Grafos

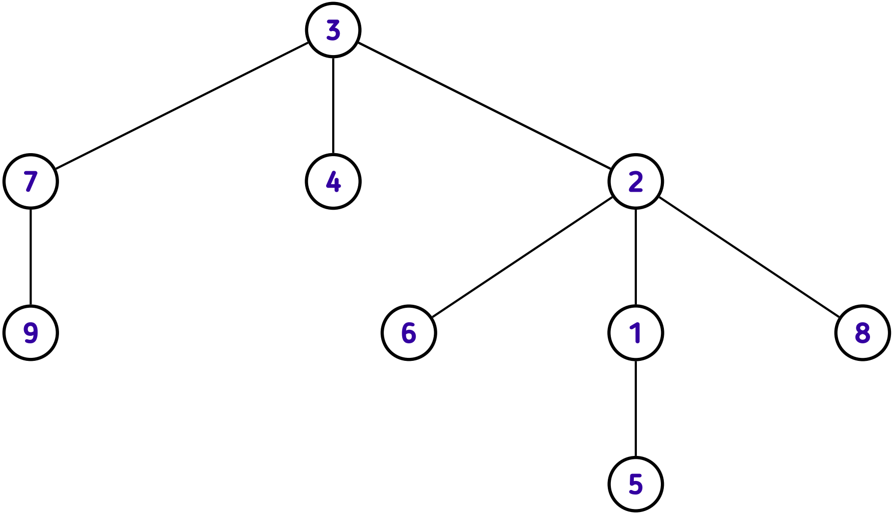
Ancestrais

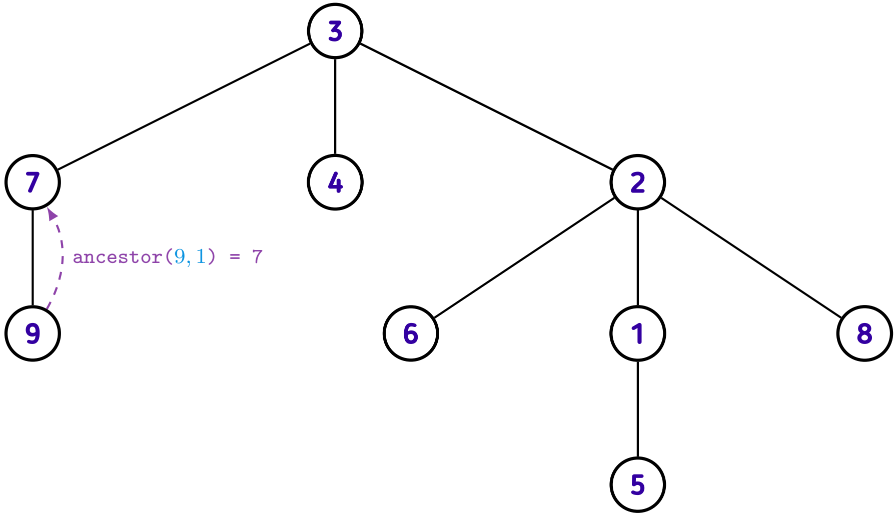
Prof. Edson Alves

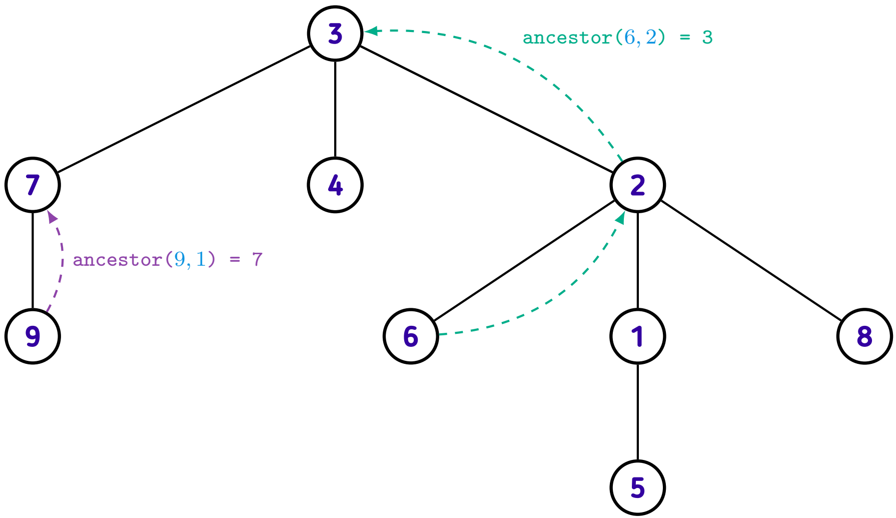
Faculdade UnB Gama

k-ésimo ancestral

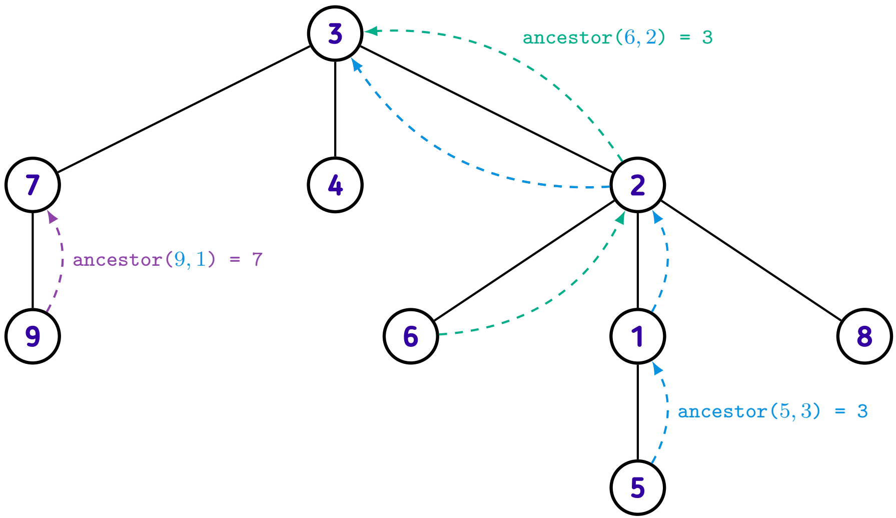
Seja T uma árvore enraizada, u um vértice de T e k um inteiro positivo. O k-ésimo ancestral de u é o nó v que encerra o caminho que parte de u e segue k níveis, em direção à raiz. Notação: v = $\texttt{ancestor}(u, k)$.


3
7
4
2
9
6
1
8
5


3
7
4
2
9
6
1
8
5
ancestor(9, 1) = 7



ancestor(6, 2) = 3
ancestor(9, 1) = 7
3
7
4
2
9
6
1
8
5


ancestor(6, 2) = 3
ancestor(9, 1) = 7
ancestor(5, 3) = 3
3
7
4
2
9
6
1
8
5

Identificação do k-ésimo ancestral em $O(\log N)$

Identificação do k-eśimo ancestral em $O(\log N)$

$\star$ É possível identificar o k-ésimo ancestral em $O(\log N)$, onde N é o número vértices da árvore, por meio de um algoritmo de programação dinâmica

Identificação do k-eśimo ancestral em $O(\log N)$

- É possível identificar o k-ésimo ancestral em $O(\log N)$, onde N é o número vértices da árvore, por meio de um algoritmo de programação dinâmica

- Seja $p(u)$ o ancestral direto de u na árvore enraizada

Identificação do k-eśimo ancestral em $O(\log N)$

- ⋆ **É possível identificar o k-ésimo ancestral em $O(\log N)$, onde N é o número vértices da árvore, por meio de um algoritmo de programação dinâmica**

- ⋆ **Seja $p(u)$ o ancestral direto de u na árvore enraizada**

- ⋆ **O caso base ocorre com $k = 2^0 = 1$:** `ancestor(`$u, 1$`)` **=** $p(u)$

Identificação do k-eśimo ancestral em $O(\log N)$

- É possível identificar o k-ésimo ancestral em $O(\log N)$, onde N é o número vértices da árvore, por meio de um algoritmo de programação dinâmica

- Seja $p(u)$ o ancestral direto de u na árvore enraizada

- O caso base ocorre com $k = 2^0 = 1$: `ancestor(`$u, 1$`) = `$p(u)$

- A transição é dada por:

$$\texttt{ancestor}(u, 2^k) = \texttt{ancestor}(\texttt{ancestor}(u, 2^{k-1}), 2^{k-1})$$

Identificação do k-ésimo ancestral em $O(\log N)$

Identificação do k-eśimo ancestral em $O(\log N)$

- Seja k um inteiro positivo

Identificação do k-eśimo ancestral em $O(\log N)$

- ⋆ **Seja k um inteiro positivo**
- ⋆ **É possível escrever k como a soma de potências distintas de 2:**

$$k = 2^{\alpha} + 2^{\beta} + \ldots + 2^{\omega}$$

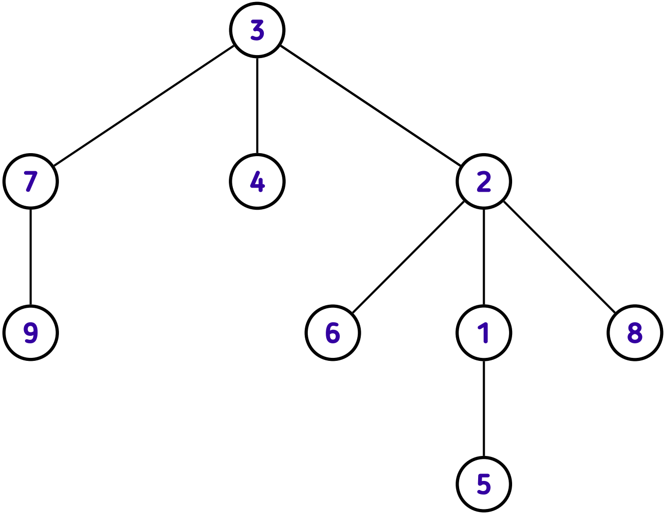
Identificação do k-eśimo ancestral em $O(\log N)$

- $\star$ **Seja k um inteiro positivo**
- $\star$ **É possível escrever k como a soma de potências distintas de 2:**

$$k = 2^\alpha + 2^\beta + \ldots + 2^\omega$$

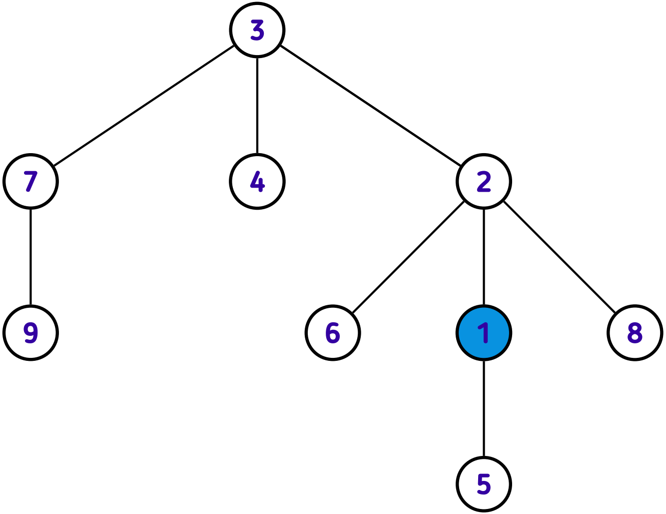
- $\star$ **Deste modo,**

$$\texttt{ancestor}(u, k) = \texttt{ancestor}(\texttt{ancestor}(\texttt{ancestor}(\texttt{ancestor}(u, 2^\alpha), 2^\beta), \ldots), 2^\omega)$$



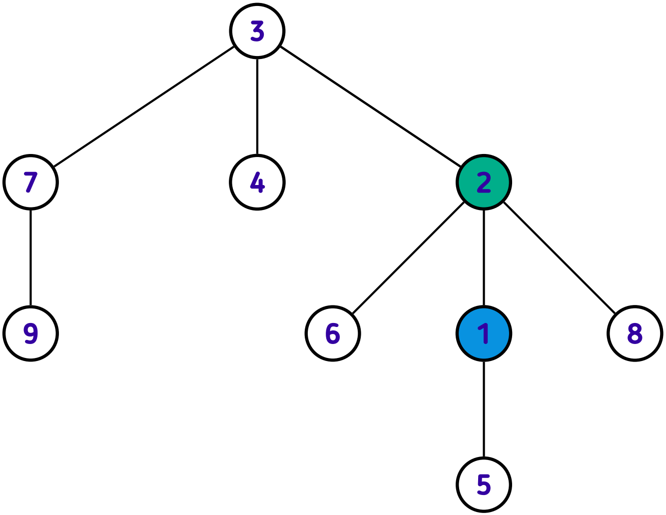
	2^0	2^1	2^2
1			
2			
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



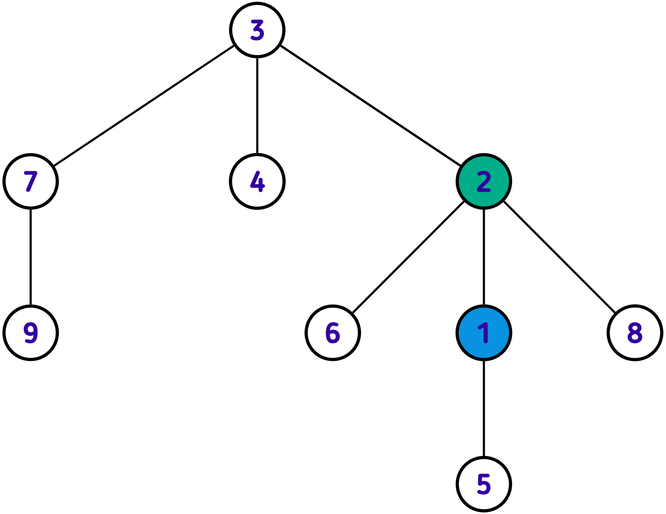
	2^0	2^1	2^2
1			
2			
3			
4			
5			
6			
7			
8			
9			

$\mathsf{succ}(u, 2^k)$



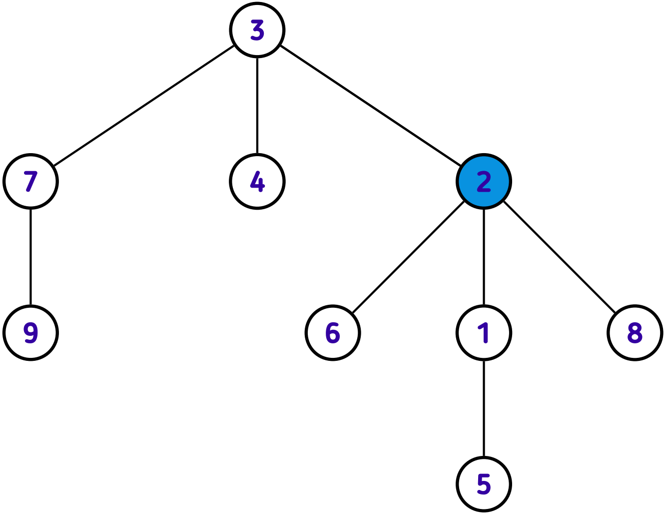
	2^0	2^1	2^2
1			
2			
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



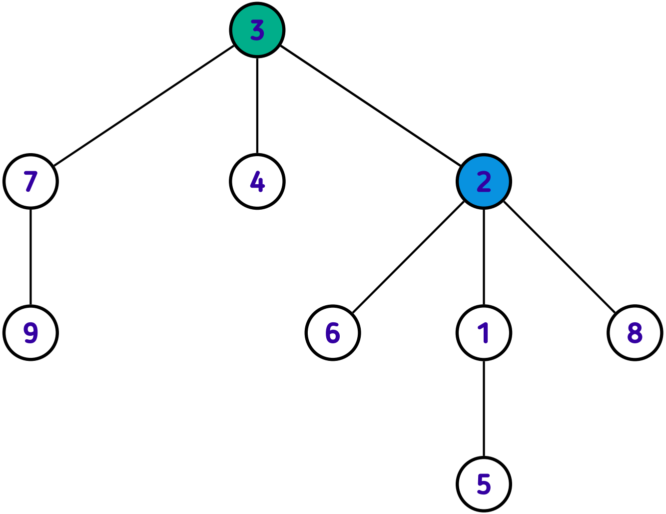
	2^0	2^1	2^2
1	2		
2			
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



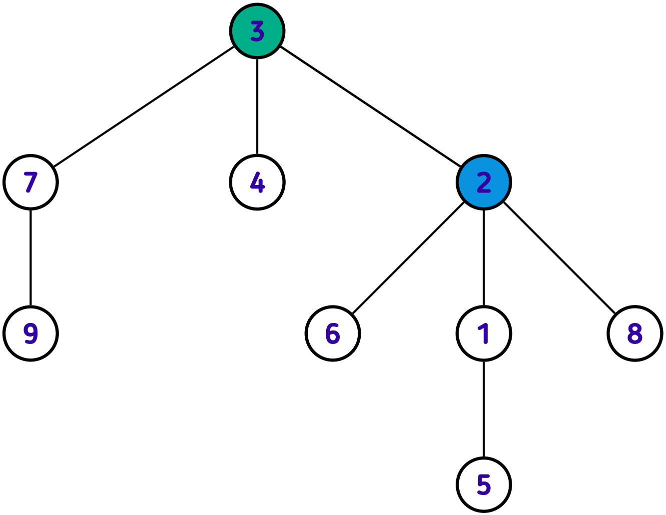
	2^0	2^1	2^2
1	2		
2			
3			
4			
5			
6			
7			
8			
9			

$\mathsf{succ}(u, 2^k)$



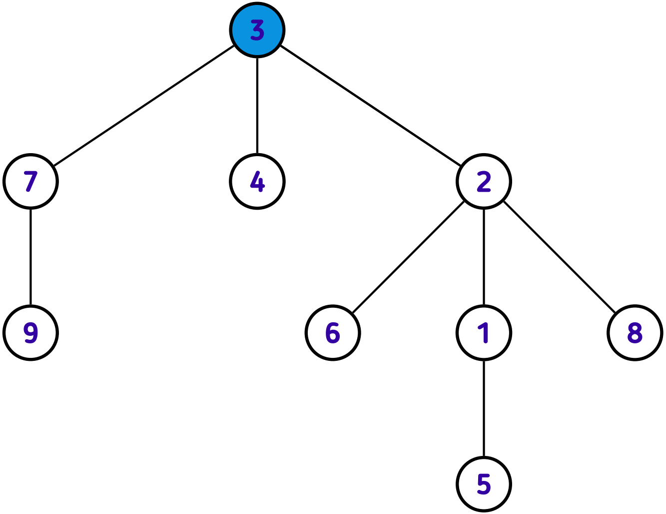
	2^0	2^1	2^2
1	2		
2			
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



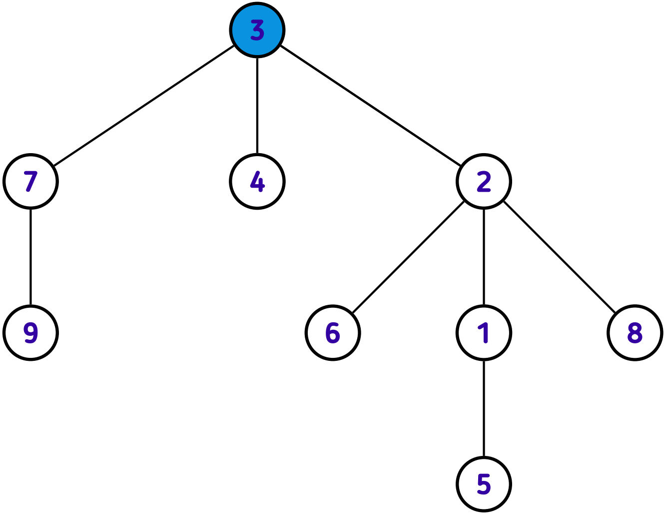
	2^0	2^1	2^2
1	2		
2	3		
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



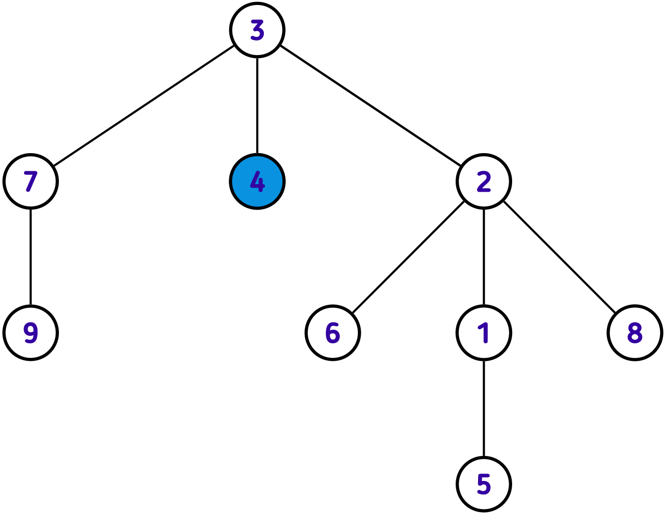
	2^0	2^1	2^2
1	2		
2	3		
3			
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



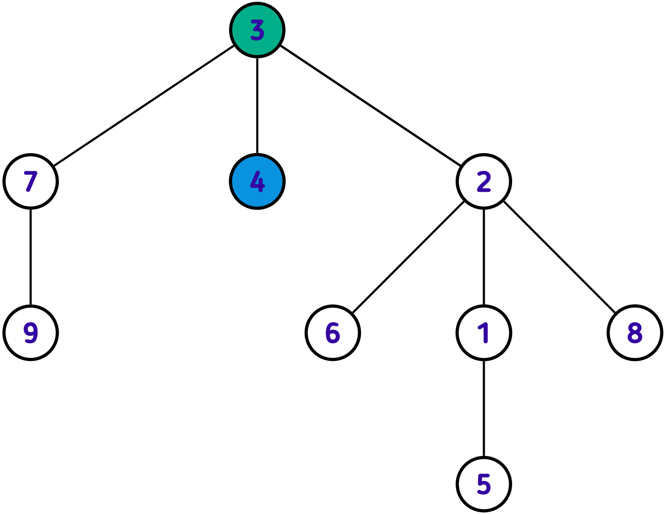
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



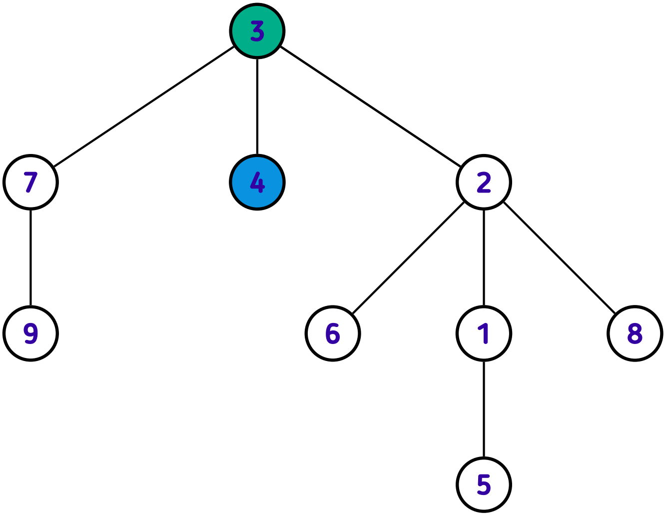
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



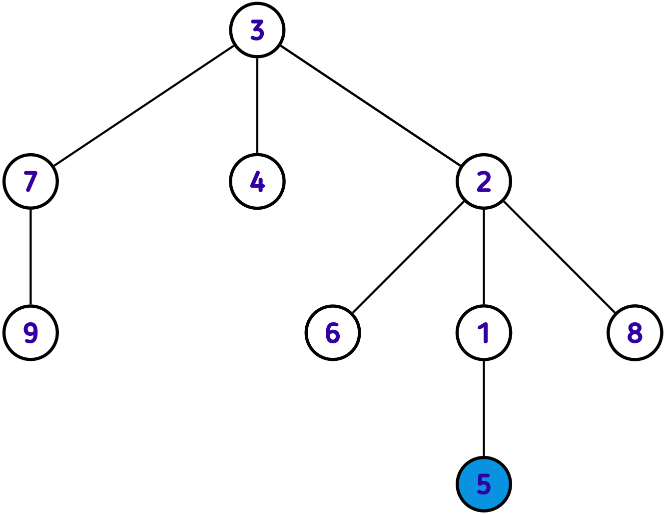
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4			
5			
6			
7			
8			
9			

succ$(u, 2^k)$



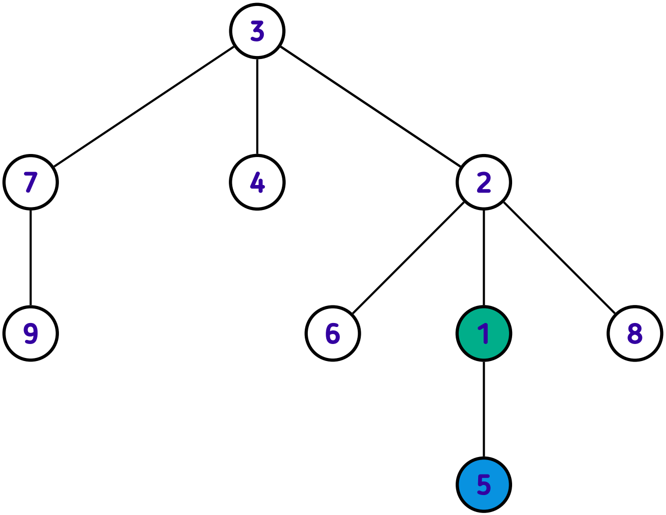
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5			
6			
7			
8			
9			

succ$(u, 2^k)$



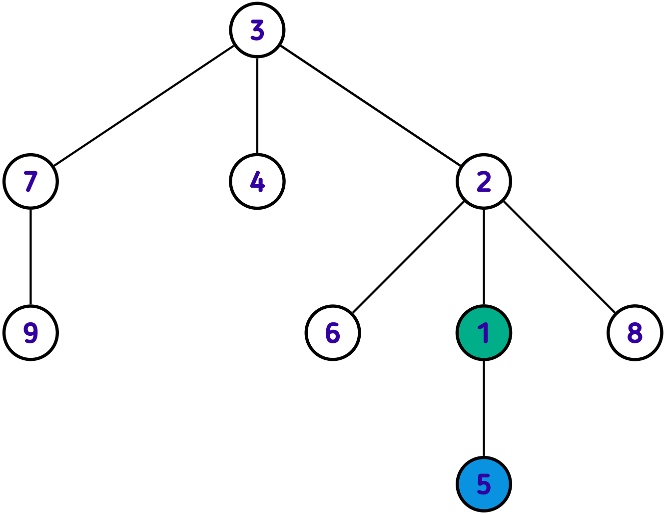
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5			
6			
7			
8			
9			

succ$(u, 2^k)$



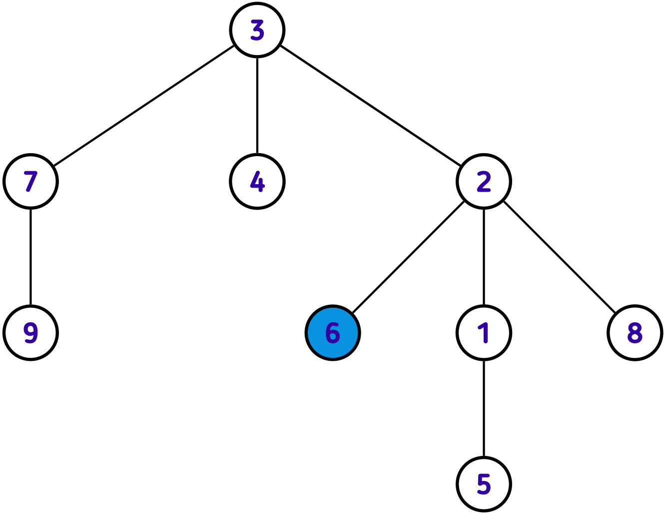
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5			
6			
7			
8			
9			

succ$(u, 2^k)$



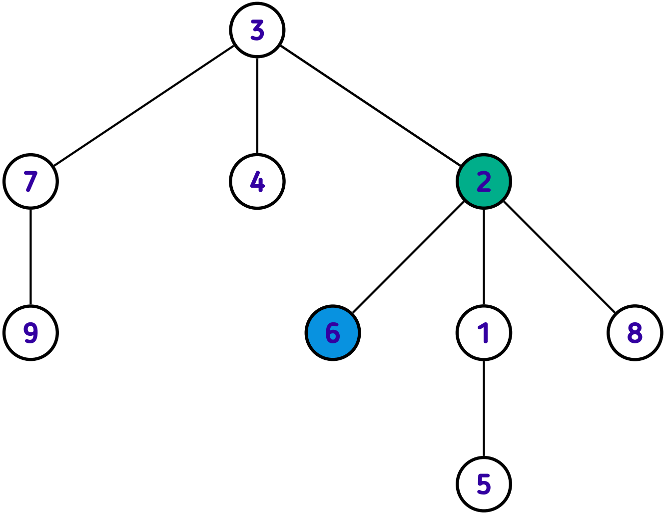
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6			
7			
8			
9			

succ$(u, 2^k)$



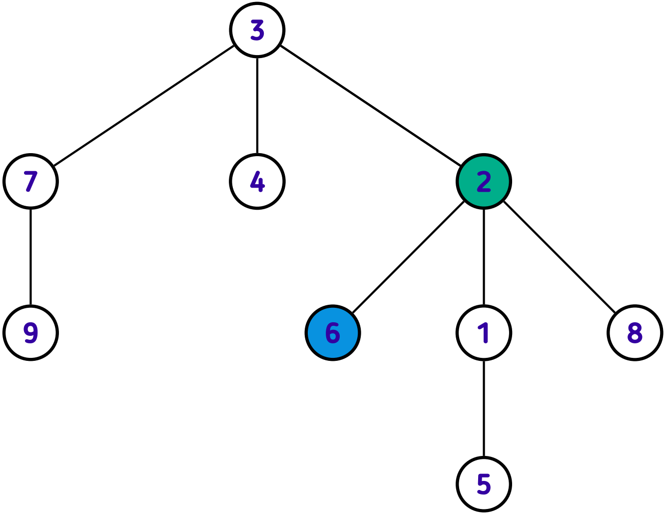
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6			
7			
8			
9			

succ$(u, 2^k)$



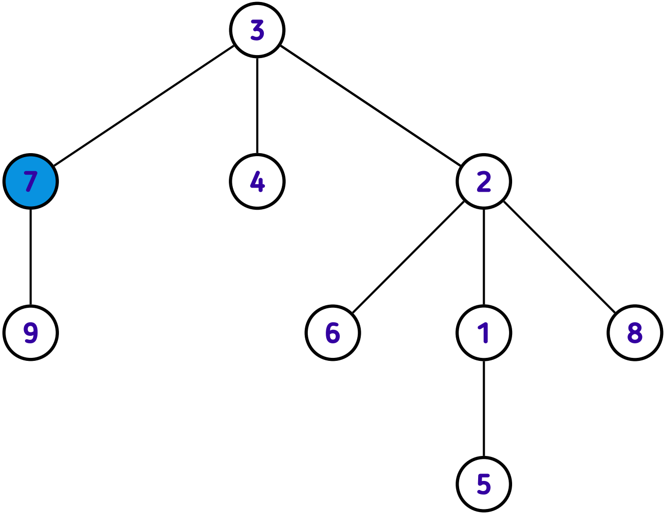
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6			
7			
8			
9			

succ$(u, 2^k)$



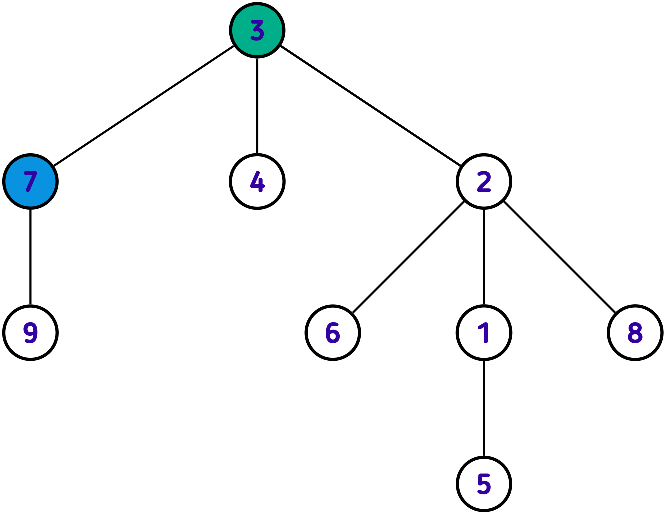
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7			
8			
9			

succ$(u, 2^k)$



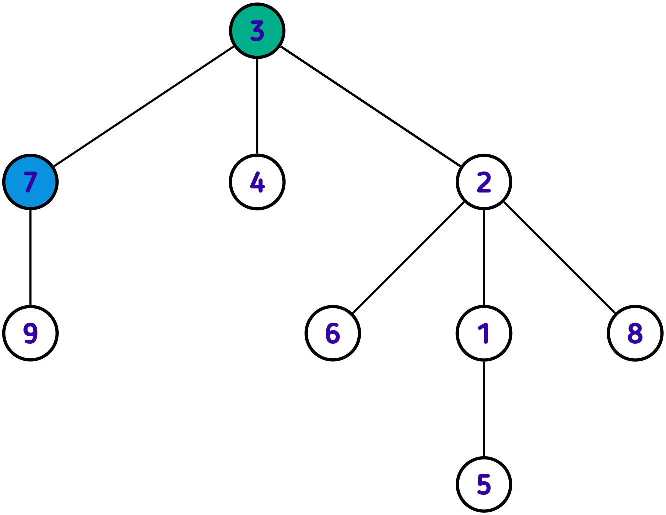
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7			
8			
9			

succ$(u, 2^k)$



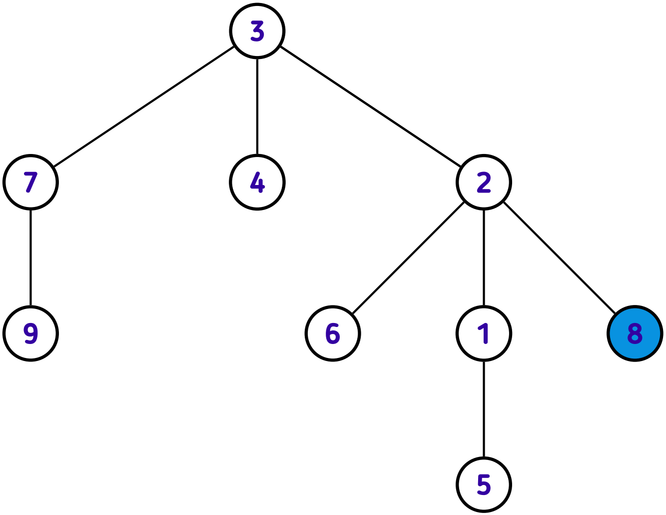
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7			
8			
9			

succ$(u, 2^k)$



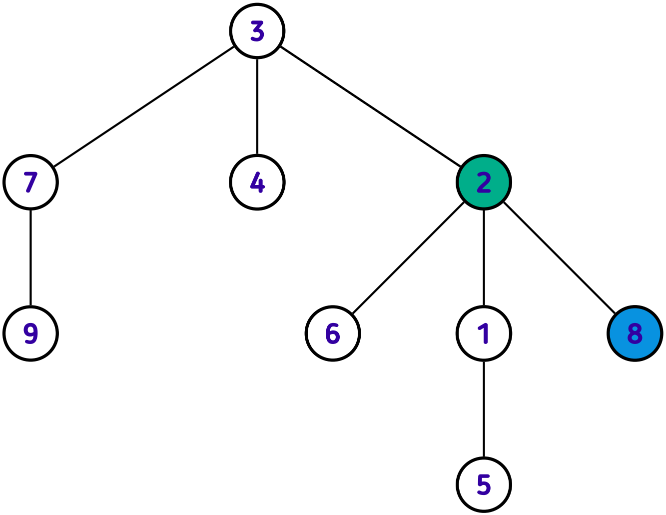
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8			
9			

succ$(u, 2^k)$



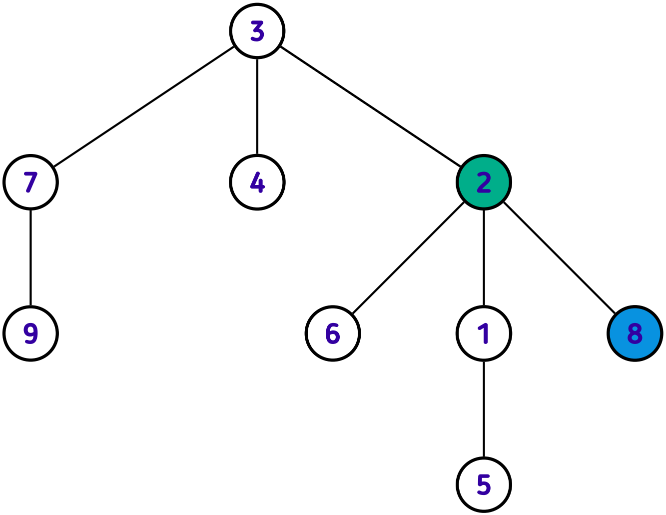
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8			
9			

succ$(u, 2^k)$



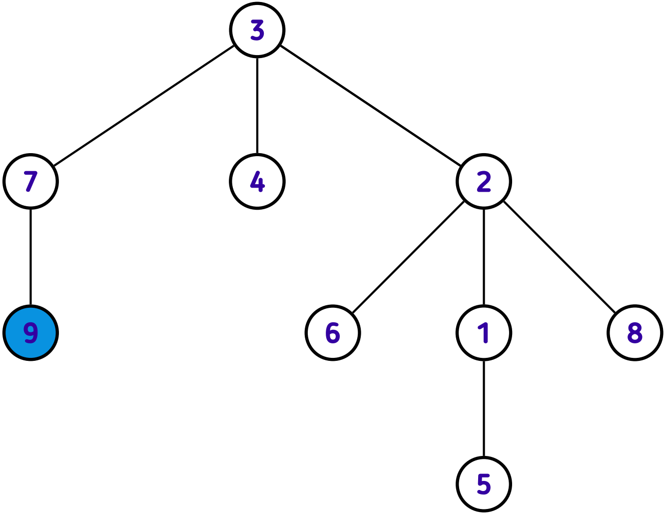
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8			
9			

succ$(u, 2^k)$



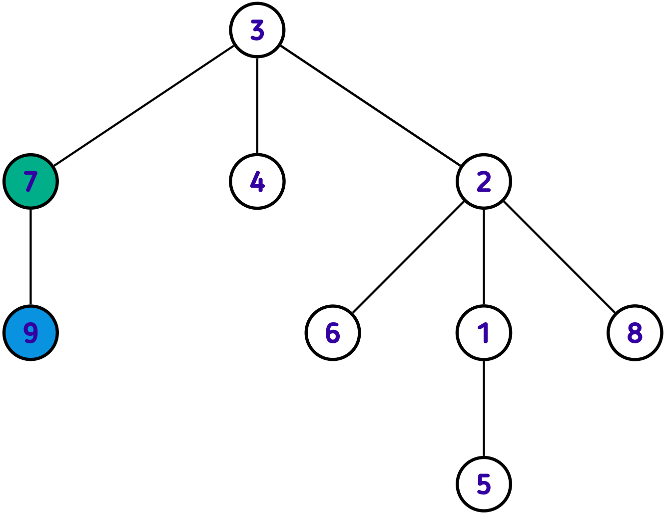
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9			

succ$(u, 2^k)$



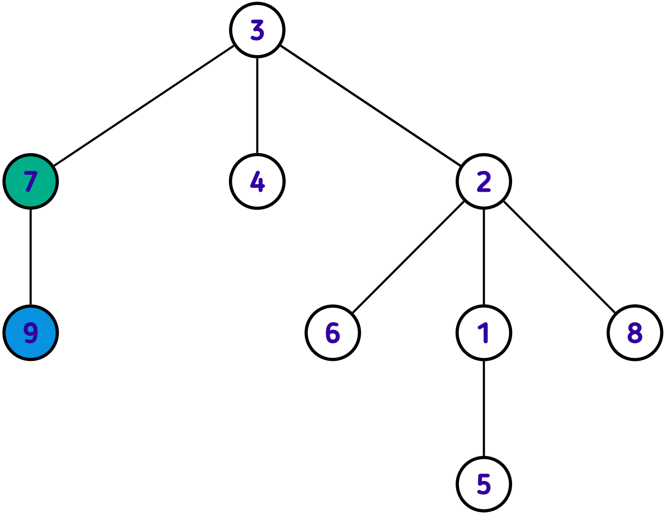
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9			

$\mathsf{succ}(u, 2^k)$



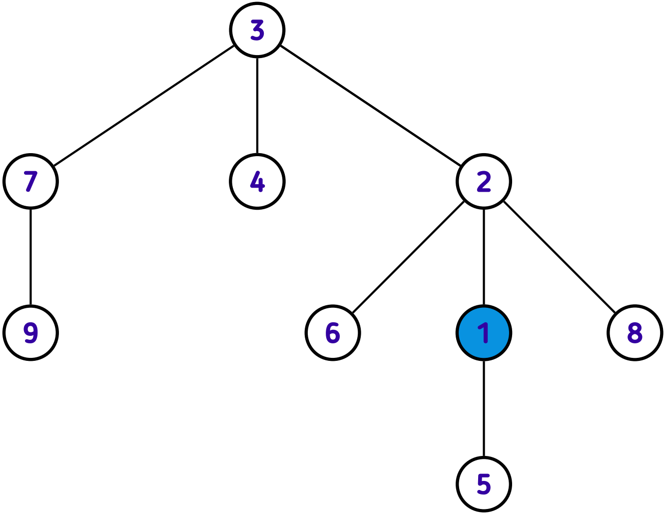
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9			

succ$(u, 2^k)$



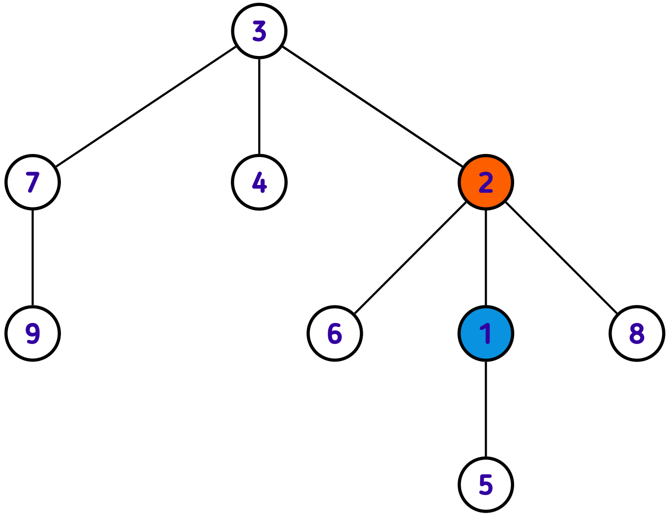
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



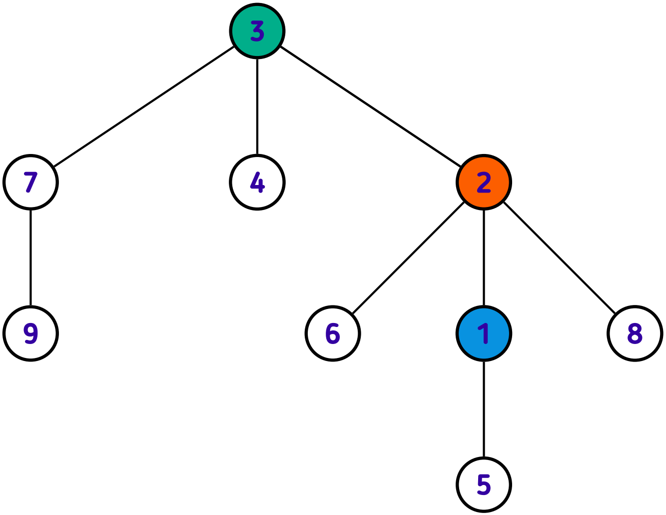
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



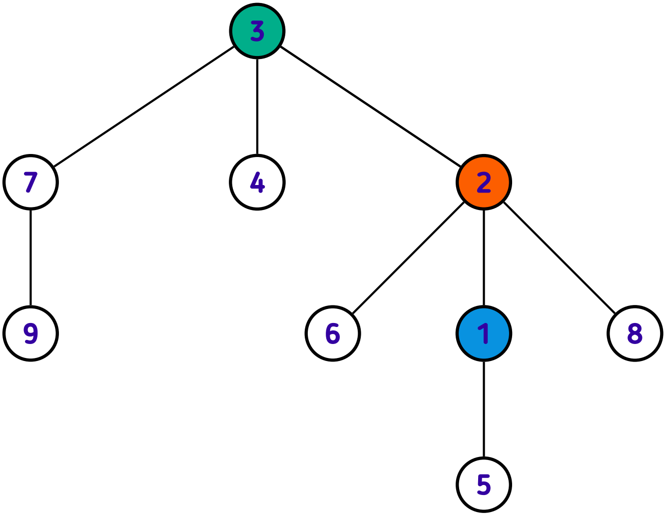
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



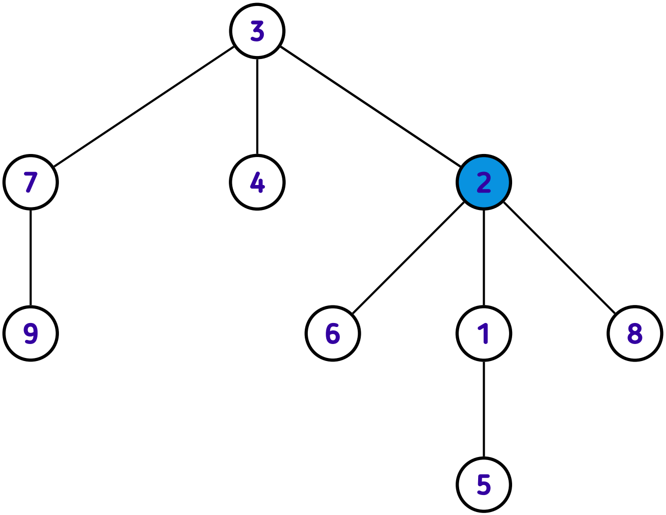
	2^0	2^1	2^2
1	2		
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



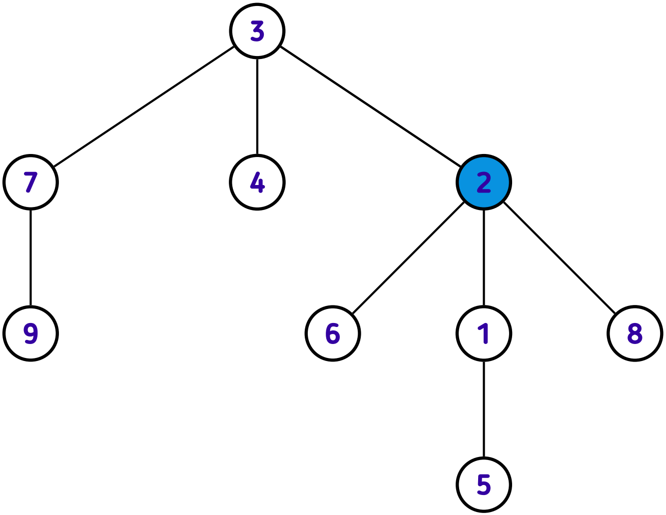
	2^0	2^1	2^2
1	2	3	
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



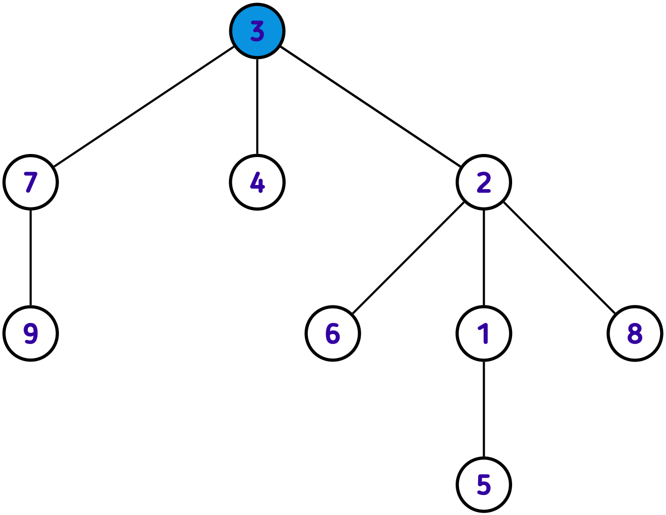
	2^0	2^1	2^2
1	2	3	
2	3		
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



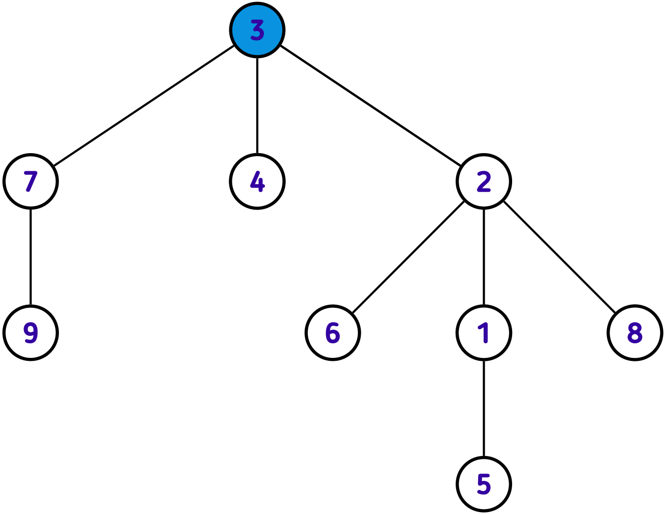
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



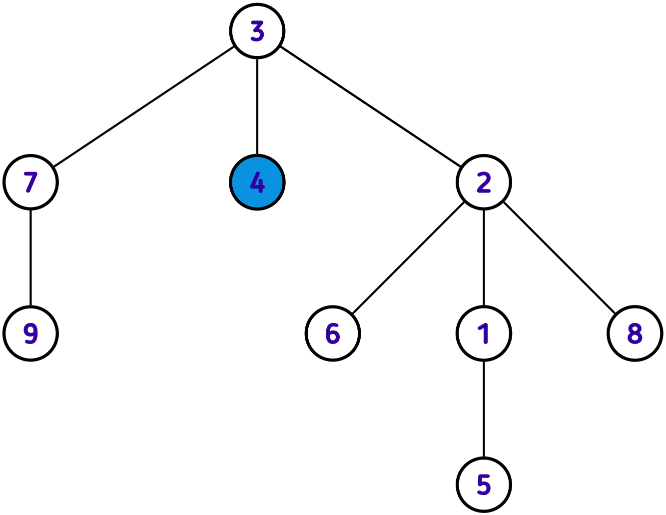
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-		
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



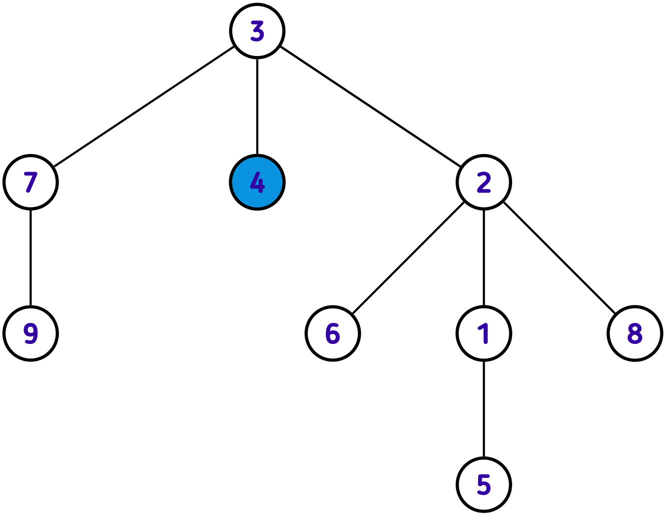
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



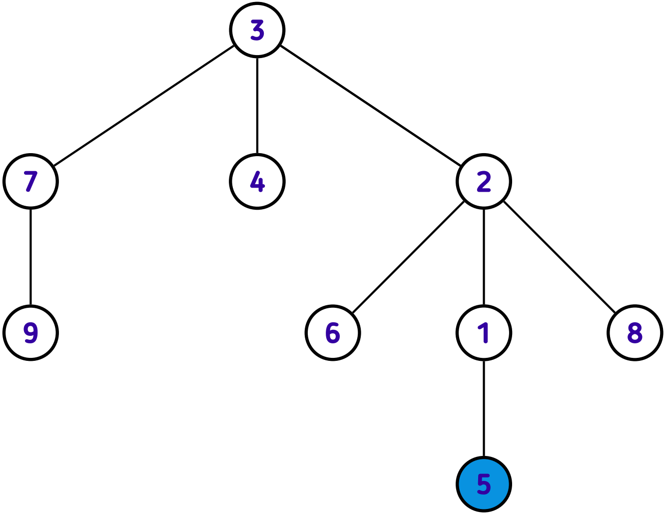
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3		
5	1		
6	2		
7	3		
8	2		
9	7		

$\mathrm{succ}(u, 2^k)$



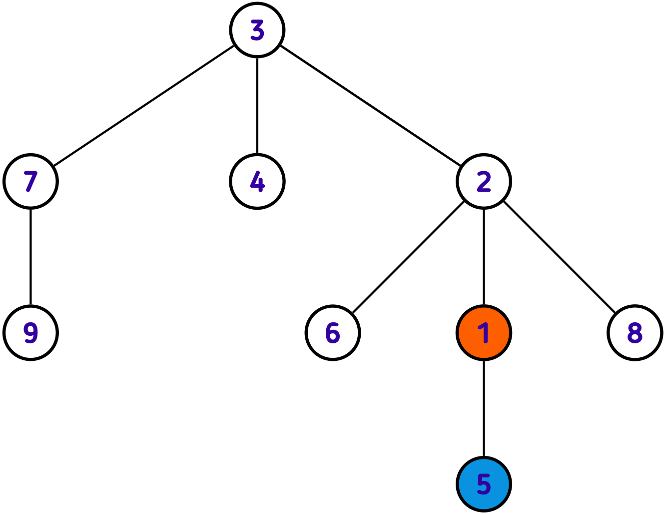
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



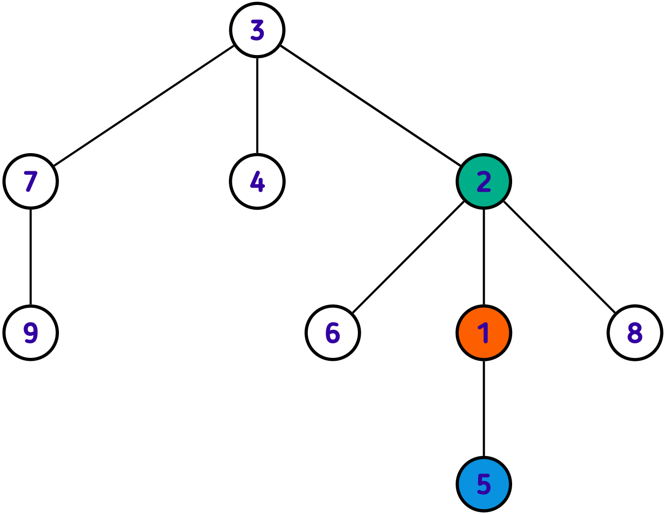
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



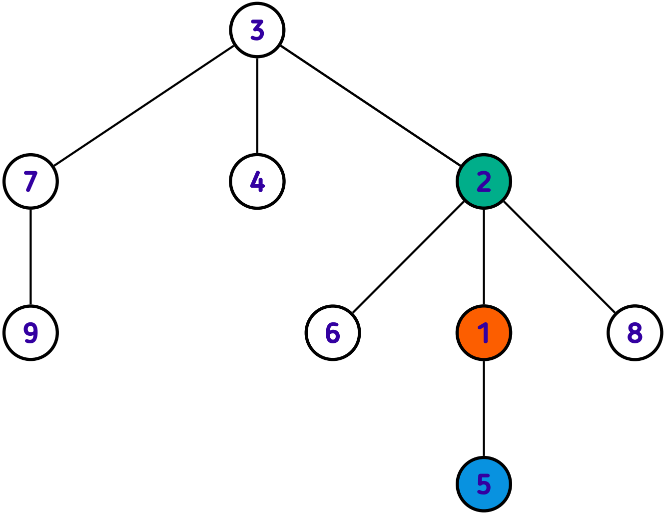
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1		
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



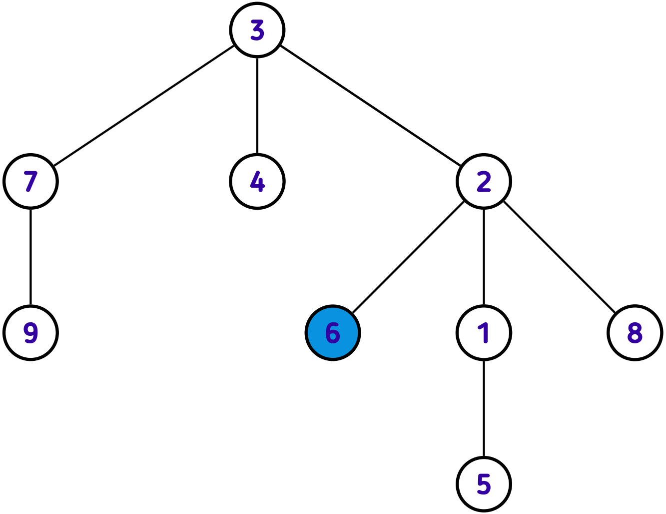
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1		
6	2		
7	3		
8	2		
9	7		

$\text{succ}(u, 2^k)$



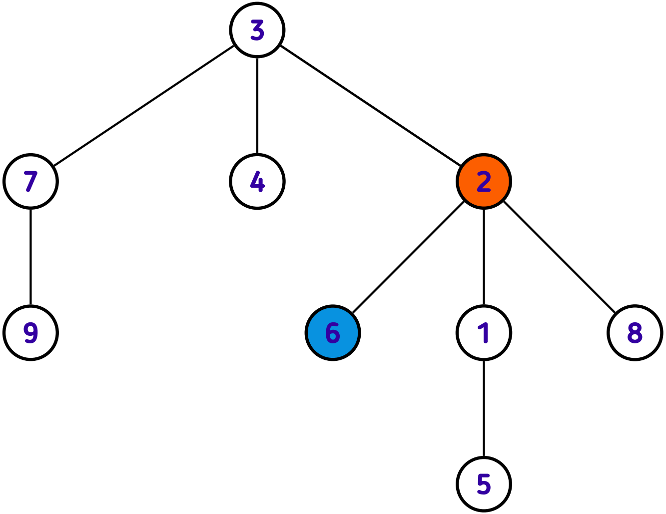
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



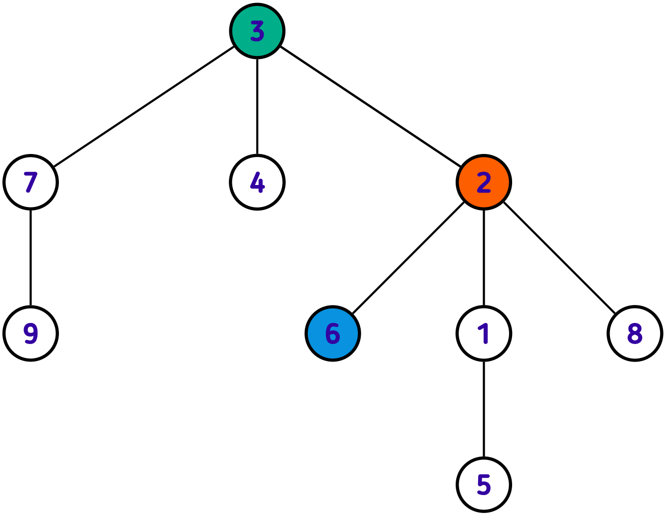
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



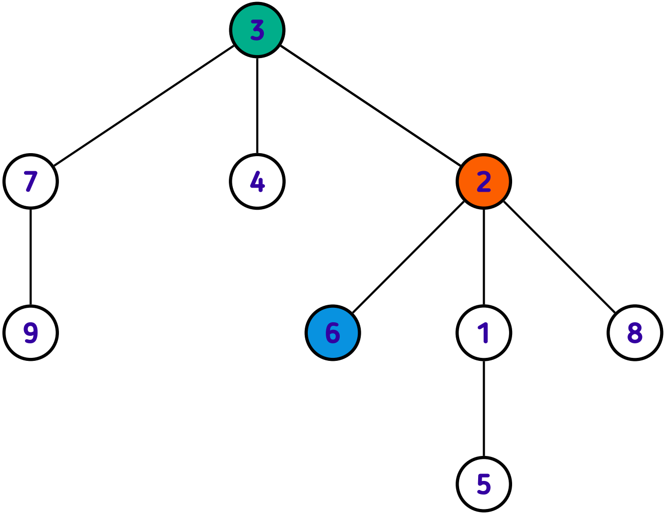
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



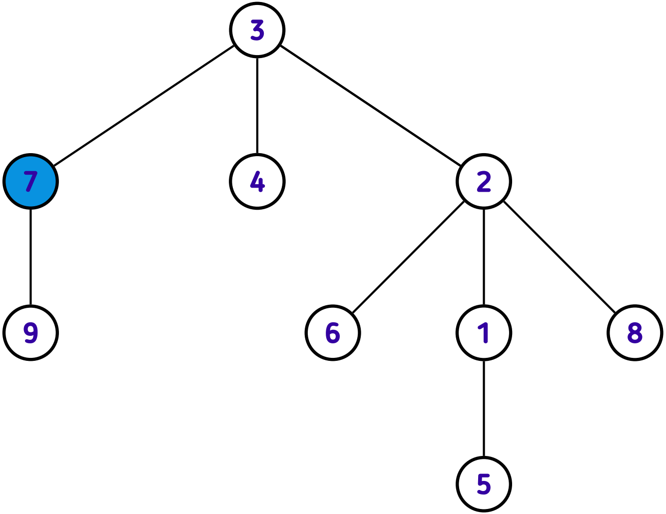
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2		
7	3		
8	2		
9	7		

succ$(u, 2^k)$



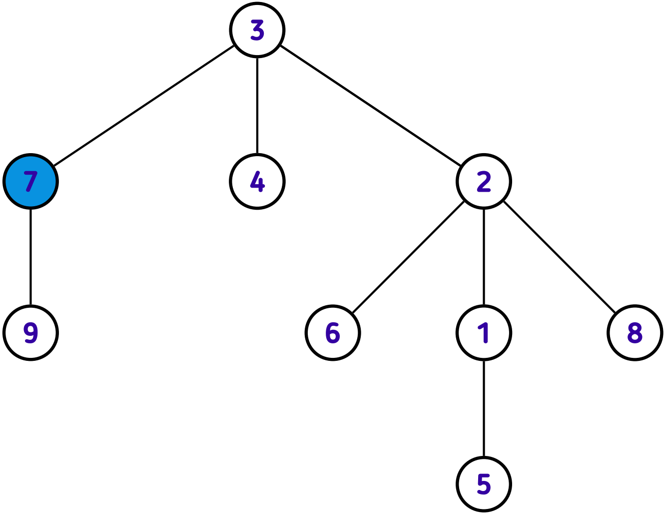
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3		
8	2		
9	7		

succ$(u, 2^k)$



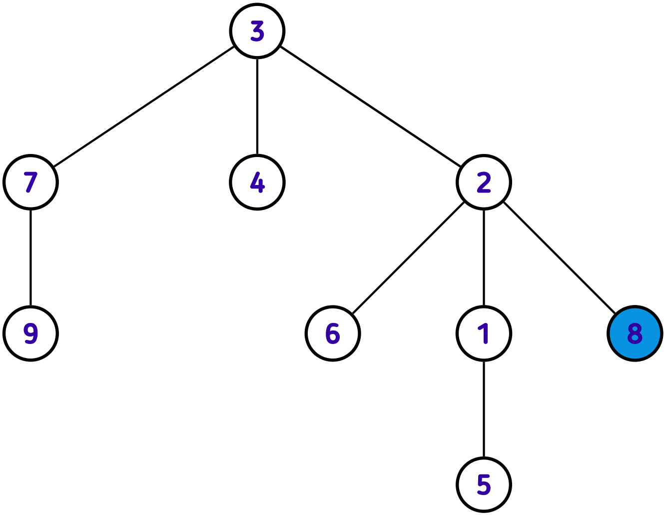
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3		
8	2		
9	7		

succ$(u, 2^k)$



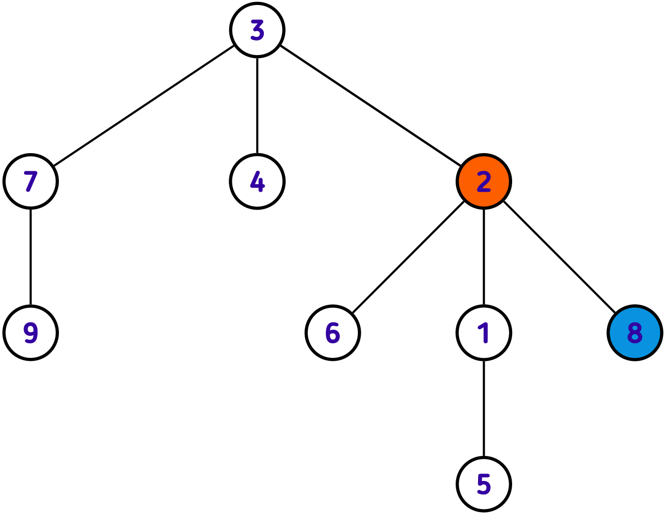
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2		
9	7		

succ$(u, 2^k)$



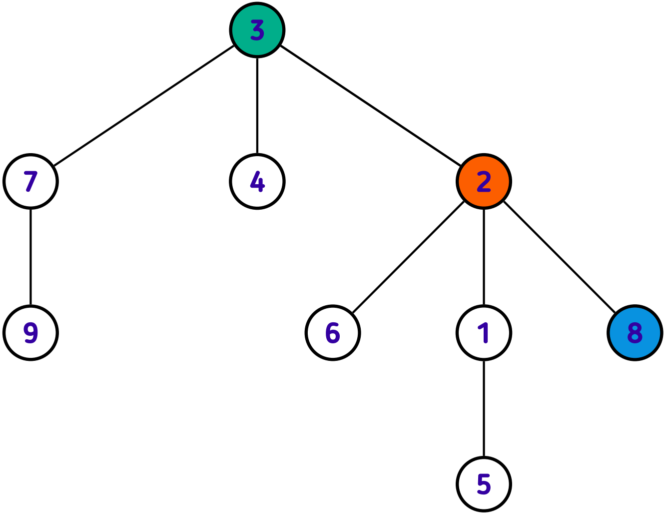
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2		
9	7		

succ$(u, 2^k)$



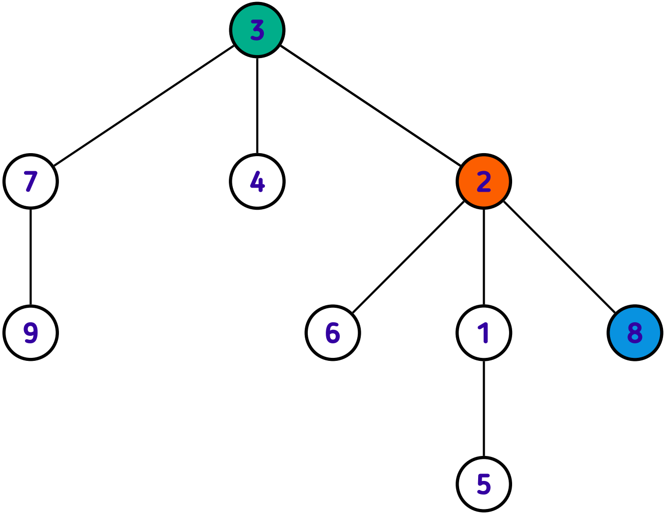
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2		
9	7		

succ$(u, 2^k)$



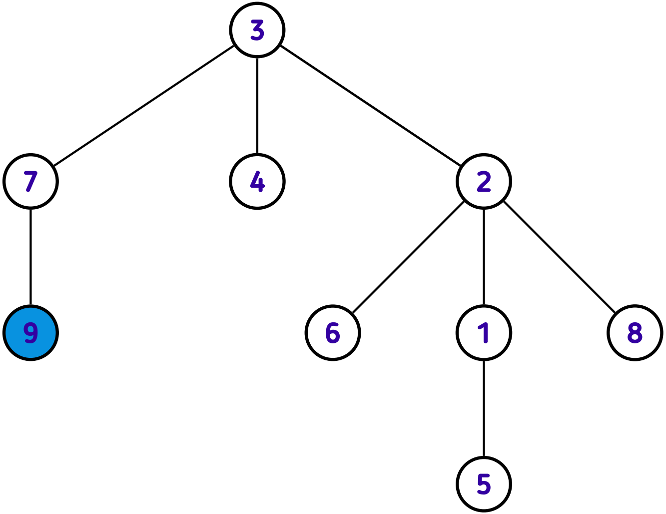
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2		
9	7		

succ$(u, 2^k)$



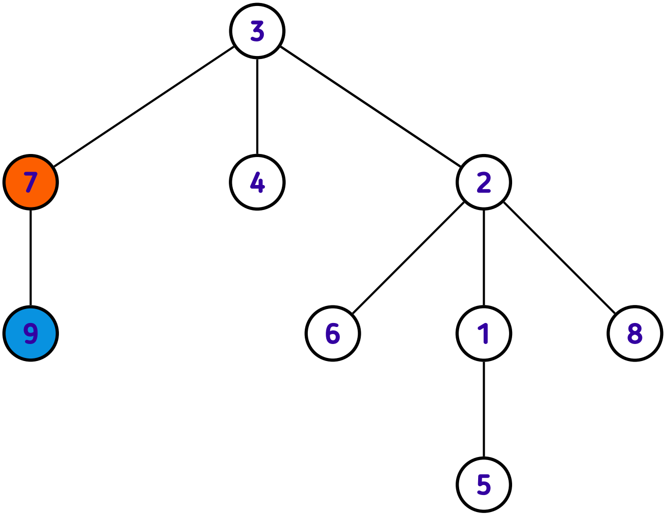
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2	3	
9	7		

succ$(u, 2^k)$



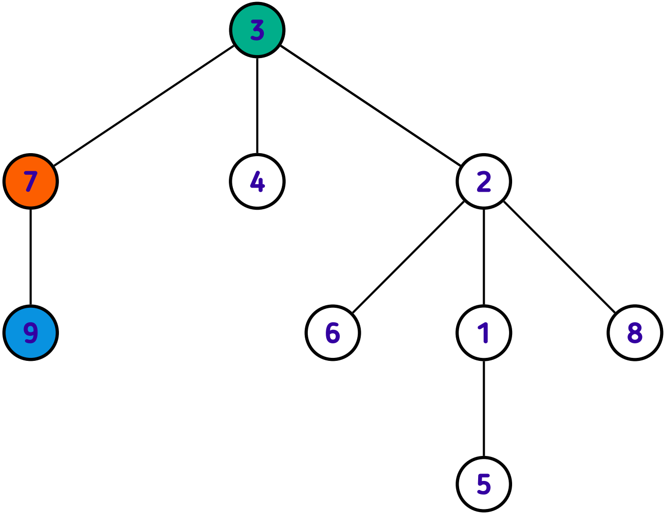
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2	3	
9	7		

succ$(u, 2^k)$



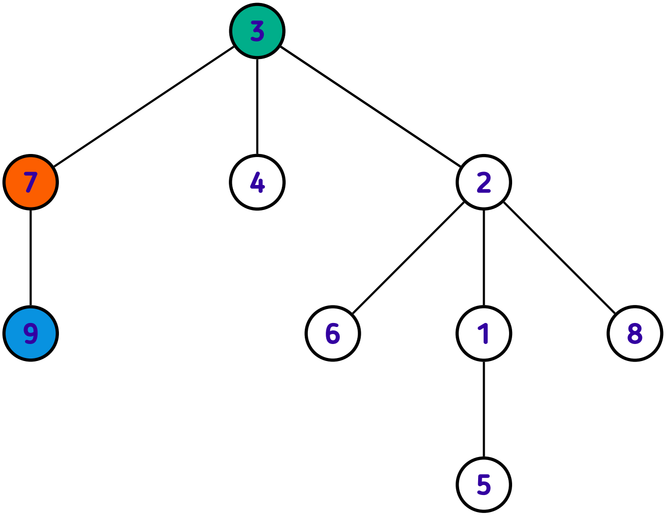
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2	3	
9	7		

succ$(u, 2^k)$



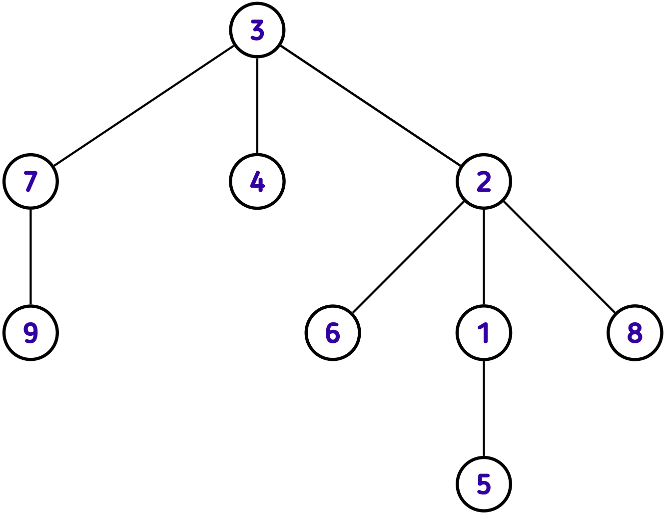
	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2	3	
9	7		

succ$(u, 2^k)$



	2^0	2^1	2^2
1	2	3	
2	3	-	
3	-	-	
4	3	-	
5	1	2	
6	2	3	
7	3	-	
8	2	3	
9	7	3	

succ$(u, 2^k)$



	2^0	2^1	2^2
1	2	3	-
2	3	-	-
3	-	-	-
4	3	-	-
5	1	2	-
6	2	3	-
7	3	-	-
8	2	3	-
9	7	3	-

succ$(u, 2^k)$

```
int ancestor(int u, int k, int N, const vector<vector<int>>& as)
{
    if (k >= N)
        return 0;

    int level = 0;

    while (k)
    {
        if (k & 1)
            u = as[u][level];

        k >>= 1;
        ++level;
    }

    return u;
}
```

```
auto precomp(int root, int N)
{
    int M = 0;

    while ((1 << (M + 1)) <= N)
        ++M;

    vector<vector<int>> as(N + 1, vector<int>(M + 1, 0));

    vector<int> ps(N + 1, 0);
    dfs(root, 0, ps);

    for (int u = 1; u <= N; ++u)
        as[u][0] = ps[u];

    for (int i = 1; i <= M; ++i)
        for (int u = 1; u <= N; ++u)
            as[u][i] = as[as[u][i - 1]][i - 1];

    return as;
}
```

```
void dfs(int u, int p, vector<int>& ps)
{
    ps[u] = p;

    for (auto v : adj[u])
        if (v != p)
            dfs(v, u, ps);
}
```

Referências

1. **HALIM**, **Felix**; **HALIM**, **Steve**. ***Competitive Programming 3*, 2010.**

2. **LAAKSONEN**, **Antti**. ***Competitive Programmer's Handbook*, 2018.**