Antti**. *Competitive Programmer's Handbook*, **2018.**
4. **SKIENA**, **Steven**; **REVILLA**, **Miguel**. *Programming Challenges*, **2003.**
5. **Wikipédia.** *Tree (graph theory)*, **acesso em 06/08/2021.**