

# Grafos

*Travessia por largura*

**Prof. Edson Alves**

**Faculdade UnB Gama**

## Travessia por largura (*Breadth-first search*)

## Travessia por largura (*Breadth-first search*)

Seja  $s$  o vértice de partida e  $u$  o vértice observado no momento. As regras abaixo definem a BFS:

## Travessia por largura (*Breadth-first search*)

Seja  $s$  o vértice de partida e  $u$  o vértice observado no momento. As regras abaixo definem a BFS:

1. Faça  $u = s$

## Travessia por largura (*Breadth-first search*)

Seja  $s$  o vértice de partida e  $u$  o vértice observado no momento. As regras abaixo definem a BFS:

1. Faça  $u = s$

2. Visite  $u$

## Travessia por largura (*Breadth-first search*)

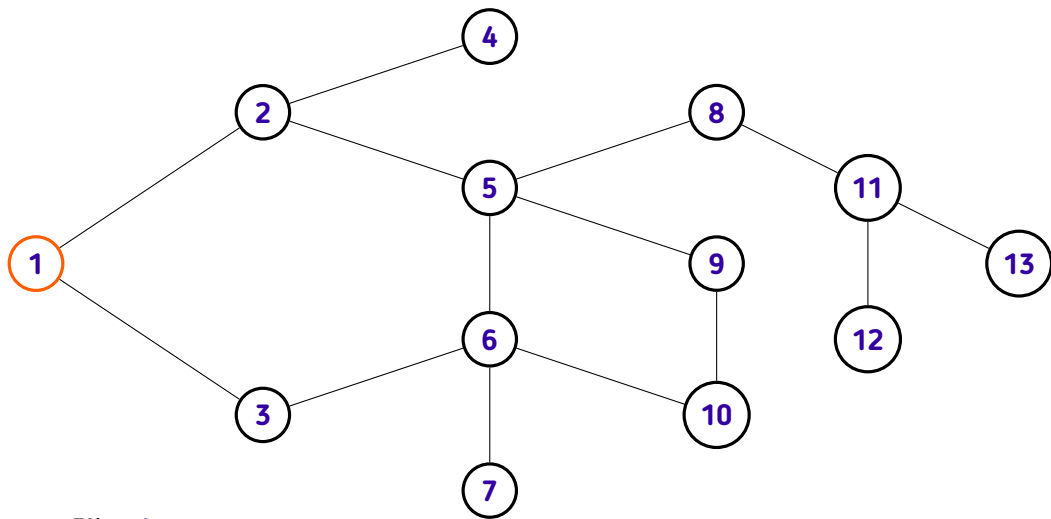
Seja  $s$  o vértice de partida e  $u$  o vértice observado no momento. As regras abaixo definem a BFS:

1. Faça  $u = s$
2. Visite  $u$
3. Enfileire todos vizinhos de  $u$  não visitados

## Travessia por largura (*Breadth-first search*)

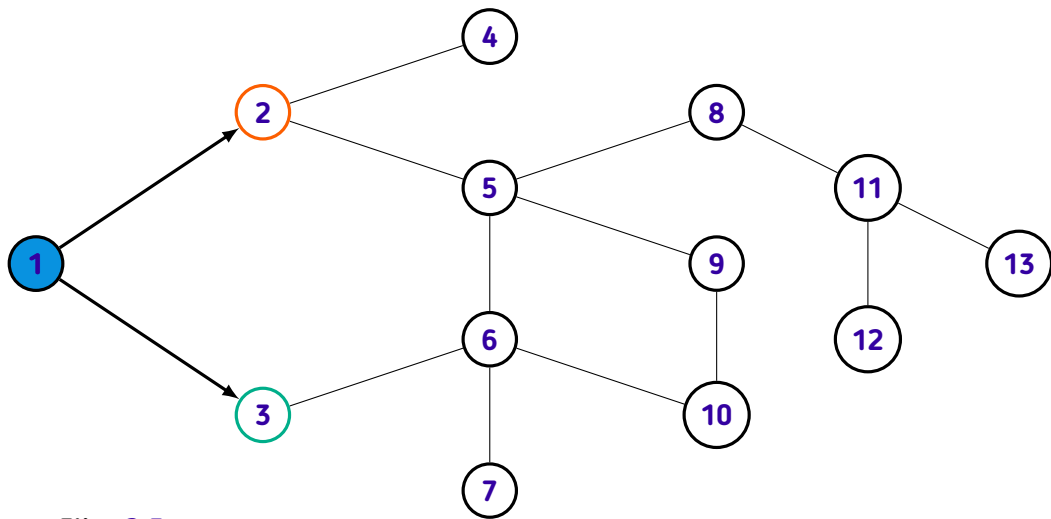
Seja  $s$  o vértice de partida e  $u$  o vértice observado no momento. As regras abaixo definem a BFS:

1. Faça  $u = s$
2. Visite  $u$
3. Enfileire todos vizinhos de  $u$  não visitados
4. Extraia o próximo elemento  $p$  da fila e faça  $u = p$

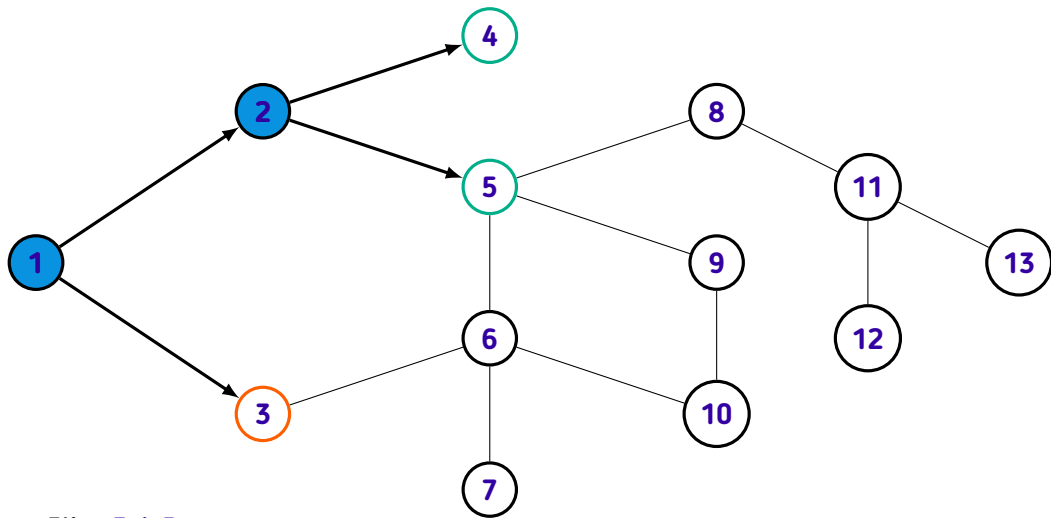


Fila: 1

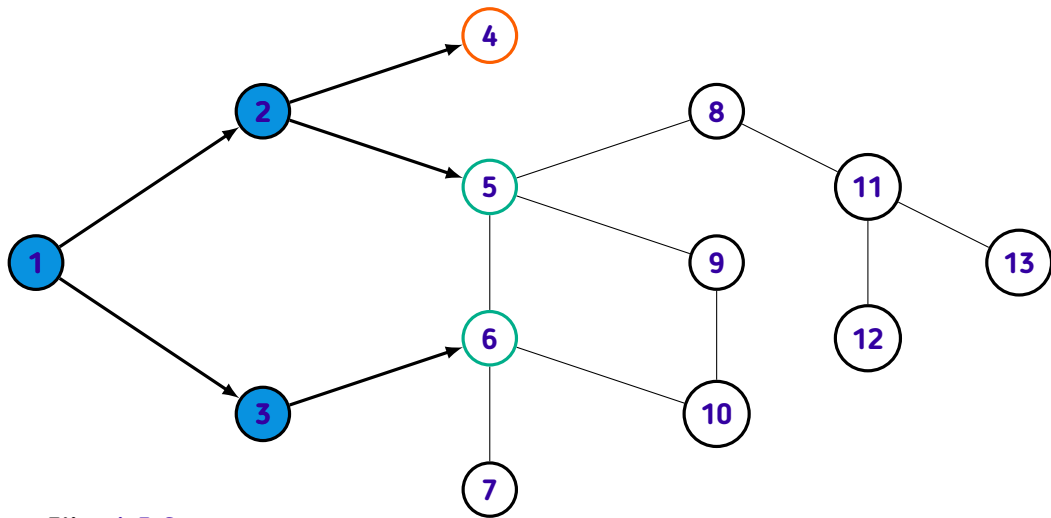




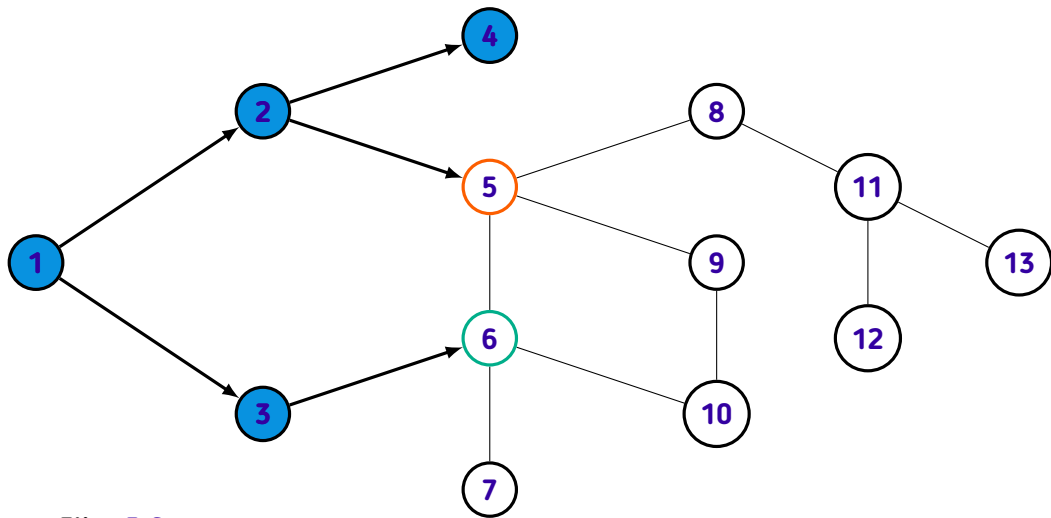
Fila: 2 3



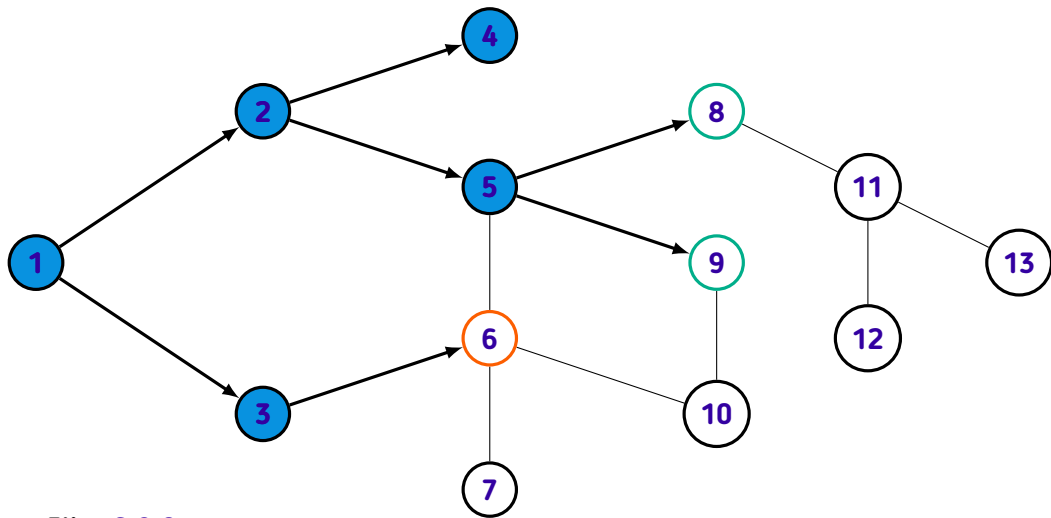
Fila: 3 4 5



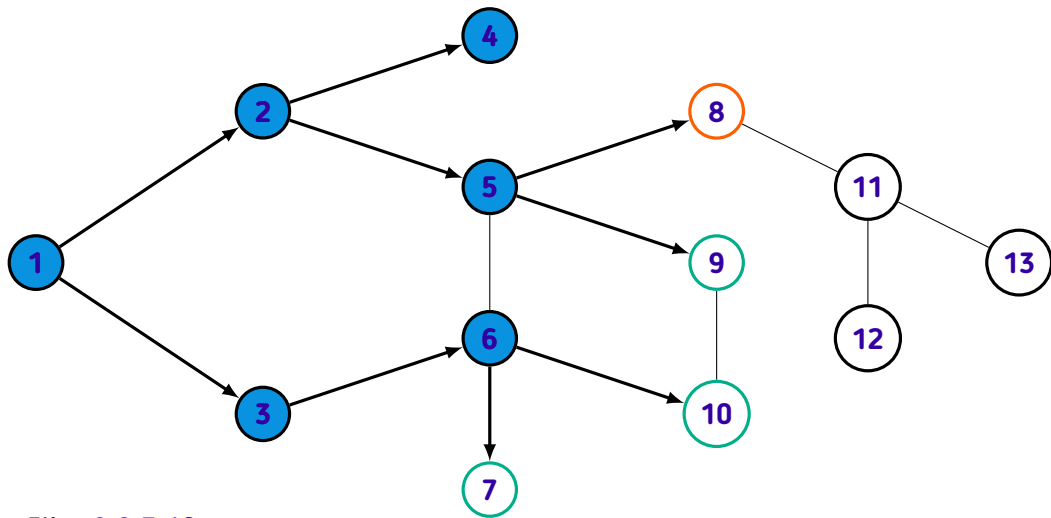
Fila: 4 5 6



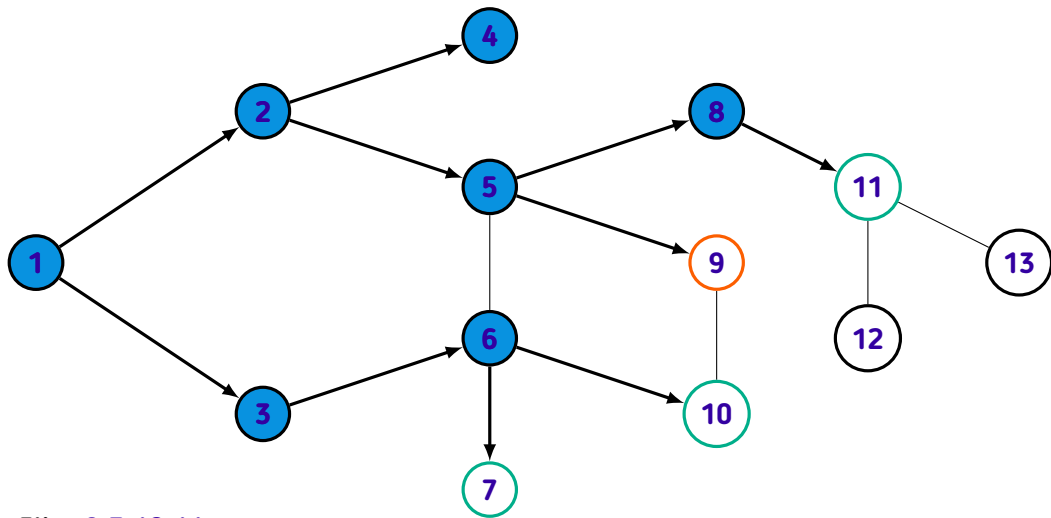
Fila: 5 6



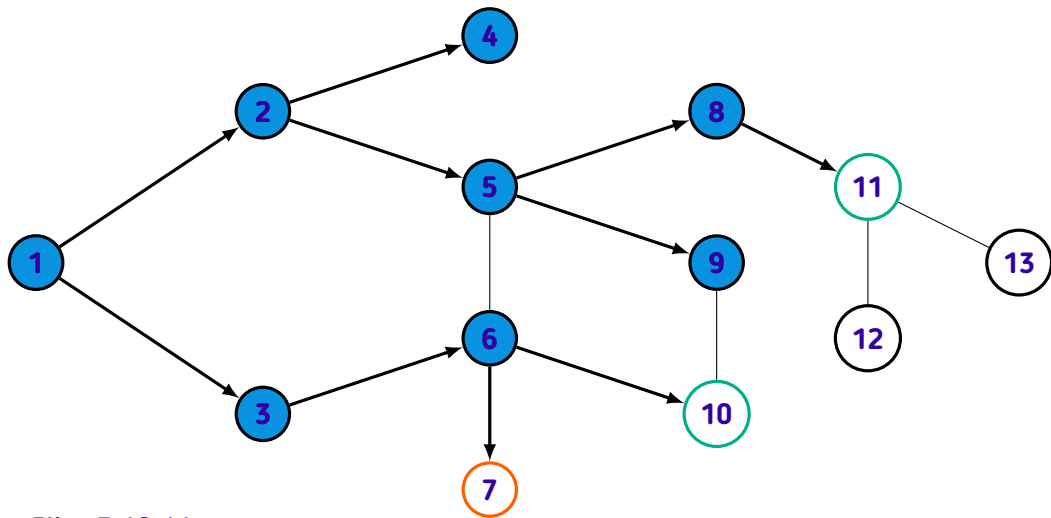
Fila: 6 8 9



Fila: 8 9 7 10

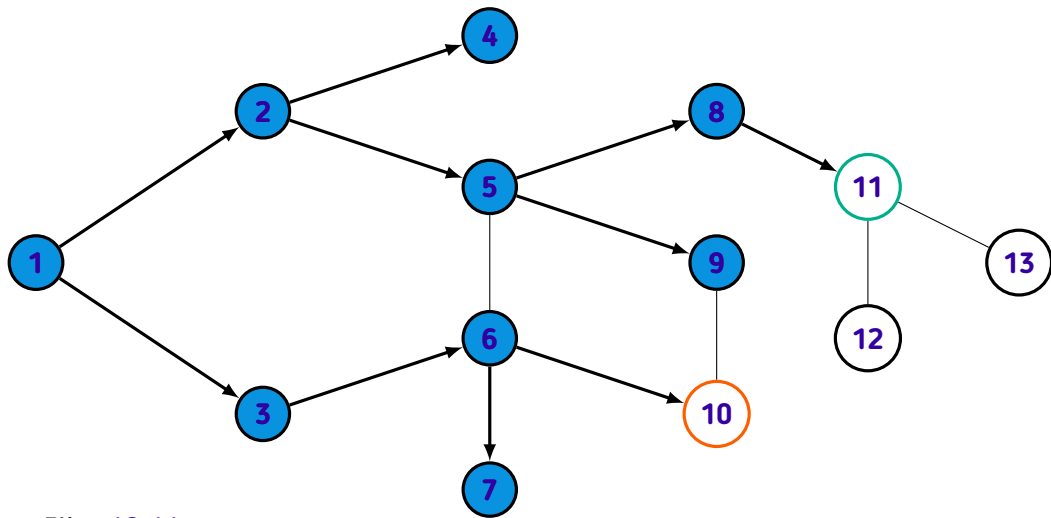


Fila: 9 7 10 11

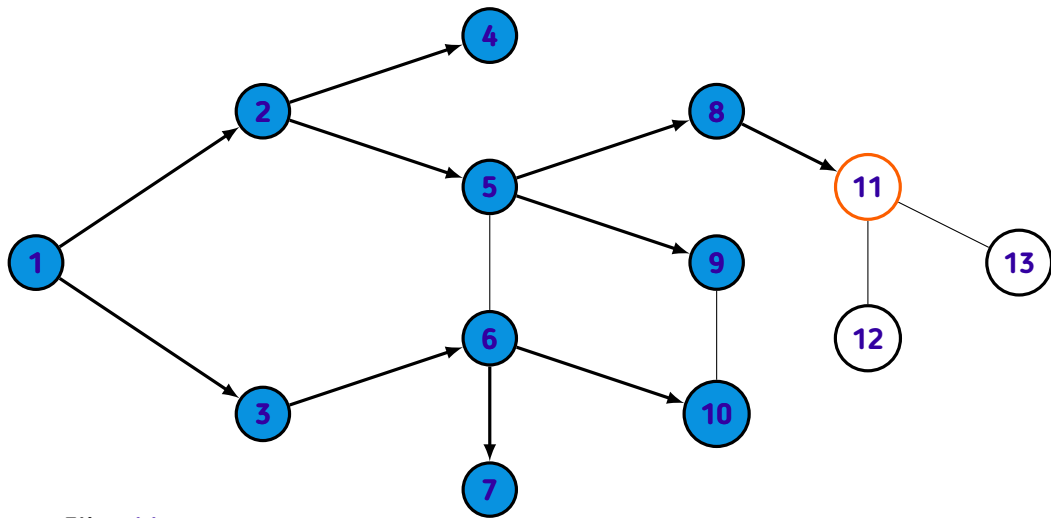


Fila: 7 10 11

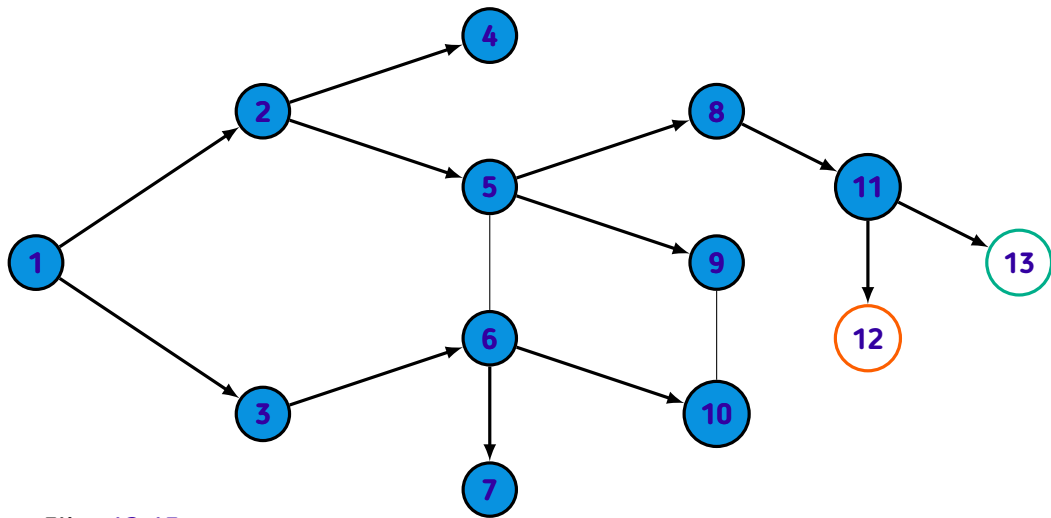




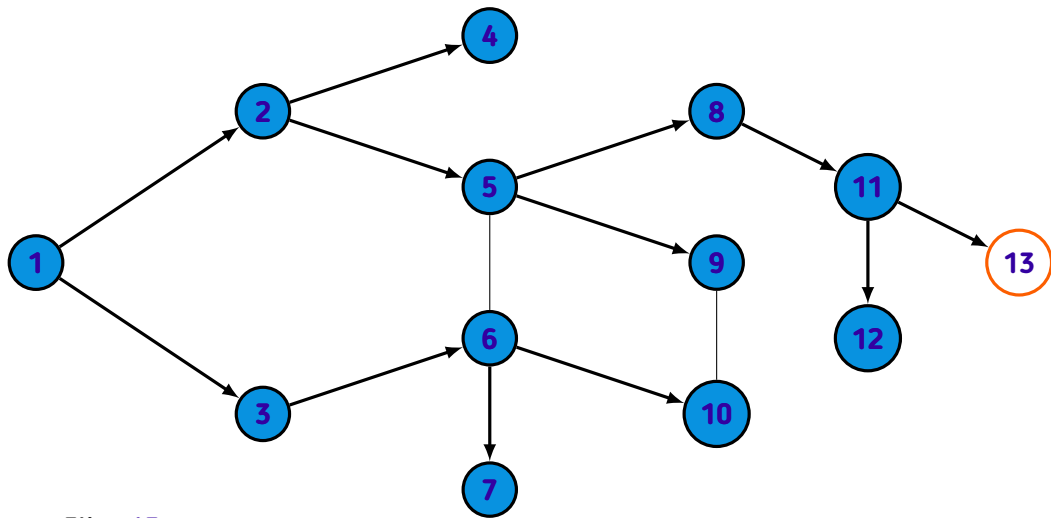
Fila: 10 11



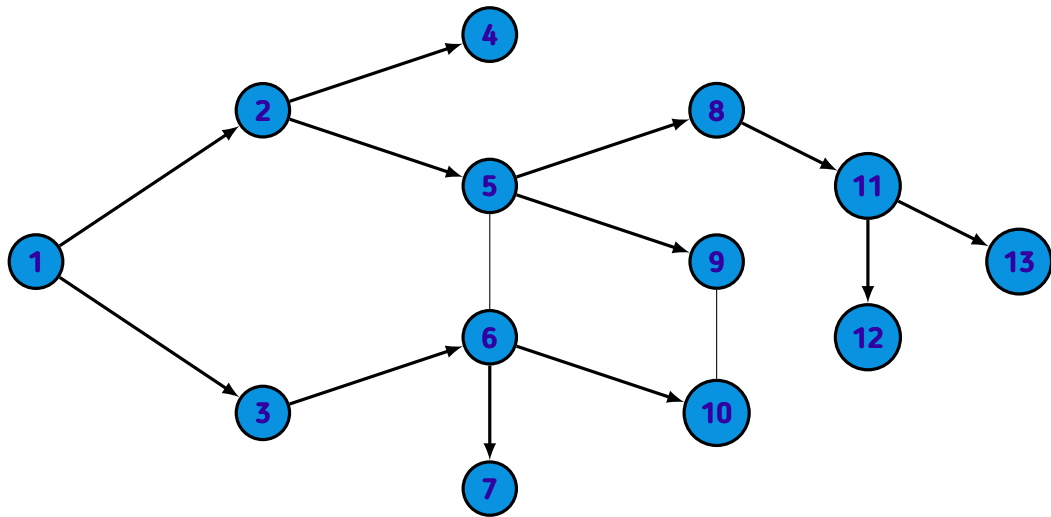
Fila: 11



Fila: 12 13



Fila: 13



## **Características da BFS**

## Características da BFS

- ★ Um subproduto da BFS são as distâncias, em arestas, até  $s$

## Características da BFS

- ★ Um subproduto da BFS são as distâncias, em arestas, até  $s$
- ★ A DFS e a BFS visitam os mesmos vértices, em ordem distintas



## Características da BFS

- ★ Um subproduto da BFS são as distâncias, em arestas, até  $s$
- ★ A DFS e a BFS visitam os mesmos vértices, em ordem distintas
- ★ Complexidade:  $O(N + M)$ , a mesma da DFS

## Características da BFS

- ★ Um subproduto da BFS são as distâncias, em arestas, até  $s$
- ★ A DFS e a BFS visitam os mesmos vértices, em ordem distintas
- ★ Complexidade:  $O(N + M)$ , a mesma da DFS
- ★ Em matrizes de adjacências, a complexidade é  $O(N^2)$

## **Implementação da BFS**

## Implementação da BFS

- ★ Mais elaborada do que a da DFS, pois não usa recursão

## Implementação da BFS

- ★ Mais elaborada do que a da DFS, pois não usa recursão
- ★ Ela demanda uma fila para a manutenção da ordem de travessia

## Implementação da BFS

- ★ Mais elaborada do que a da DFS, pois não usa recursão
- ★ Ela demanda uma fila para a manutenção da ordem de travessia
- ★ O vetor  $\text{dist}[u]$  armazena a distância de  $u$  até  $s$ , em arestas

```
vector<int> bfs(int s, int N) {  
    vector<int> dist(N + 1, oo);  
    queue<int> q;  
  
    dist[s] = 0; q.push(s);  
  
    while (not q.empty())  
    {  
        auto u = q.front(); q.pop();  
  
        // visita/processa u  
  
        for (auto v : adj[u]) {  
            if (dist[v] == oo) {  
                dist[v] = dist[u] + 1; q.push(v);  
            }  
        }  
    }  
  
    return dist;  
}
```

## Problemas sugeridos

1. [AtCoder Beginner Contest 132 – Problem E: Hopscotch Addict](#)
2. [Codeforces Beta Round #3 – Problem A: Shortest Path of the King](#)
3. [Codeforces Round #470 \(rated, Div. 2\) – Problem A: Protect Sheep](#)
4. [OJ 10687 – Monitoring the Amazon](#)



## Referências

1. FILIPEK, Bartlomej. *C++17 in Detail*, 2018.
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.