Estruturas de Dados II

Grafos: Busca em Largura

Prof[a]. Juliana de Santi
Prof. Rodrigo Minetto
Universidade Tecnológica Federal do Paraná
Material compilado de: Cormen e material de Cid et al.
IC-UNICAMP.

Sumário

Busca em largura

Dizemos que um vértice v é **alcançável** a partir de um vértice s em um grafo G se existe um caminho de s a v em G.

Definição: a distância de s a v é o comprimento de um caminho mais curto de s a v. Se v não é alcançável a partir de s, então dizemos que a distância de s a v é ∞ (infinita).

Busca em largura

Busca em largura recebe um grafo $G = (V, E)$ e um vértice especificado s chamado **fonte (source)**. Percorre todos os vértices alcançáveis a partir de s em ordem de distância deste. Vértices a mesma distância podem ser percorridos em qualquer ordem.

Busca em largura

Constrói uma árvore de busca em largura com raiz s. Cada caminho de s a um vértice v nesta árvore corresponde a um caminho mais curto de s a v.

Busca em largura

Inicialmente a árvore de busca em largura contém apenas o vértice fonte s. Para cada vizinho v de s, o vértice v e a aresta (s, v) são acrescentadas à árvore.

Busca em largura

O processo é repetido para os vizinhos dos vizinhos de s e assim por diante, até que todos os vértices atingíveis por s sejam inseridos na árvore. Este processo é implementado através de uma **fila Q**.

Busca em largura

Para facilitar o entendimento do algoritmo, vamos atribuir cores durante a busca em largura. Ou seja, para cada vértice **v** guarda-se sua cor atual **cor[v]** que pode ser branco, cinza ou preto.

- Cor branca = "não visitado".
- Cor **cinza** = "visitado pela primeira vez".
- Cor **preta** = "teve seus vizinhos visitados".

Para efeito de implementação, isto não é realmente necessário.

Busca em largura

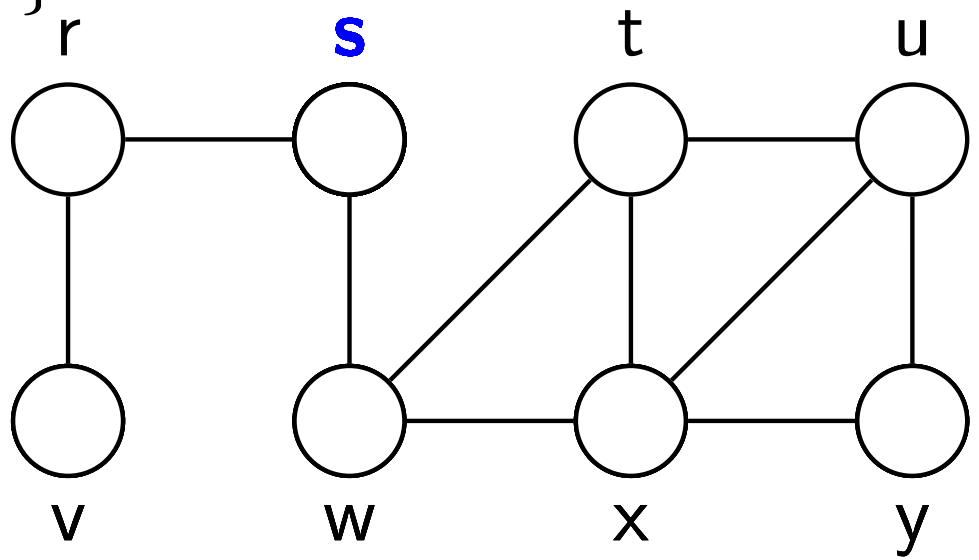
- A raiz da árvore de busca em largura é s.
- Cada vértice v (diferente de s) possui um pai $\pi[v]$.
- O caminho de v a s na árvore é dado por: $v, \pi[v], \pi[\pi[v]], \pi[\pi[\pi[v]]], \ldots, s$.
- Uma variável $d[v]$ é usada para armazenar a distância de s a v (que será determinada durante a busca).

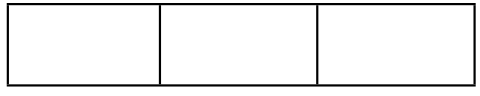
Busca em largura

Recebe um grafo G (na forma de listas de adjacências) e um vértice $s \in V[G]$ e devolve: (i) para cada vértice v, a distância de s a v em G; e (ii) uma árvore de busca em largura.

Busca-em-largura (G, s)

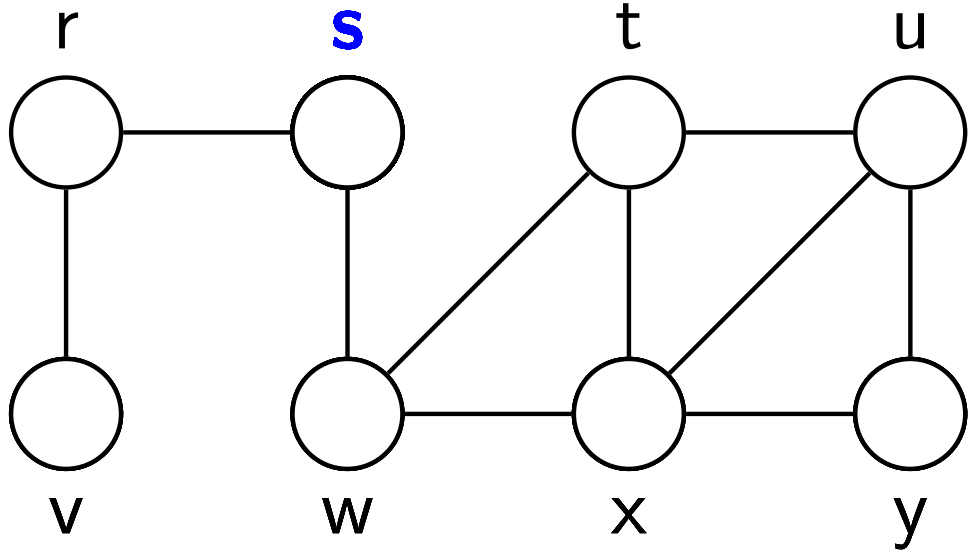
▷ **Para cada u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;

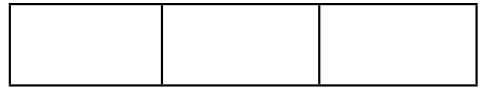


Q = 

Busca-em-largura (G, s)

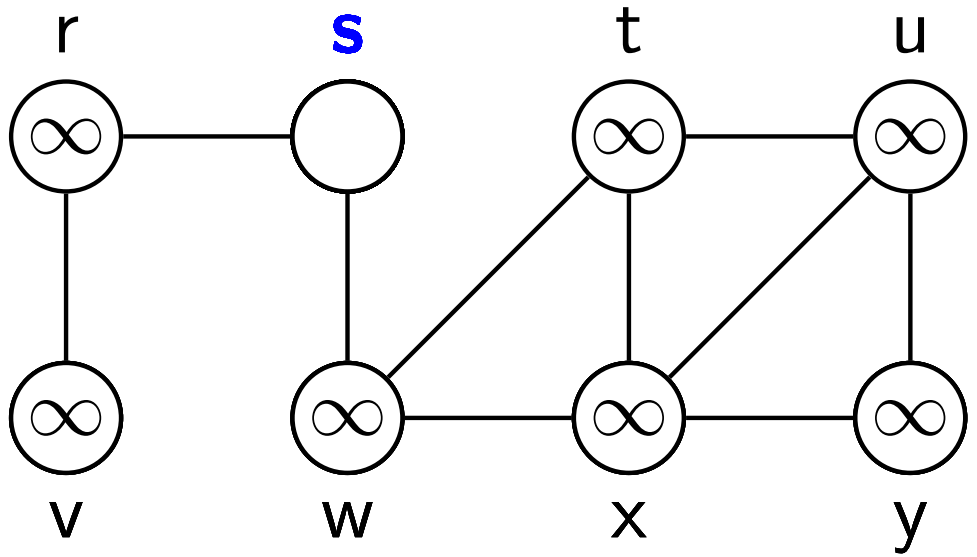
Para cada **u** $\in V[G] - \{s\}$
$\triangleright$ cor[**u**] $\leftarrow$ branco;
d[**u**] $\leftarrow \infty$;
π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;



Q = 

Busca-em-largura (G, s)

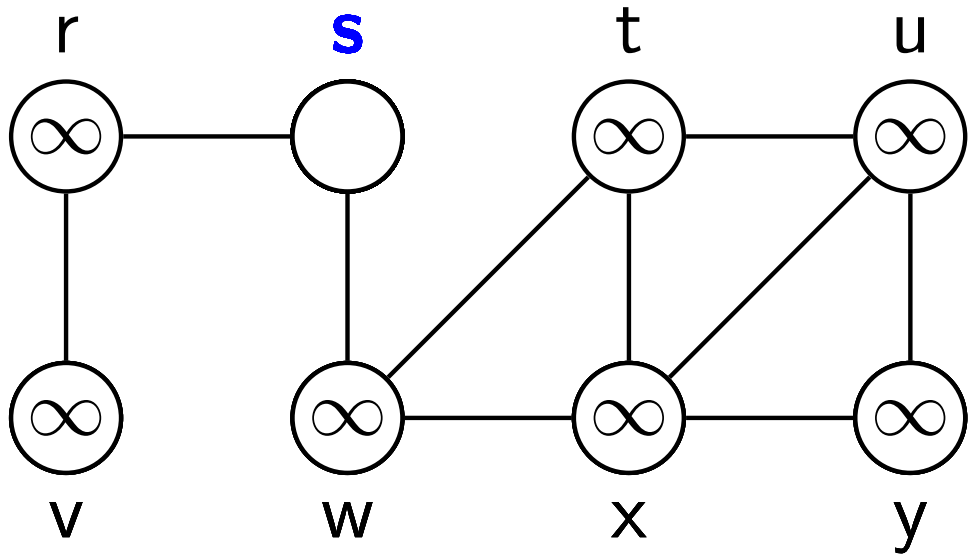
Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
$\triangleright$ d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;

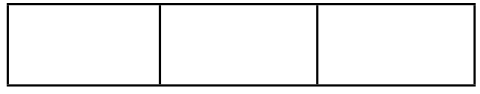


Q =

Busca-em-largura (G, s)

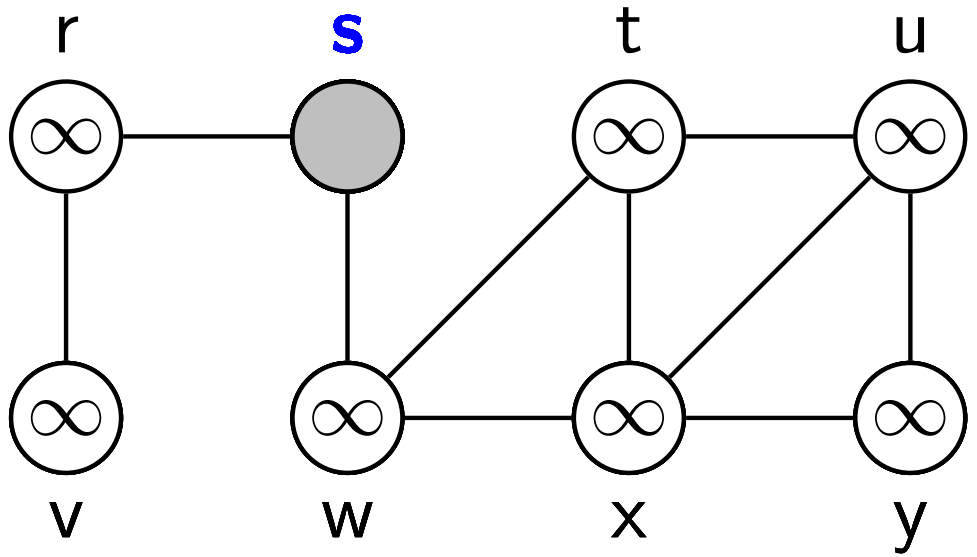
Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
$\triangleright$ π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;



Q = 

Busca-em-largura (G, s)

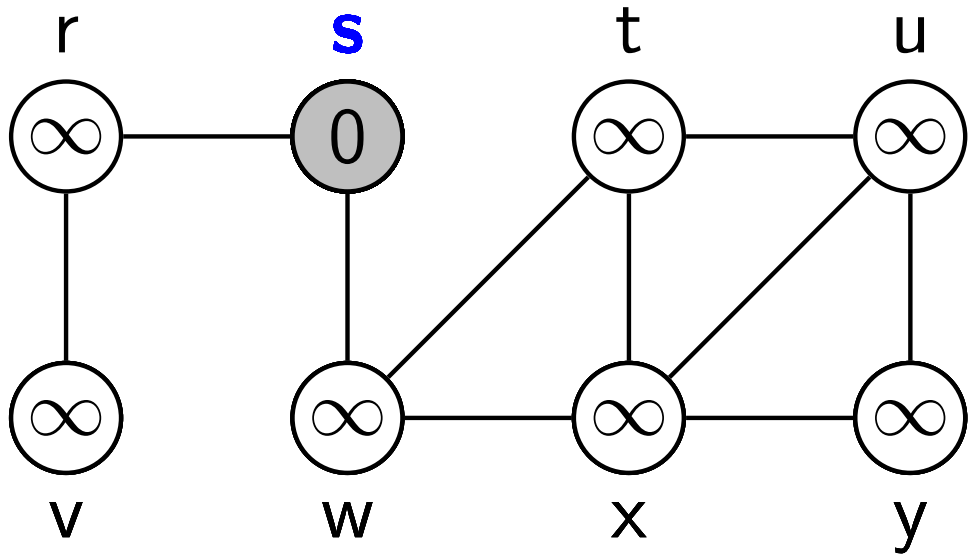
Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
$\triangleright$cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;

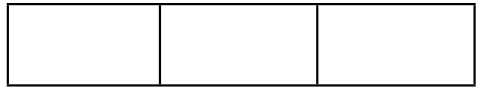


Q =

Busca-em-largura (G, s)

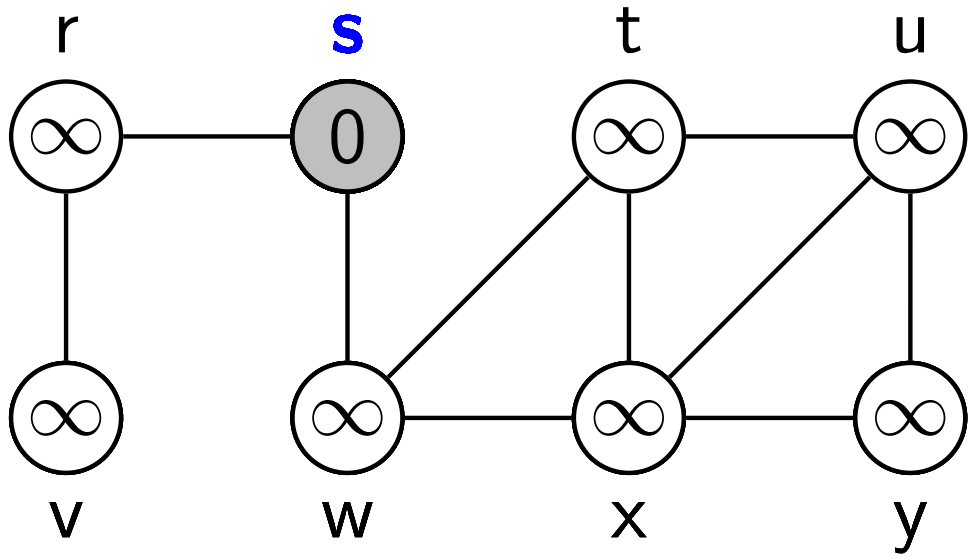
Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
$\triangleright$d[**s**] $\leftarrow$ 0;
π[**s**] $\leftarrow$ NIL;



Q = 

Busca-em-largura (G, s)

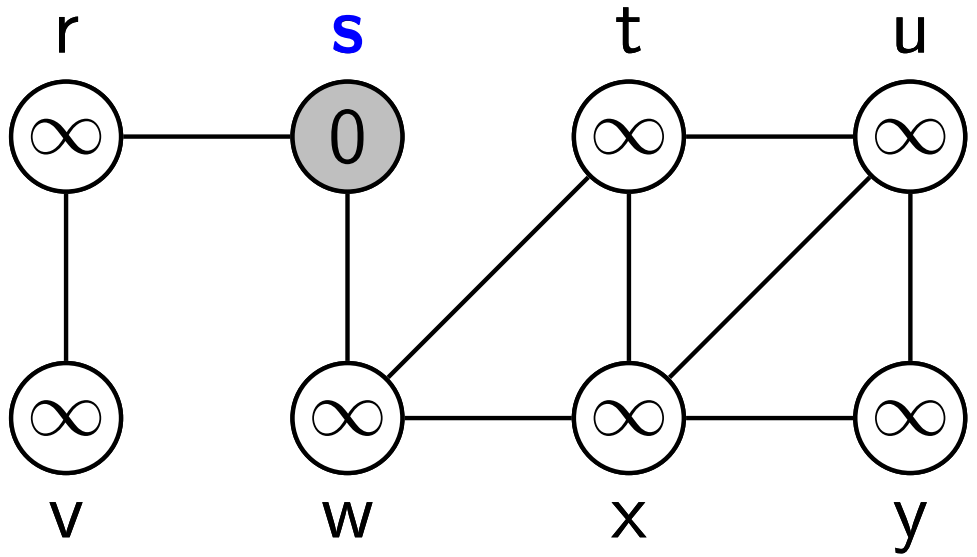
Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow$ 0;
$\triangleright \pi$[**s**] $\leftarrow$ NIL;

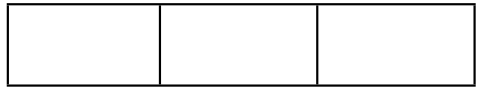


Q =

Busca-em-largura (G, s)

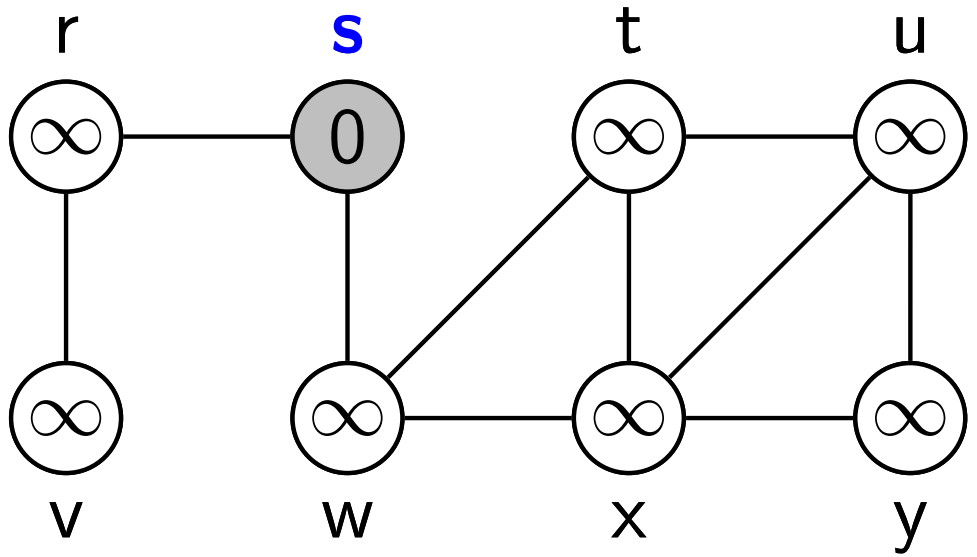
$\triangleright$Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ DEQUEUE (Q);
    Para cada **v** $\in Adj$[**u**]
        Se cor[**v**] $=$ branco;
            cor[**v**] $\leftarrow$ cinza;
            d[**v**] $\leftarrow$ d[**u**] $+ 1$;
            π[**v**] $\leftarrow$ **u**;
            ENQUEUE (Q,**v**);
    cor[**u**] $\leftarrow$ preto;
Retorne d, π;



Q $=$ 

Busca-em-largura (G, s)

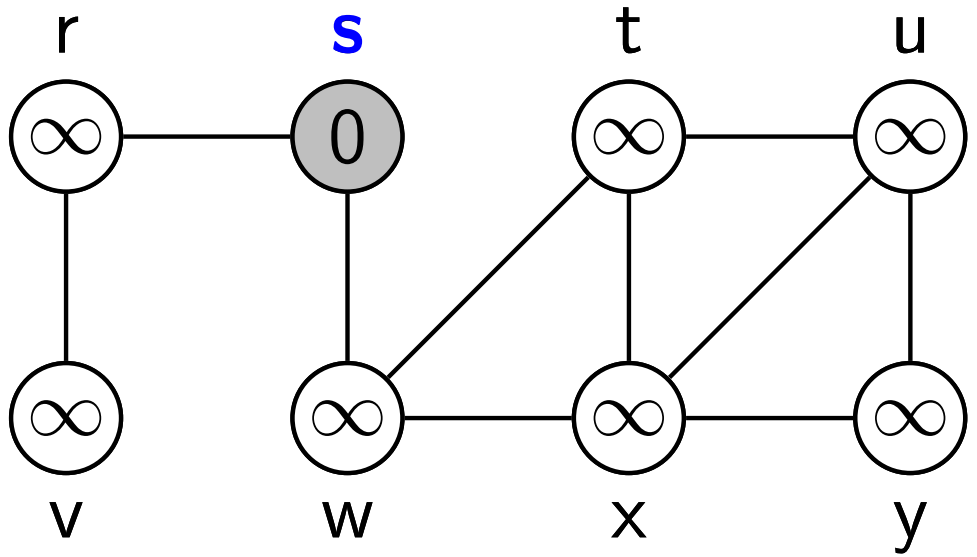
```
Q ← ∅;
▷ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← DEQUEUE (Q);
    Para cada v ∈ Adj[u]
        Se cor[v] = branco;
            cor[v] ← cinza;
            d[v] ← d[u] + 1;
            π[v] ← u;
            ENQUEUE (Q,v);
    cor[u] ← preto;
Retorne d, π;
```



Q = | s | | |

Busca-em-largura (G, s)

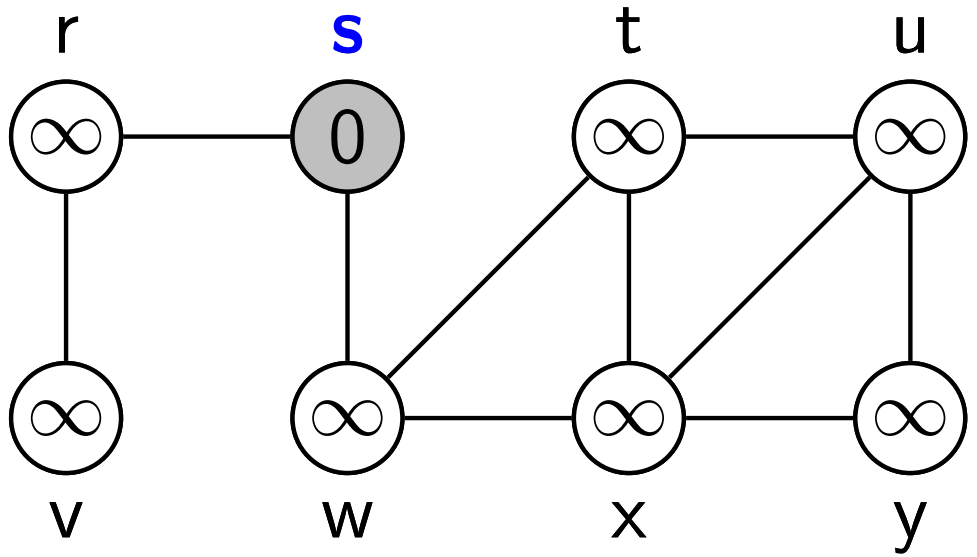
```
Q ← ∅;
ENQUEUE (Q,s);
▷Enquanto Q ≠ ∅
    u ← DEQUEUE (Q);
    Para cada v ∈ Adj[u]
        Se cor[v] = branco;
            cor[v] ← cinza;
            d[v] ← d[u] + 1;
            π[v] ← u;
            ENQUEUE (Q,v);
    cor[u] ← preto;
Retorne d, π;
```



Q = | **s** | | |

Busca-em-largura (G, s)

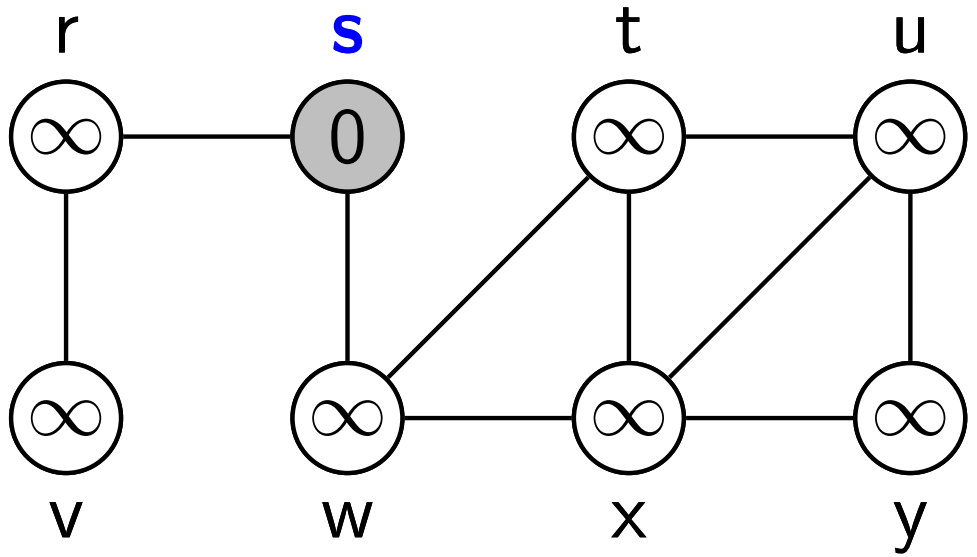
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada v ∈ Adj[u]
            Se cor[v] = branco;
                  cor[v] ← cinza;
                  d[v] ← d[u] + 1;
                  π[v] ← u;
                  ENQUEUE (Q,v);
      cor[u] ← preto;
Retorne d, π;
```

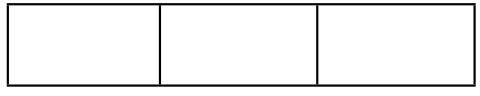


Q = | s | | |

Busca-em-largura (G, s)

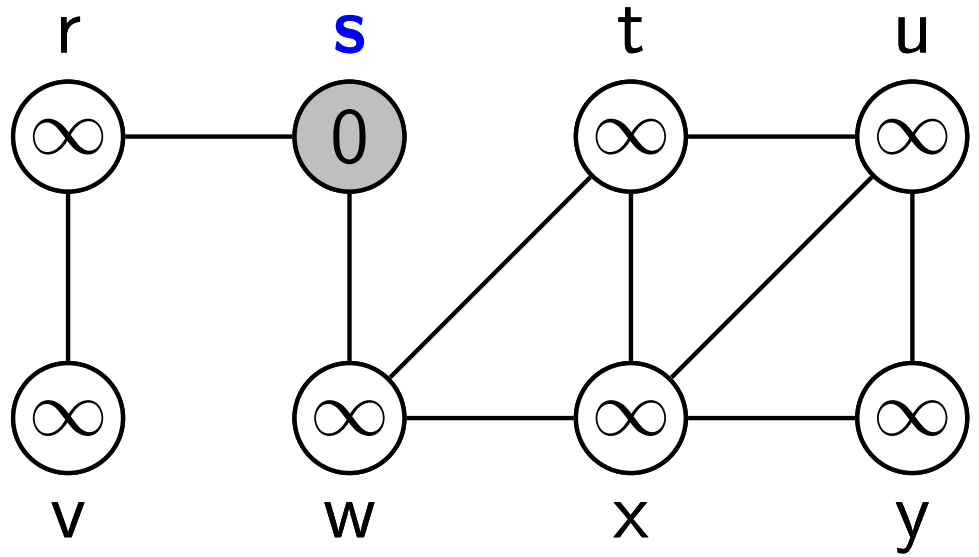
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← s;
    Para cada w ∈ Adj[s]
        Se cor[w] = branco;
            cor[w] ← cinza;
            d[w] ← d[s] + 1;
            π[w] ← s;
            ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```

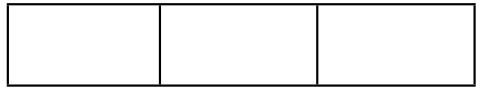


Q = 

Busca-em-largura (G, s)

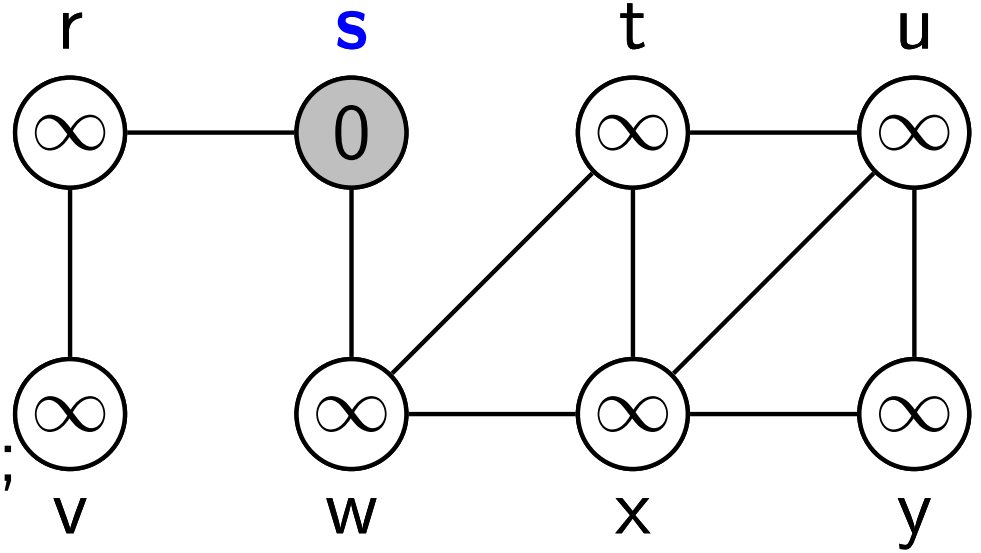
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
▷   Para cada w ∈ Adj[s]
      Se cor[w] = branco;
          cor[w] ← cinza;
          d[w] ← d[s] + 1;
          π[w] ← s;
          ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```

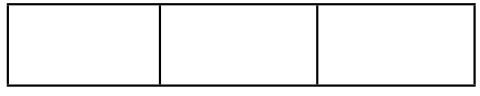


Q = 

Busca-em-largura (G, s)

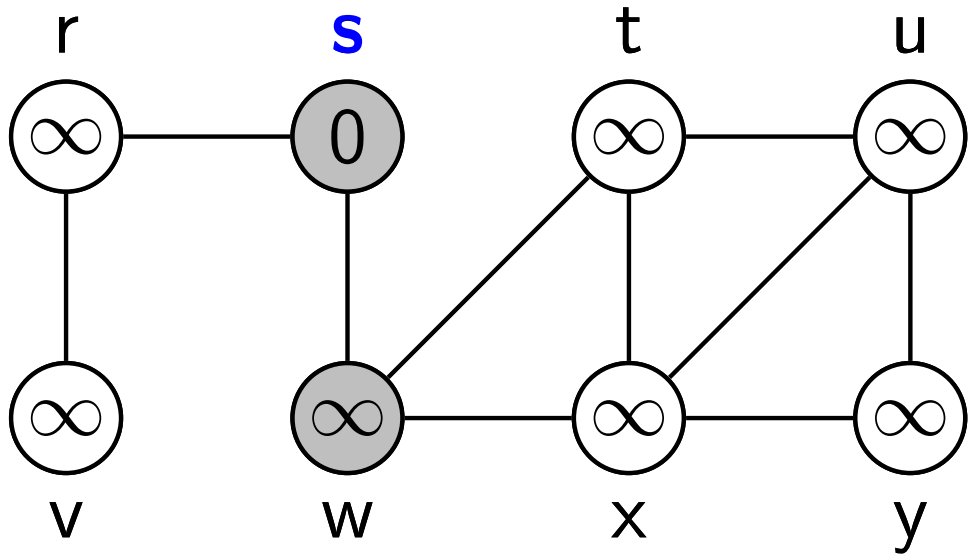
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada w ∈ Adj[s]
▷       Se cor[w] = branco;
          cor[w] ← cinza;
          d[w] ← d[s] + 1;
          π[w] ← s;
          ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```

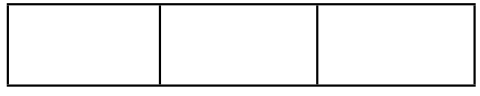


Q = 

Busca-em-largura (G, s)

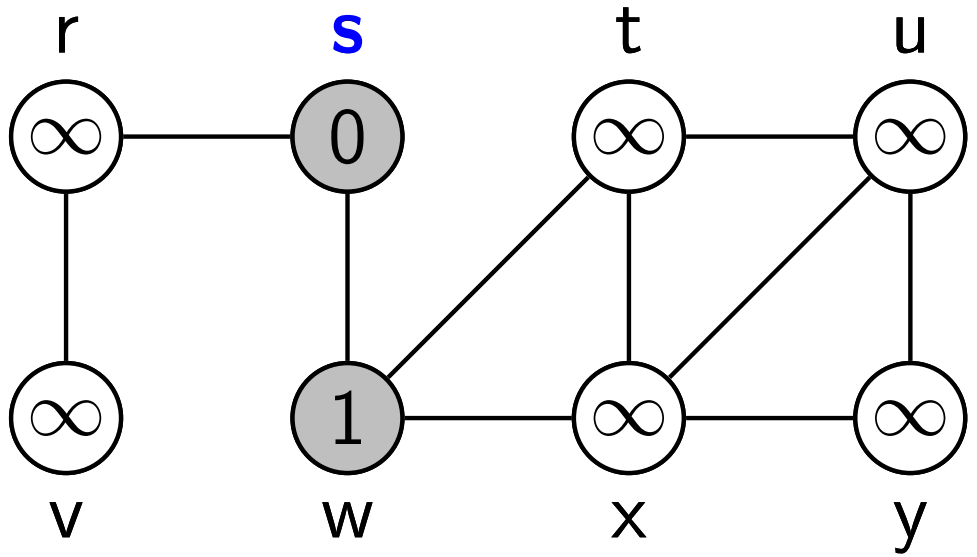
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada w ∈ Adj[s]
        Se cor[w] = branco;
▷                 cor[w] ← cinza;
                  d[w] ← d[s] + 1;
                  π[w] ← s;
                  ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```



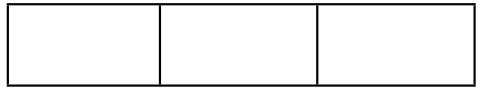
Q = 

Busca-em-largura (G, s)

```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada w ∈ Adj[s]
        Se cor[w] = branco;
            cor[w] ← cinza;
▷               d[w] ← d[s] + 1;
            π[w] ← s;
            ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```

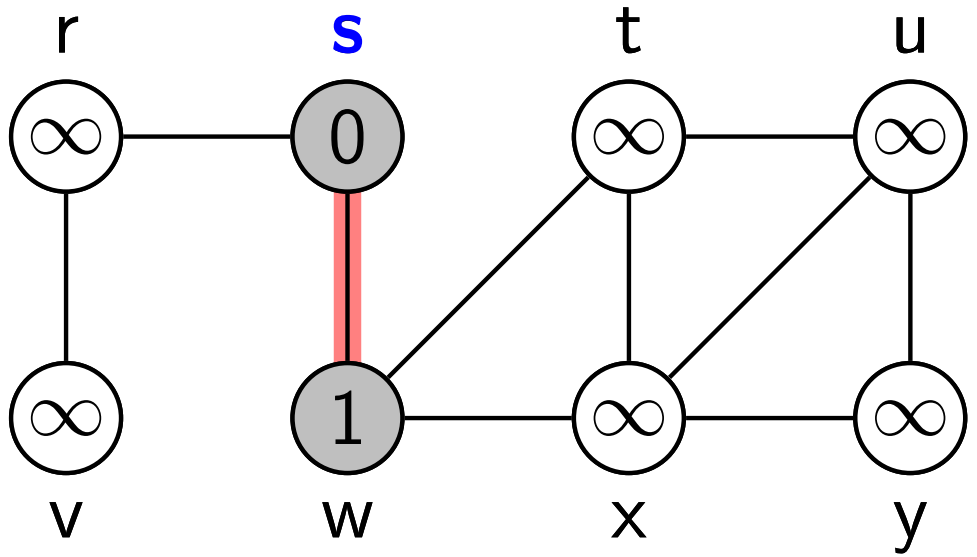


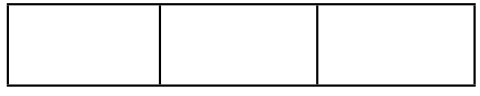
Q =



Busca-em-largura (G, s)

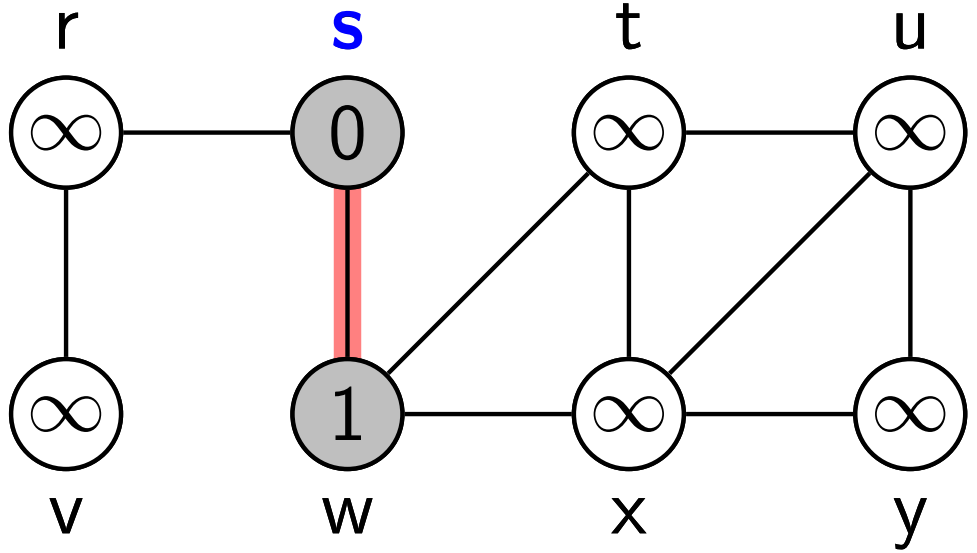
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada w ∈ Adj[s]
        Se cor[w] = branco;
            cor[w] ← cinza;
            d[w] ← d[s] + 1;
▷               π[w] ← s;
            ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```



Q = 

Busca-em-largura (G, s)

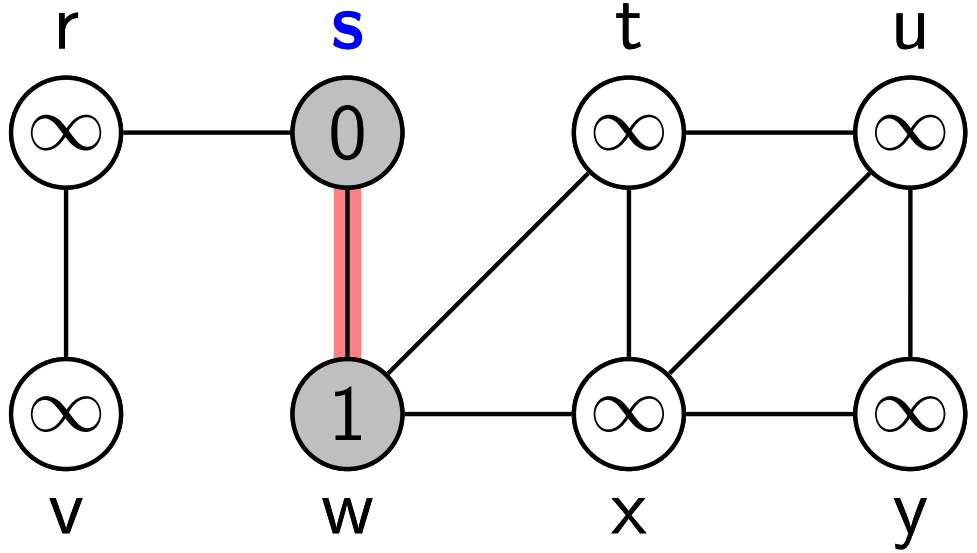
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada w ∈ Adj[s]
        Se cor[w] = branco;
            cor[w] ← cinza;
            d[w] ← d[s] + 1;
            π[w] ← s;
▷               ENQUEUE (Q,w);
    cor[s] ← preto;
Retorne d, π;
```



Q = | **w** | | |

Busca-em-largura (G, s)

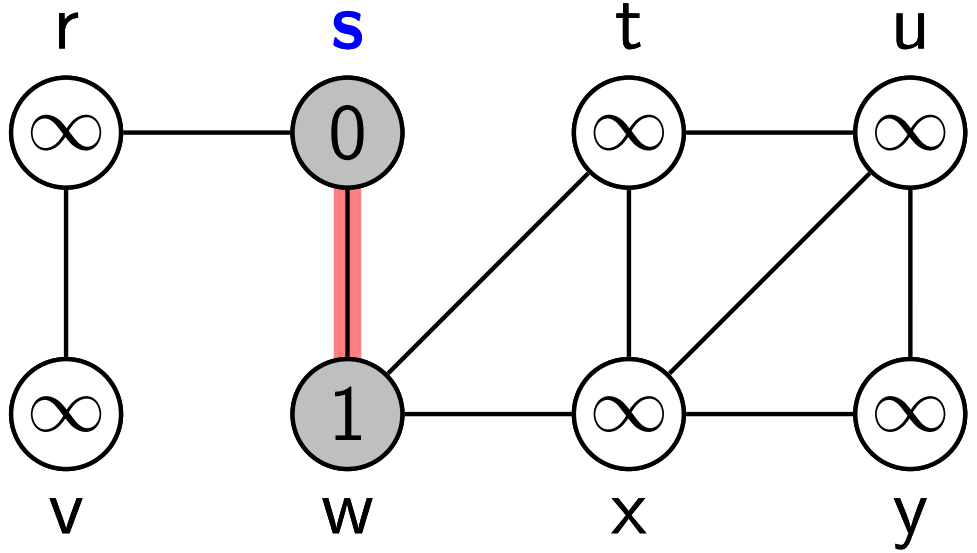
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
▷     Para cada r ∈ Adj[s]
        Se cor[r] = branco;
            cor[r] ← cinza;
            d[r] ← d[s] + 1;
            π[r] ← s;
            ENQUEUE (Q,r);
    cor[s] ← preto;
Retorne d, π;
```



Q = | **w** | | |

Busca-em-largura (G, s)

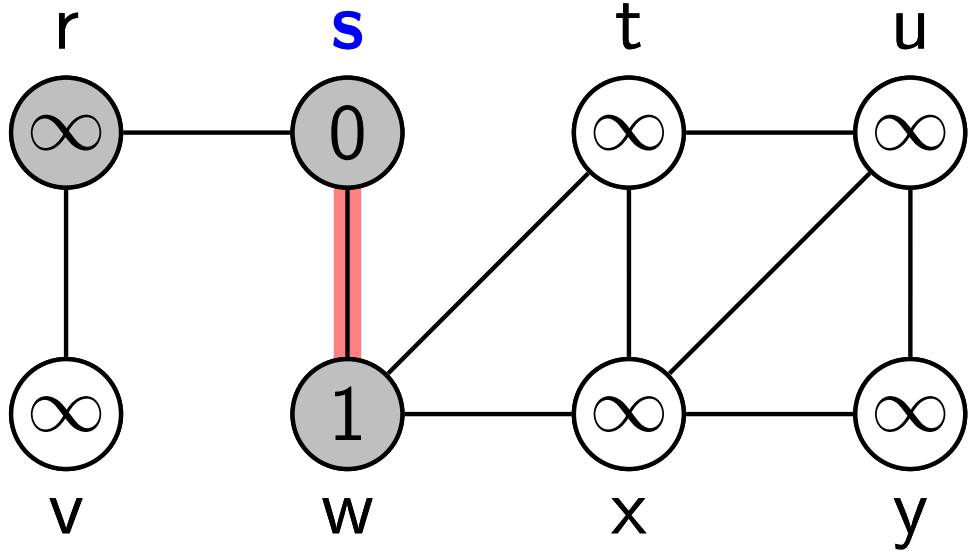
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada r ∈ Adj[s]
▷       Se cor[r] = branco;
          cor[r] ← cinza;
          d[r] ← d[s] + 1;
          π[r] ← s;
          ENQUEUE (Q,r);
    cor[s] ← preto;
Retorne d, π;
```



Q = | **w** | | |

Busca-em-largura (G, s)

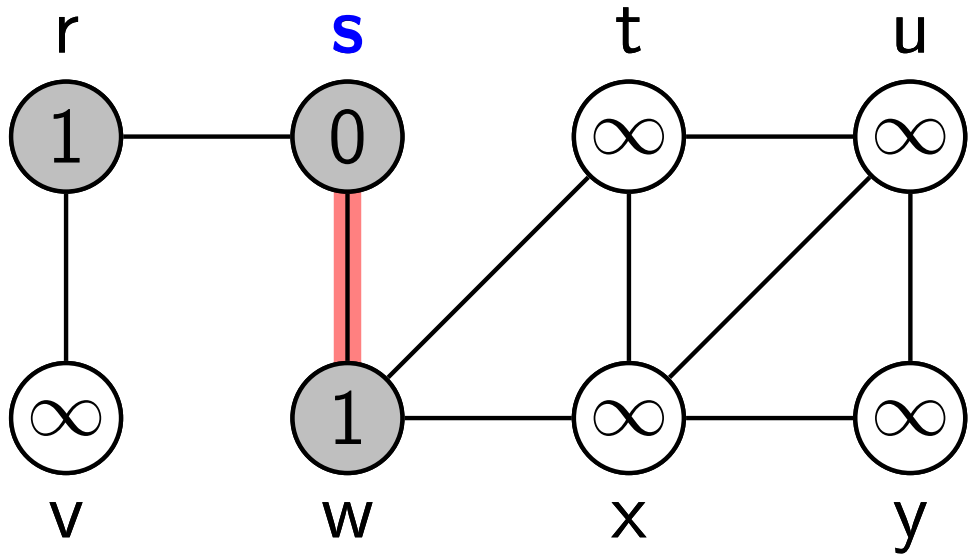
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada r ∈ Adj[s]
        Se cor[r] = branco;
▷                 cor[r] ← cinza;
              d[r] ← d[s] + 1;
              π[r] ← s;
              ENQUEUE (Q,r);
    cor[s] ← preto;
Retorne d, π;
```



Q = | **w** | | |

Busca-em-largura (G, s)

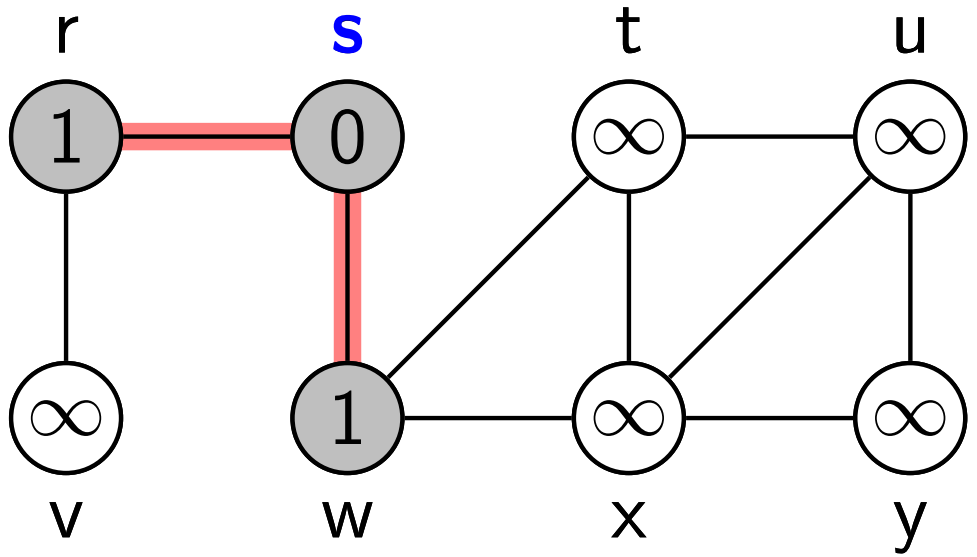
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← s;
    Para cada r ∈ Adj[s]
        Se cor[r] = branco;
            cor[r] ← cinza;
▷               d[r] ← d[s] + 1;
            π[r] ← s;
            ENQUEUE (Q,r);
    cor[s] ← preto;
Retorne d, π;
```



Q = | **w** | | |

Busca-em-largura (G, s)

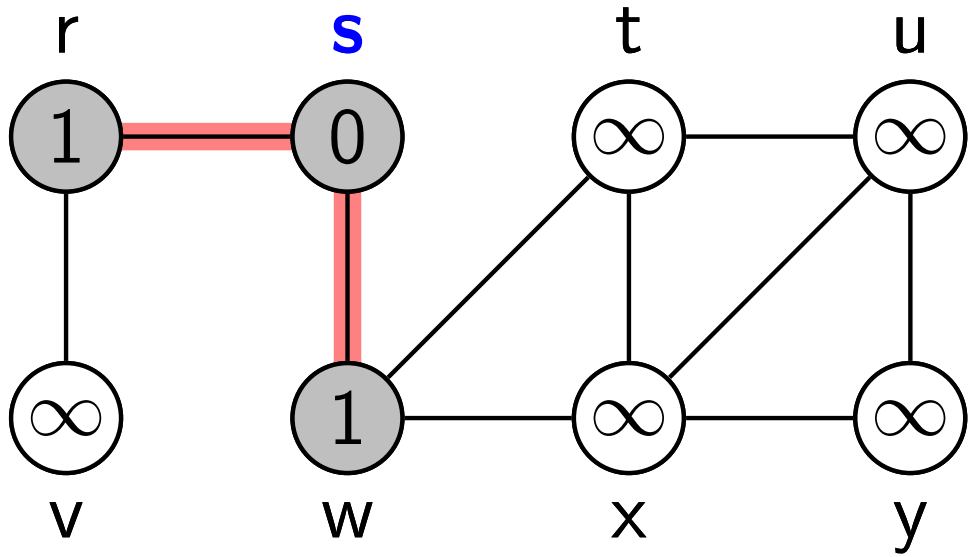
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ s;
    Para cada r $\in Adj$[**s**]
        Se cor[**r**] = branco;
            cor[**r**] $\leftarrow$ cinza;
            d[**r**] $\leftarrow$ d[**s**] + 1;
$\triangleright$ π[**r**] $\leftarrow$ **s**;
            ENQUEUE (Q,**r**);
    cor[**s**] $\leftarrow$ preto;
Retorne d, π;



Q = | **w** | | |

Busca-em-largura (G, s)

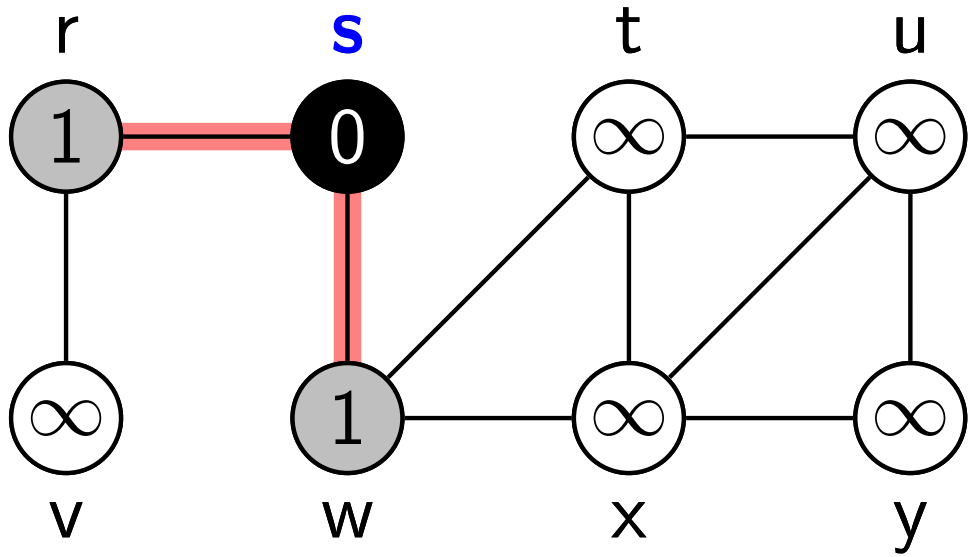
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ s;
    Para cada r $\in Adj$[**s**]
        Se cor[**r**] = branco;
            cor[**r**] $\leftarrow$ cinza;
            d[**r**] $\leftarrow$ d[**s**] + 1;
            π[**r**] $\leftarrow$ **s**;
$\triangleright$ ENQUEUE (Q,**r**);
    cor[**s**] $\leftarrow$ preto;
Retorne d, π;



Q = | **w** | **r** | |

Busca-em-largura (G, s)

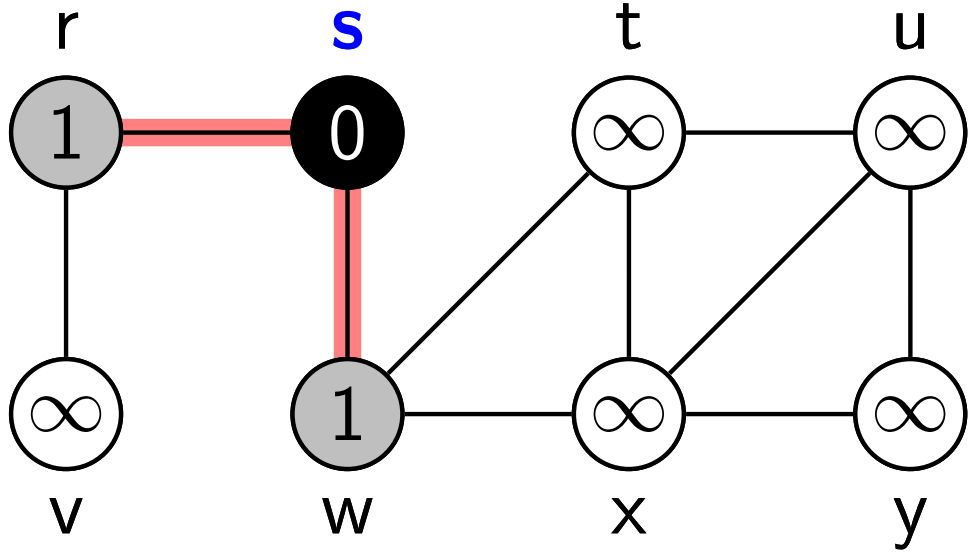
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ s;
    Para cada r $\in Adj$[**s**]
        Se cor[**r**] = branco;
            cor[**r**] $\leftarrow$ cinza;
            d[**r**] $\leftarrow$ d[**s**] + 1;
            π[**r**] $\leftarrow$ **s**;
            ENQUEUE (Q,**r**);
$\triangleright$ cor[**s**] $\leftarrow$ preto;
Retorne d, π;



Q = | **w** | **r** | |

Busca-em-largura (G, s)

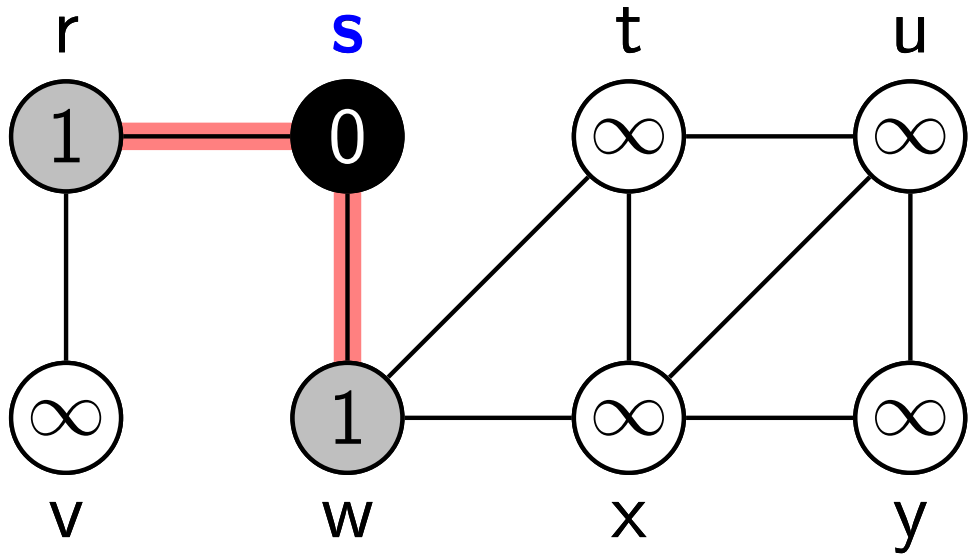
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada t ∈ Adj[w]
            Se cor[t] = branco;
                  cor[t] ← cinza;
                  d[t] ← d[w] + 1;
                  π[t] ← w;
                  ENQUEUE (Q,t);
      cor[w] ← preto;
Retorne d, π;
```



Q = | **w** | **r** | |

Busca-em-largura (G, s)

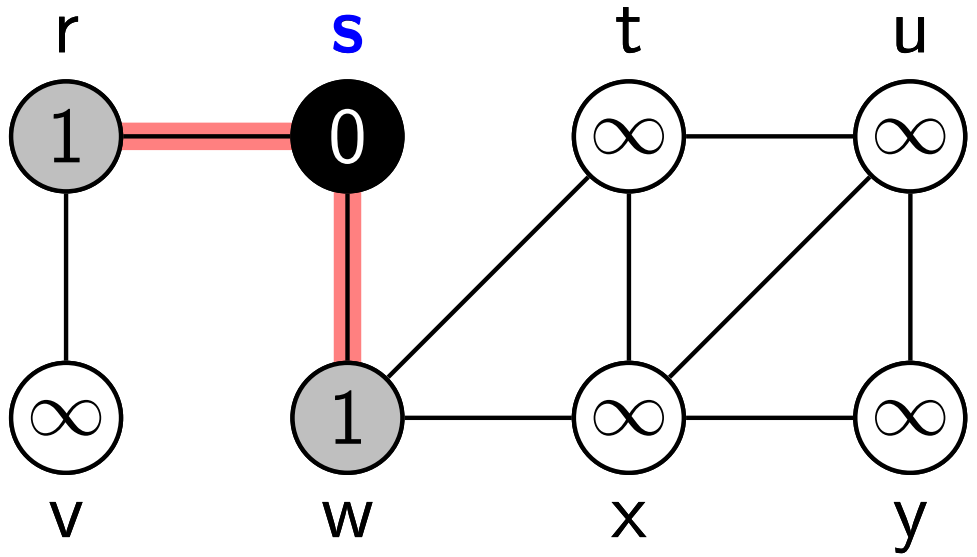
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← w;
    Para cada t ∈ Adj[w]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[w] + 1;
            π[t] ← w;
            ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | | |

Busca-em-largura (G, s)

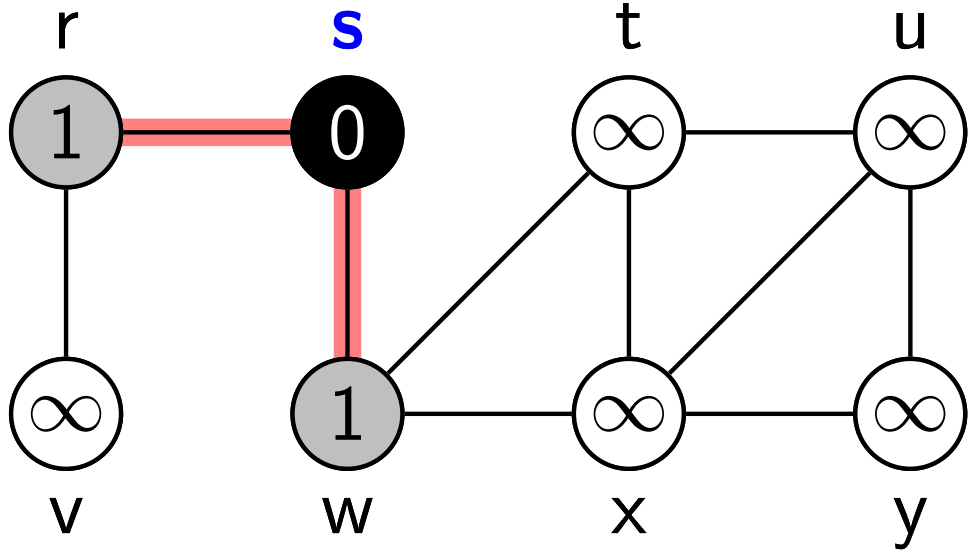
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **w**;
▷ **Para cada t** $\in Adj$[**w**]
        Se cor[**t**] = branco;
            cor[**t**] $\leftarrow$ cinza;
            d[**t**] $\leftarrow$ d[**w**] + 1;
            π[**t**] $\leftarrow$ **w**;
            ENQUEUE (Q,**t**);
    cor[**w**] $\leftarrow$ preto;
Retorne d, π;



Q = | **r** | | |

Busca-em-largura (G, s)

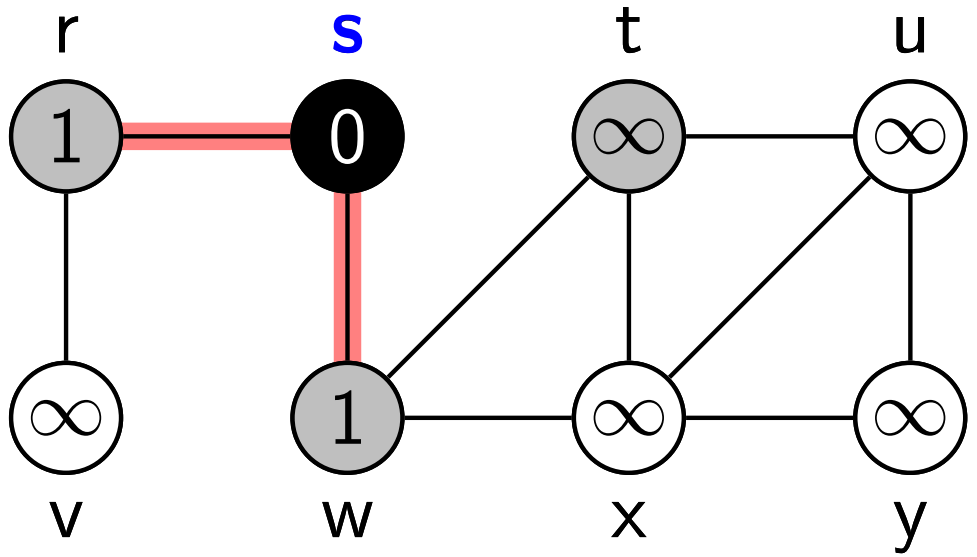
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada t ∈ Adj[w]
▷       Se cor[t] = branco;
          cor[t] ← cinza;
          d[t] ← d[w] + 1;
          π[t] ← w;
          ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | | |

Busca-em-largura (G, s)

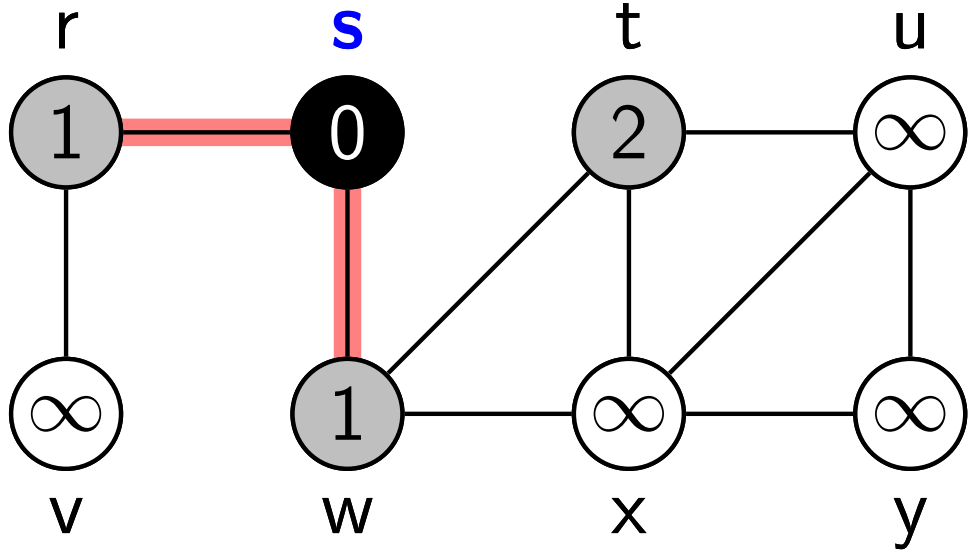
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada t ∈ Adj[w]
        Se cor[t] = branco;
▷                 cor[t] ← cinza;
                  d[t] ← d[w] + 1;
                  π[t] ← w;
                  ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | | |

Busca-em-largura (G, s)

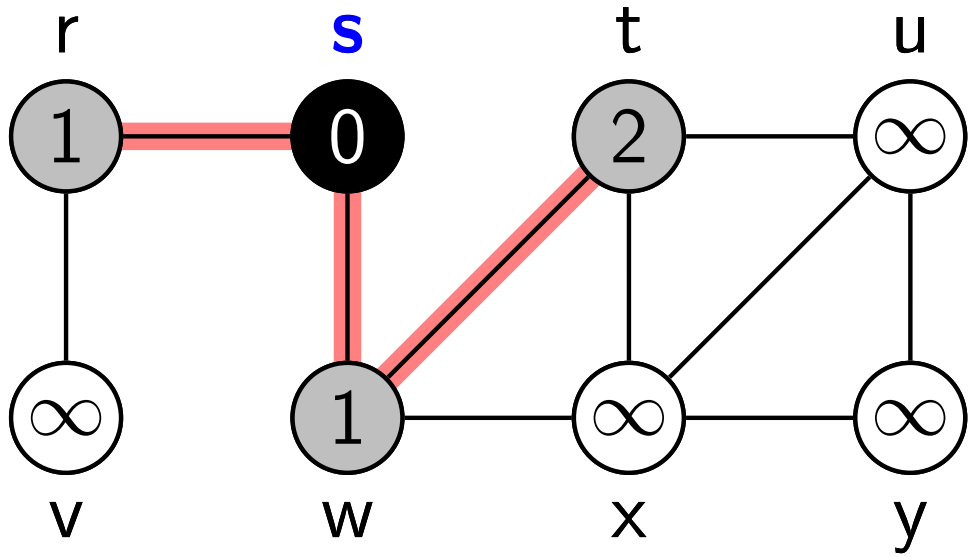
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada t ∈ Adj[w]
        Se cor[t] = branco;
            cor[t] ← cinza;
▷               d[t] ← d[w] + 1;
            π[t] ← w;
            ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | | |

Busca-em-largura (G, s)

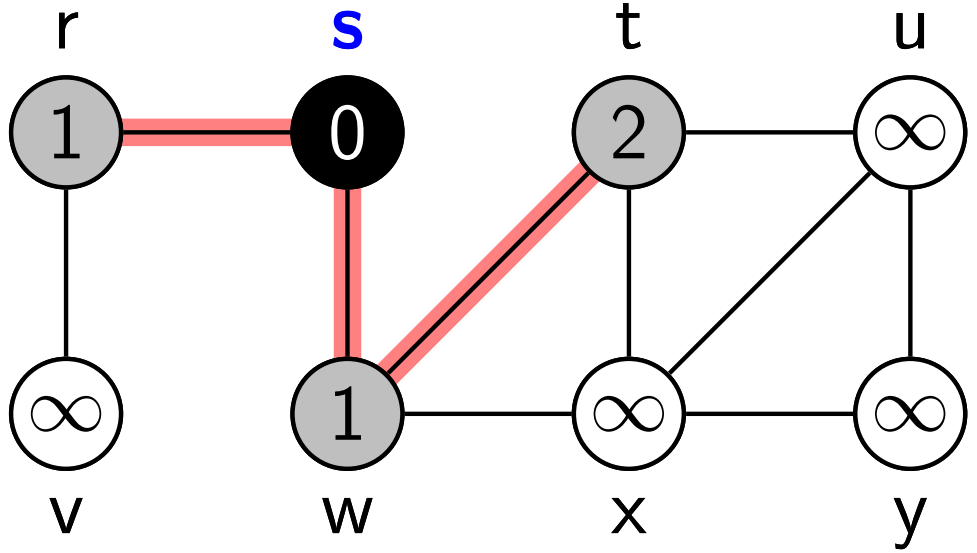
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada t ∈ Adj[w]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[w] + 1;
▷           π[t] ← w;
            ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | | |

Busca-em-largura (G, s)

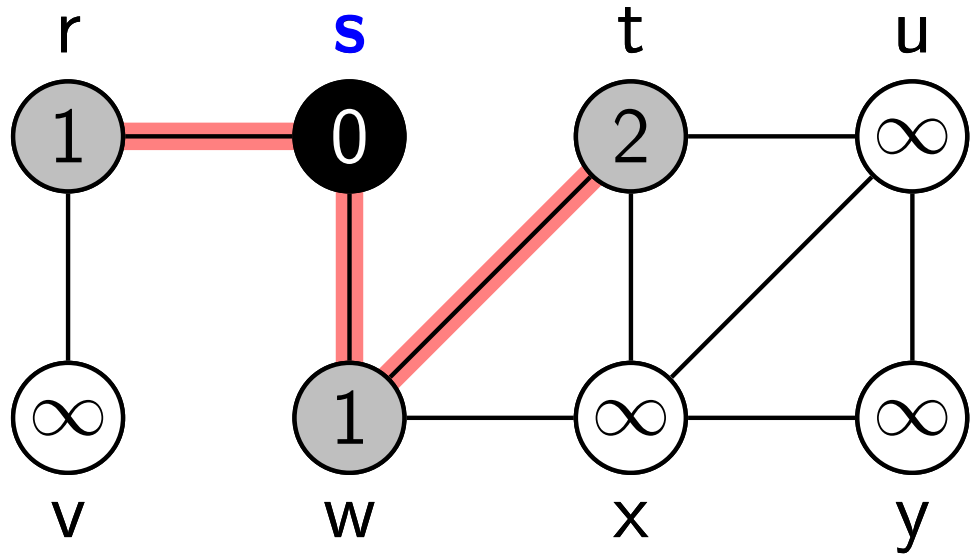
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada t ∈ Adj[w]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[w] + 1;
            π[t] ← w;
▷               ENQUEUE (Q,t);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

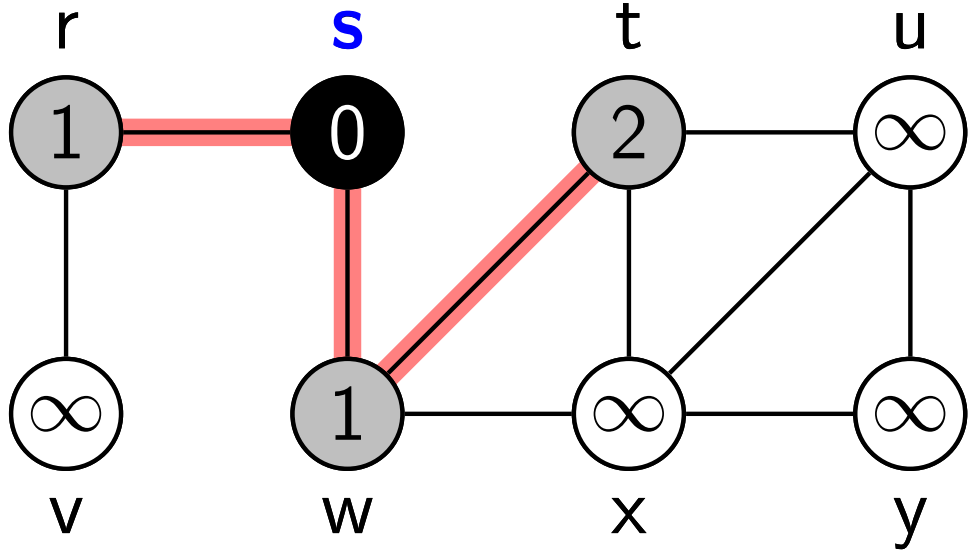
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
▷   Para cada x ∈ Adj[w]
      Se cor[x] = branco;
          cor[x] ← cinza;
          d[x] ← d[w] + 1;
          π[x] ← w;
          ENQUEUE (Q,x);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

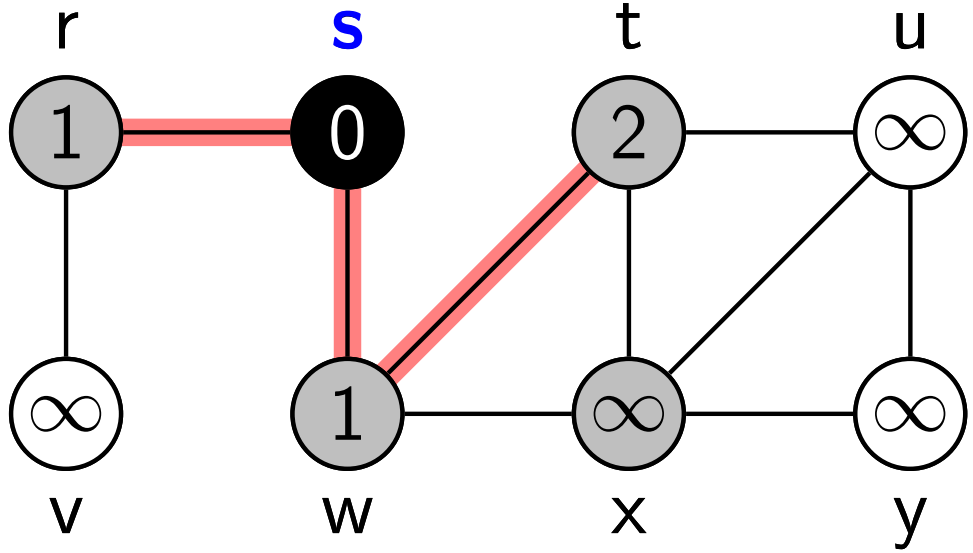
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada x ∈ Adj[w]
▷       Se cor[x] = branco;
          cor[x] ← cinza;
          d[x] ← d[w] + 1;
          π[x] ← w;
          ENQUEUE (Q,x);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

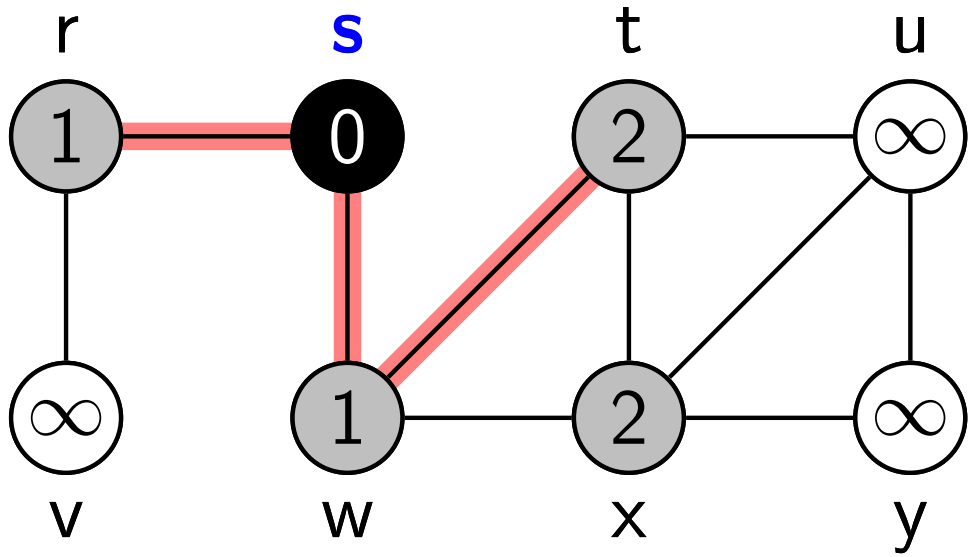
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada x ∈ Adj[w]
        Se cor[x] = branco;
▷                 cor[x] ← cinza;
                  d[x] ← d[w] + 1;
                  π[x] ← w;
                  ENQUEUE (Q,x);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

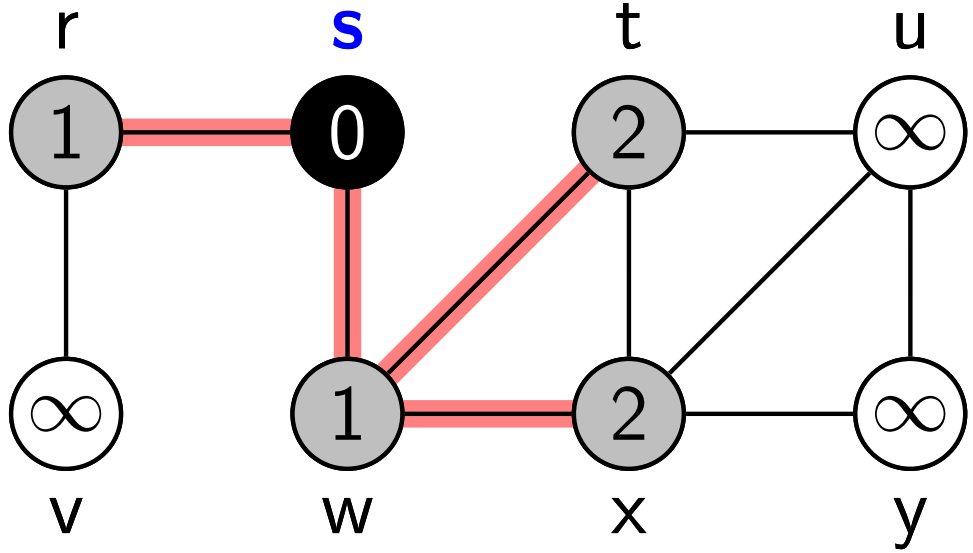
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada x ∈ Adj[w]
        Se cor[x] = branco;
            cor[x] ← cinza;
▷               d[x] ← d[w] + 1;
            π[x] ← w;
            ENQUEUE (Q,x);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

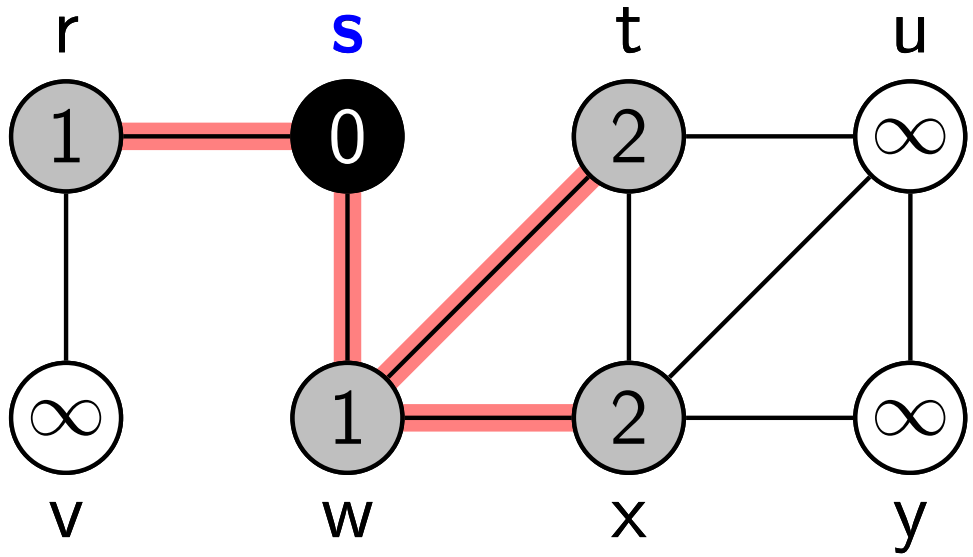
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada x ∈ Adj[w]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[w] + 1;
▷               π[x] ← w;
            ENQUEUE (Q,x);
    cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | |

Busca-em-largura (G, s)

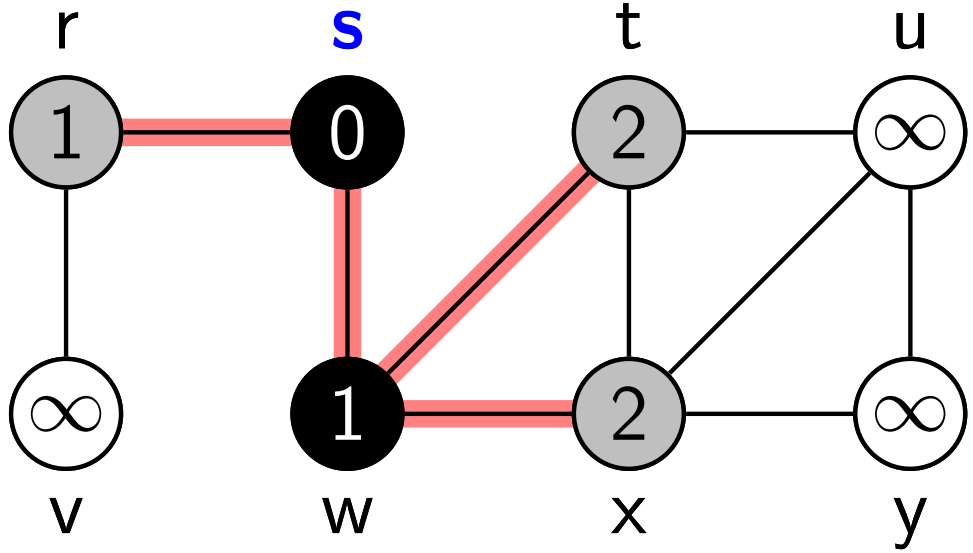
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ w;
    Para cada x $\in Adj$[**w**]
        Se cor[**x**] = branco;
            cor[**x**] $\leftarrow$ cinza;
            d[**x**] $\leftarrow$ d[**w**] + 1;
            π[**x**] $\leftarrow$ **w**;
$\triangleright$ ENQUEUE (Q,**x**);
    cor[**w**] $\leftarrow$ preto;
Retorne d, π;



Q = | **r** | **t** | **x** |

Busca-em-largura (G, s)

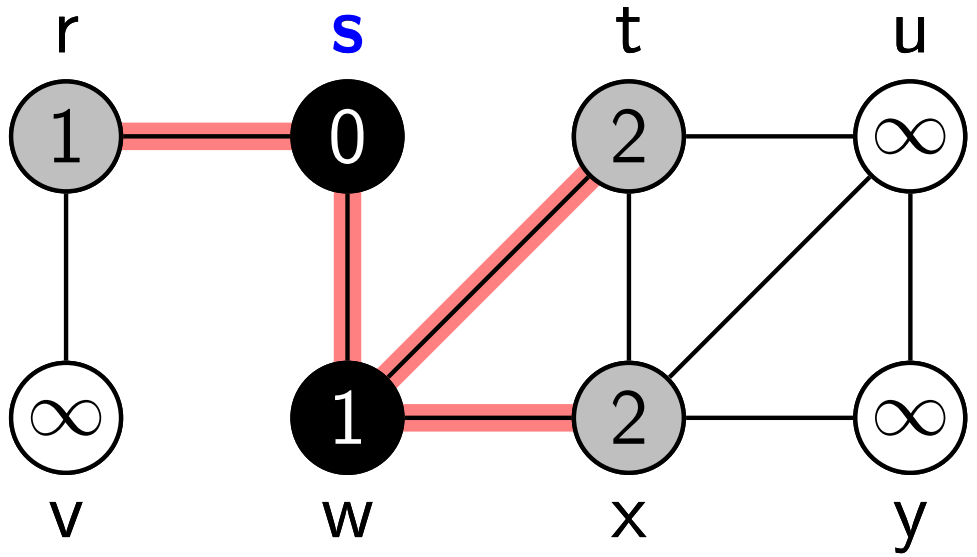
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← w;
    Para cada x ∈ Adj[w]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[w] + 1;
            π[x] ← w;
            ENQUEUE (Q,x);
▷   cor[w] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | **x** |

Busca-em-largura (G, s)

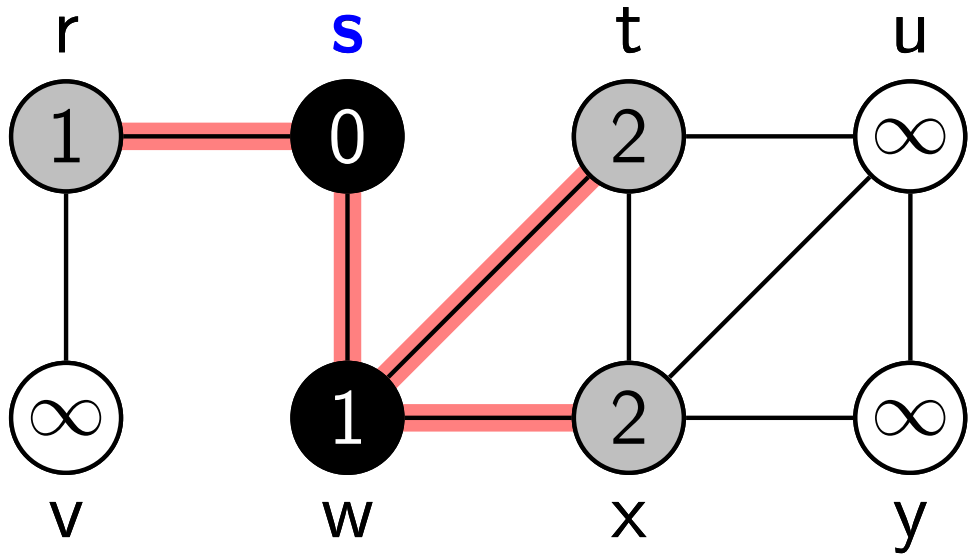
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada s ∈ Adj[r]
          Se cor[s] = branco;
              cor[s] ← cinza;
              d[s] ← d[r] + 1;
              π[s] ← r;
              ENQUEUE (Q,s);
      cor[r] ← preto;
Retorne d, π;
```



Q = | **r** | **t** | **x** |

Busca-em-largura (G, s)

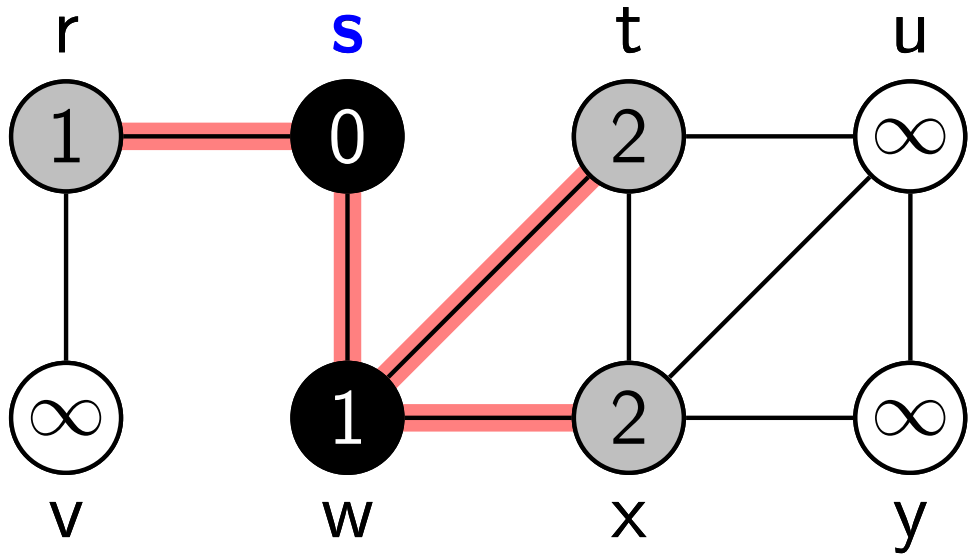
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← r;
    Para cada s ∈ Adj[r]
        Se cor[s] = branco;
            cor[s] ← cinza;
            d[s] ← d[r] + 1;
            π[s] ← r;
            ENQUEUE (Q,s);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

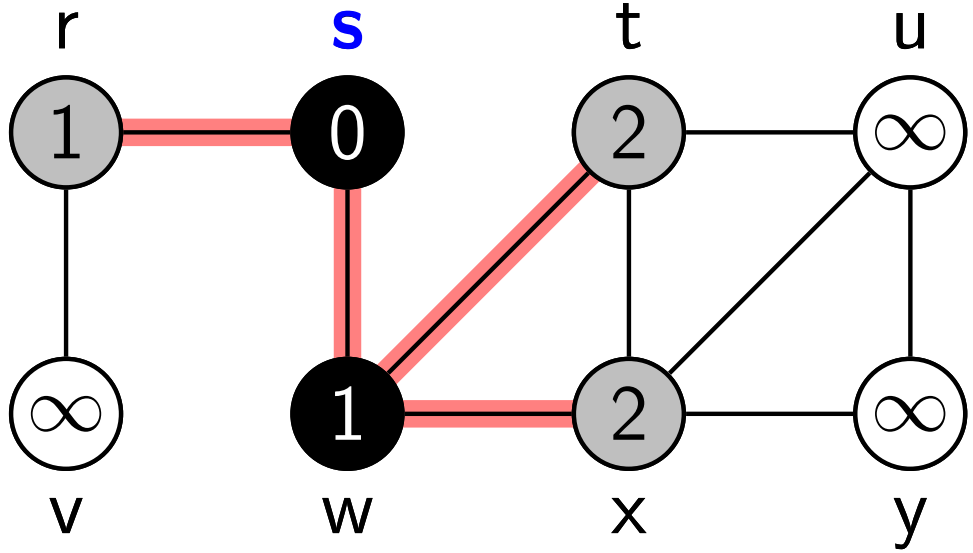
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **r**;
▷   **Para cada s** $\in Adj$[**r**]
        Se cor[**s**] = branco;
            cor[**s**] $\leftarrow$ cinza;
            d[**s**] $\leftarrow$ d[**r**] + 1;
            π[**s**] $\leftarrow$ **r**;
            ENQUEUE (Q,**s**);
    cor[**r**] $\leftarrow$ preto;
Retorne d, π;



Q = | **t** | **x** | |

Busca-em-largura (G, s)

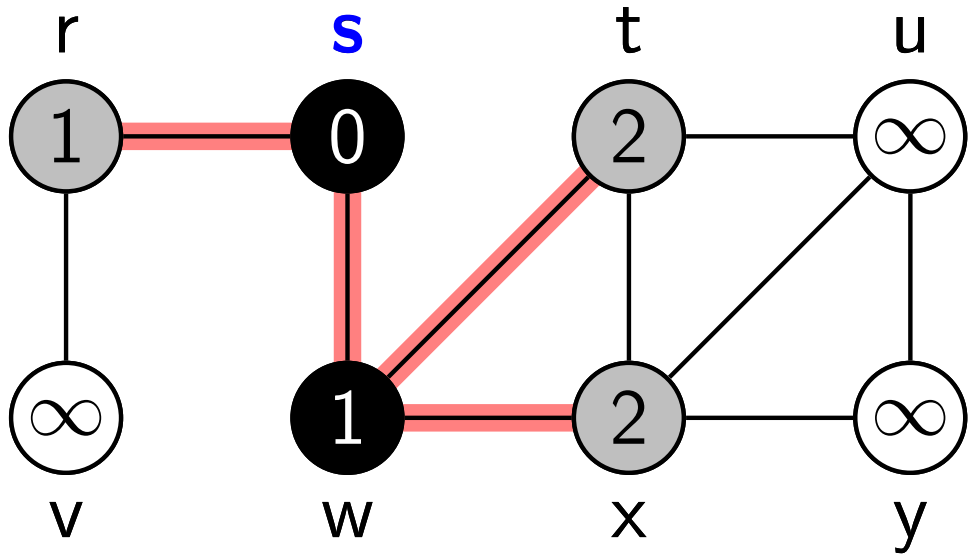
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada s ∈ Adj[r]
▷       Se cor[s] = branco;
          cor[s] ← cinza;
          d[s] ← d[r] + 1;
          π[s] ← r;
          ENQUEUE (Q,s);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

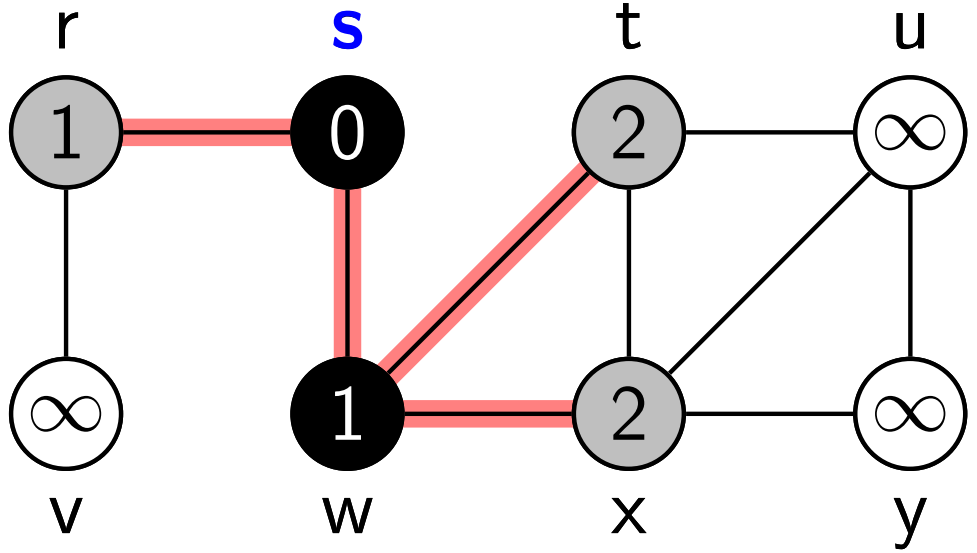
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
▷   Para cada v ∈ Adj[r]
      Se cor[v] = branco;
          cor[v] ← cinza;
          d[v] ← d[r] + 1;
          π[v] ← r;
          ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

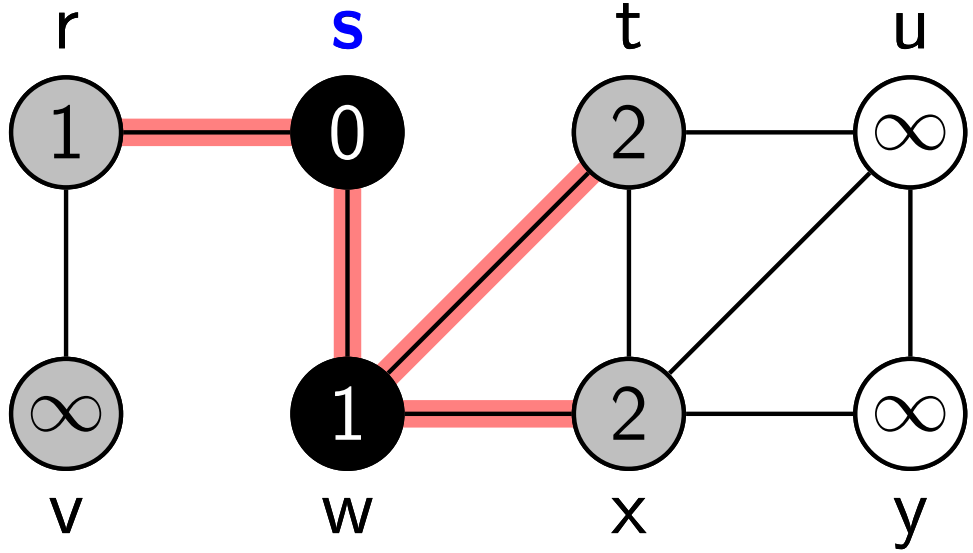
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
▷       Se cor[v] = branco;
          cor[v] ← cinza;
          d[v] ← d[r] + 1;
          π[v] ← r;
          ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

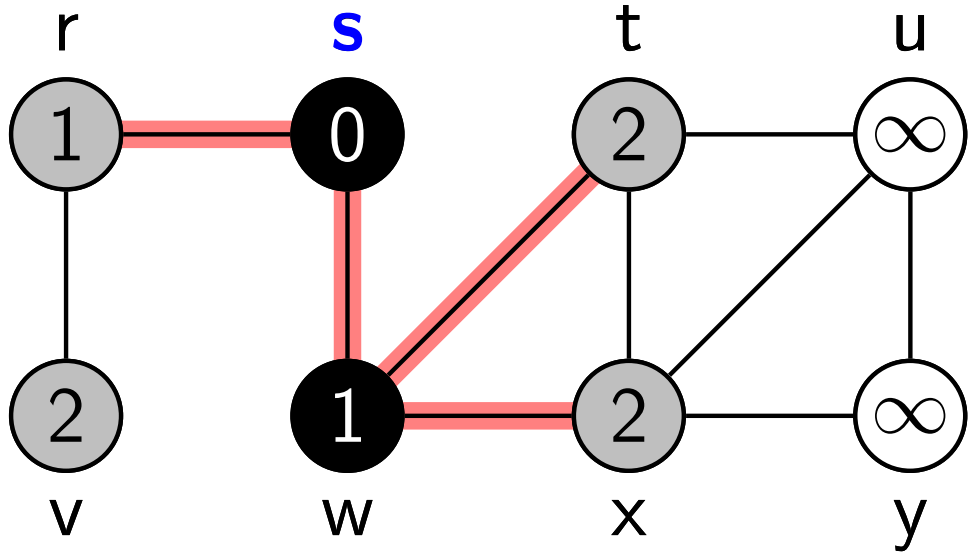
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
        Se cor[v] = branco;
▷                 cor[v] ← cinza;
                  d[v] ← d[r] + 1;
                  π[v] ← r;
                  ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

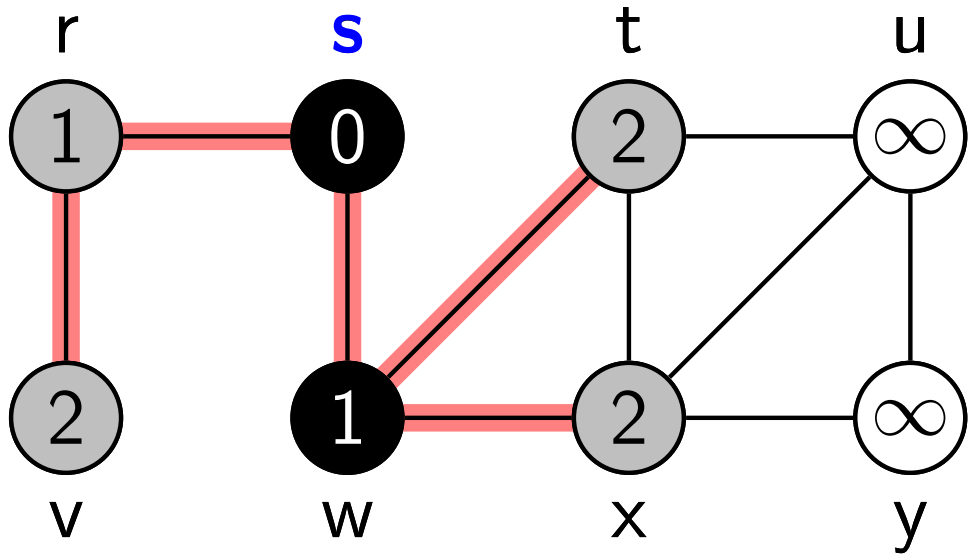
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
        Se cor[v] = branco;
            cor[v] ← cinza;
▷               d[v] ← d[r] + 1;
            π[v] ← r;
            ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

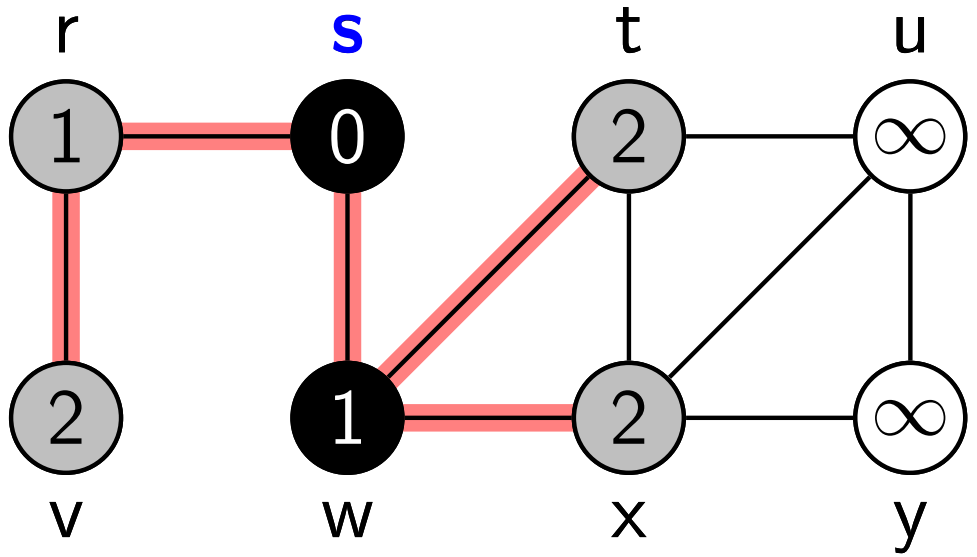
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
        Se cor[v] = branco;
            cor[v] ← cinza;
            d[v] ← d[r] + 1;
▷               π[v] ← r;
            ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | |

Busca-em-largura (G, s)

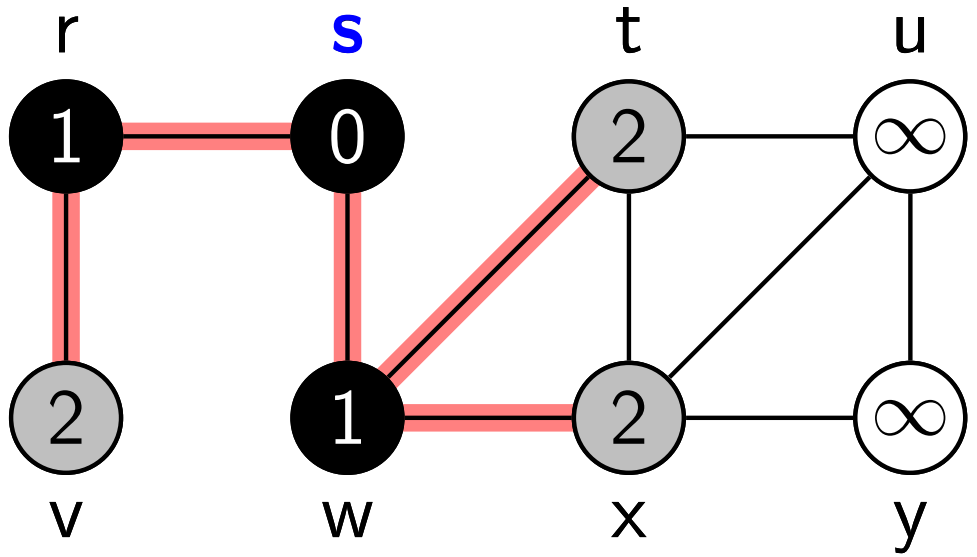
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
        Se cor[v] = branco;
            cor[v] ← cinza;
            d[v] ← d[r] + 1;
            π[v] ← r;
▷               ENQUEUE (Q,v);
    cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | **v** |

Busca-em-largura (G, s)

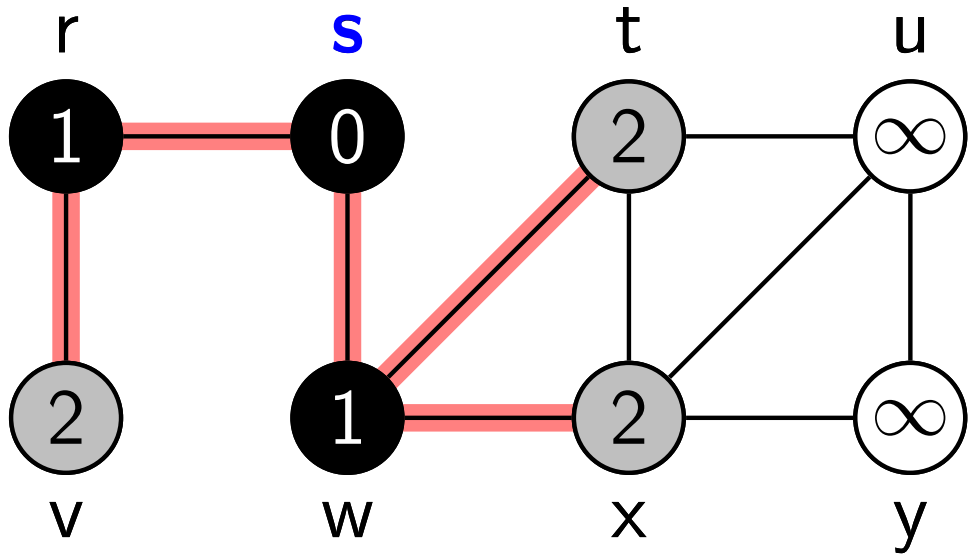
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← r;
    Para cada v ∈ Adj[r]
        Se cor[v] = branco;
            cor[v] ← cinza;
            d[v] ← d[r] + 1;
            π[v] ← r;
            ENQUEUE (Q,v);
▷   cor[r] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | **v** |

Busca-em-largura (G, s)

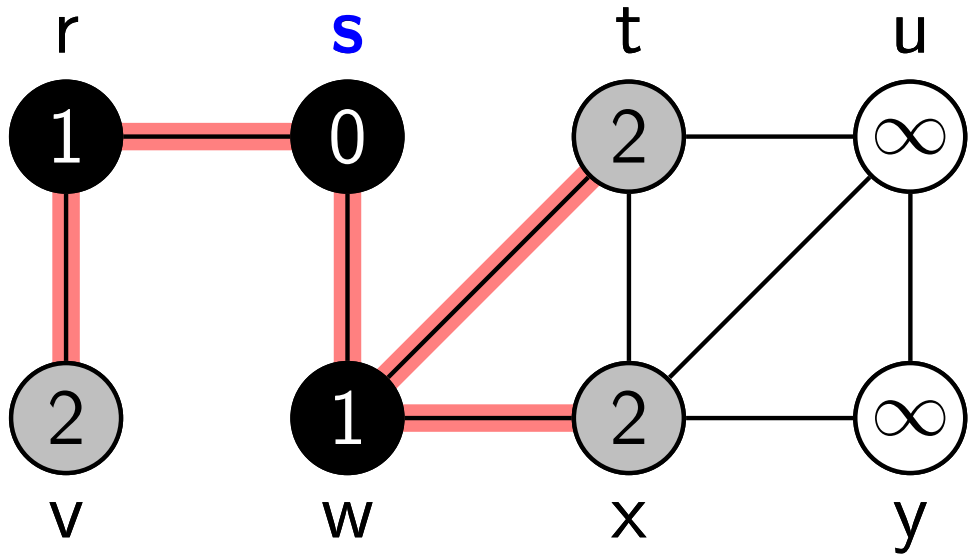
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada u ∈ Adj[t]
            Se cor[u] = branco;
                  cor[u] ← cinza;
                  d[u] ← d[t] + 1;
                  π[u] ← t;
                  ENQUEUE (Q,u);
      cor[t] ← preto;
Retorne d, π;
```



Q = | **t** | **x** | **v** |

Busca-em-largura (G, s)

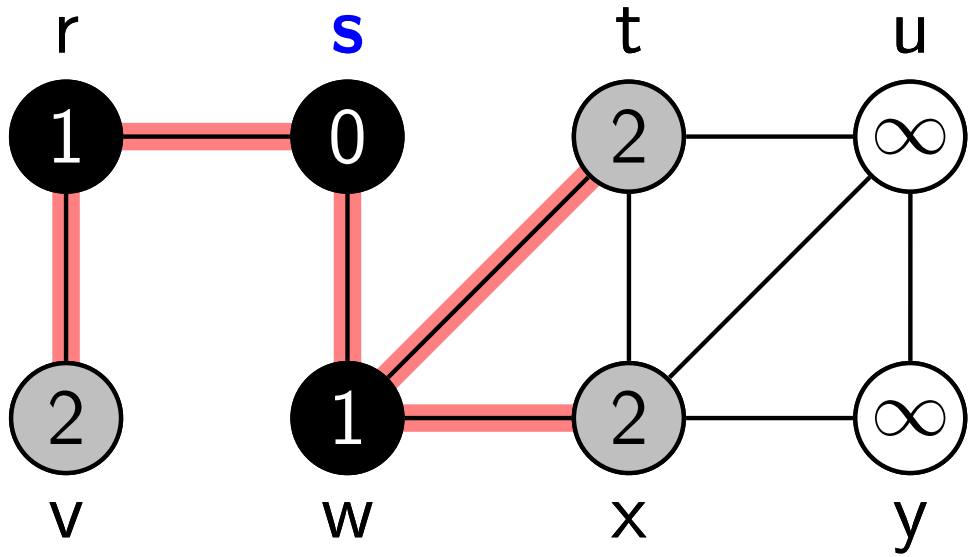
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← t;
    Para cada u ∈ Adj[t]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[t] + 1;
            π[u] ← t;
            ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | **x** | **v** | |

Busca-em-largura (G, s)

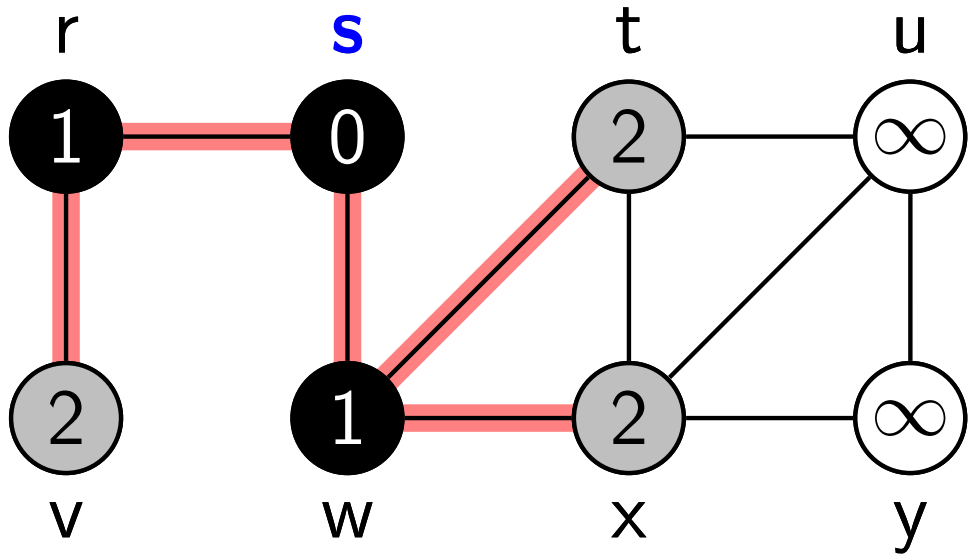
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **t**;
▷   **Para cada u** $\in Adj$[**t**]
        Se cor[**u**] = branco;
            cor[**u**] $\leftarrow$ cinza;
            d[**u**] $\leftarrow$ d[**t**] + 1;
            π[**u**] $\leftarrow$ **t**;
            ENQUEUE (Q,**u**);
    cor[**t**] $\leftarrow$ preto;
Retorne d, π;



Q = | **x** | **v** | |

Busca-em-largura (G, s)

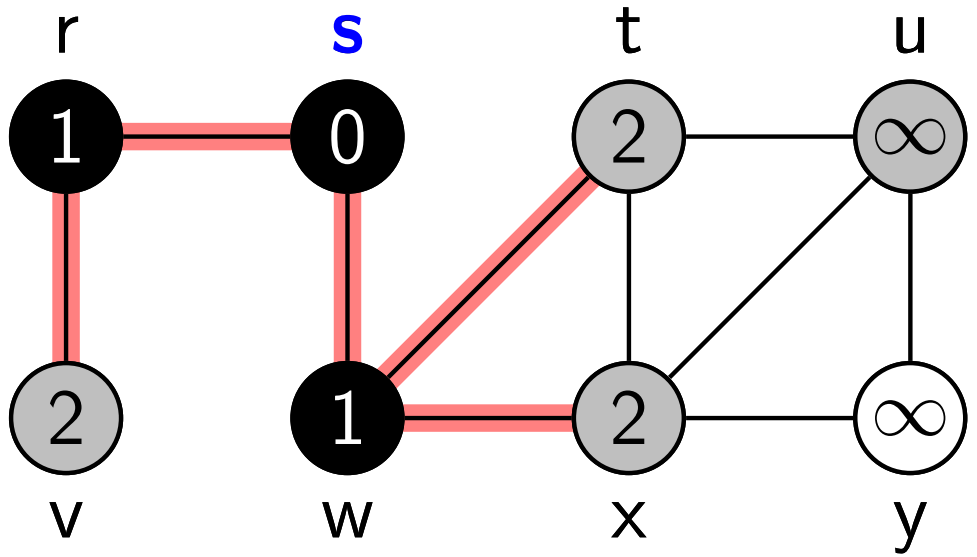
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada u ∈ Adj[t]
▷       Se cor[u] = branco;
          cor[u] ← cinza;
          d[u] ← d[t] + 1;
          π[u] ← t;
          ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | **x** | **v** | |

Busca-em-largura (G, s)

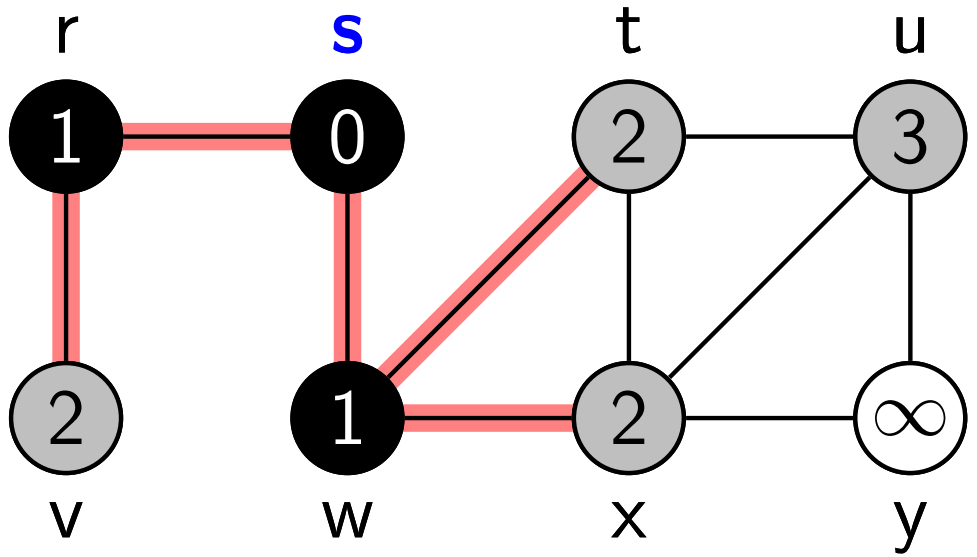
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada u ∈ Adj[t]
        Se cor[u] = branco;
▷                 cor[u] ← cinza;
                  d[u] ← d[t] + 1;
                  π[u] ← t;
                  ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | **x** | **v** | |

Busca-em-largura (G, s)

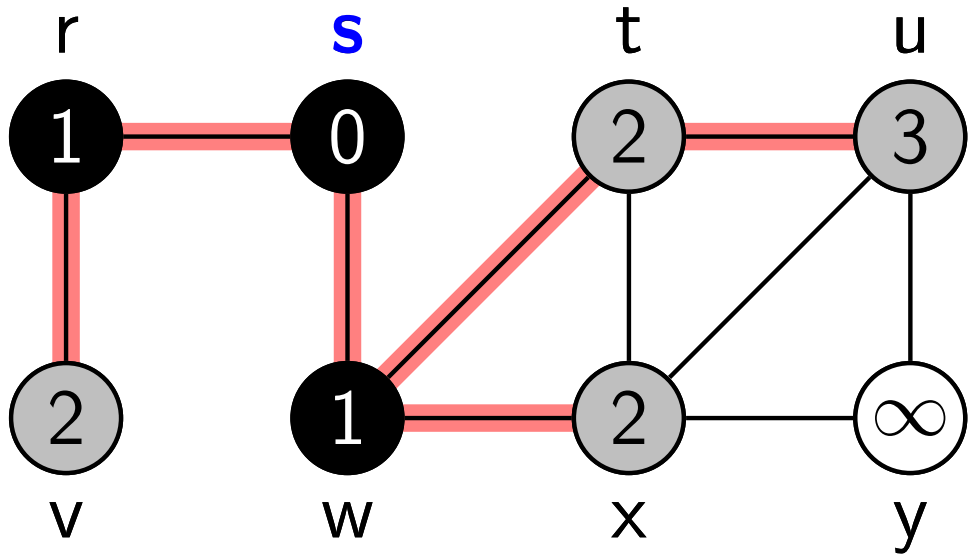
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada u ∈ Adj[t]
        Se cor[u] = branco;
            cor[u] ← cinza;
▷               d[u] ← d[t] + 1;
            π[u] ← t;
            ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | x | v | |

Busca-em-largura (G, s)

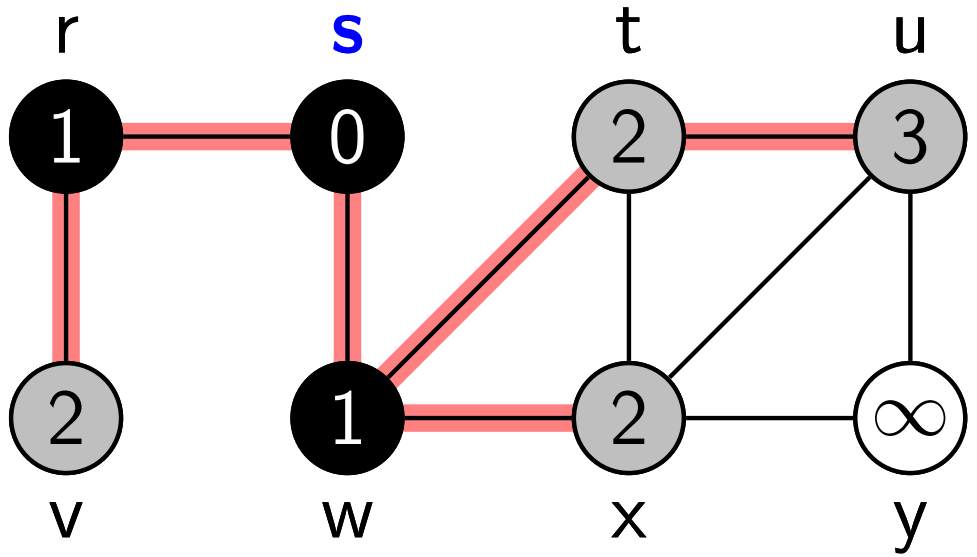
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada u ∈ Adj[t]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[t] + 1;
▷           π[u] ← t;
            ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | x | v | |

Busca-em-largura (G, s)

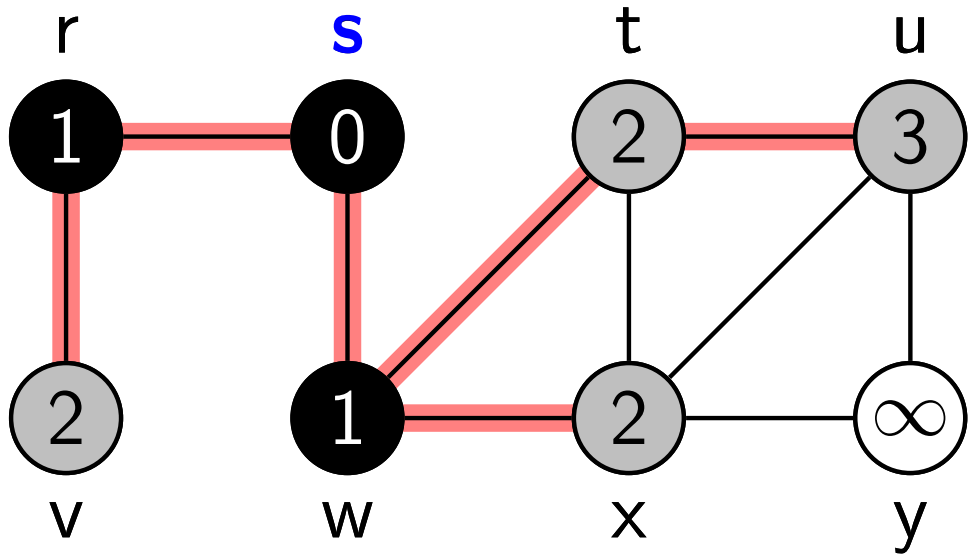
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada u ∈ Adj[t]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[t] + 1;
            π[u] ← t;
▷               ENQUEUE (Q,u);
    cor[t] ← preto;
Retorne d, π;
```



Q = | x | v | u |

Busca-em-largura (G, s)

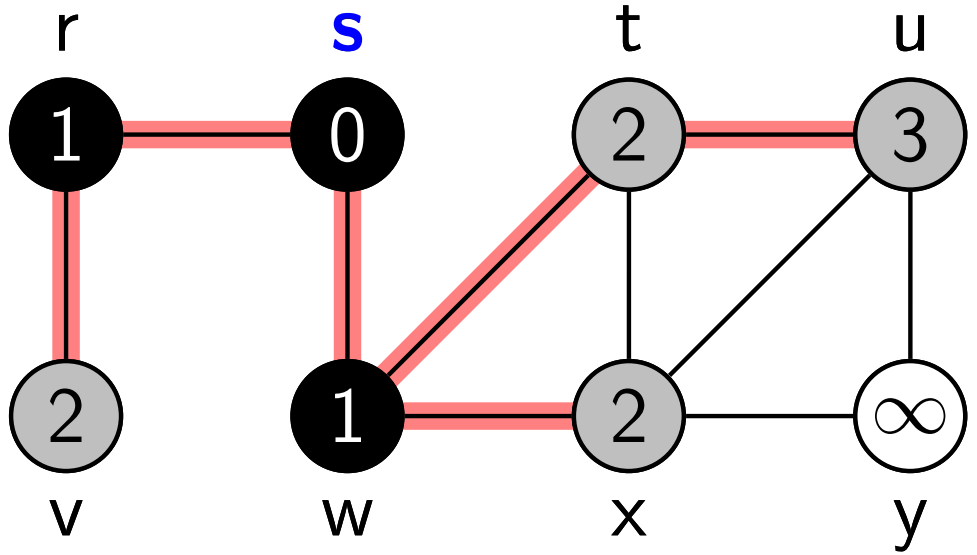
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
▷     Para cada x ∈ Adj[t]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[t] + 1;
            π[x] ← t;
            ENQUEUE (Q,x);
    cor[t] ← preto;
Retorne d, π;
```



Q = | **x** | **v** | **u** |

Busca-em-largura (G, s)

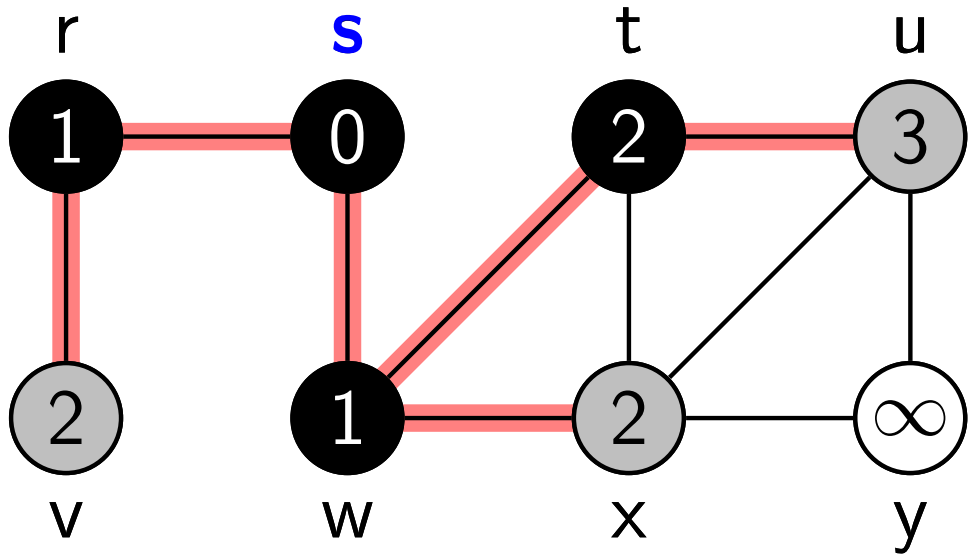
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada x ∈ Adj[t]
▷       Se cor[x] = branco;
          cor[x] ← cinza;
          d[x] ← d[t] + 1;
          π[x] ← t;
          ENQUEUE (Q,x);
    cor[t] ← preto;
Retorne d, π;
```



Q = | x | v | u |

Busca-em-largura (G, s)

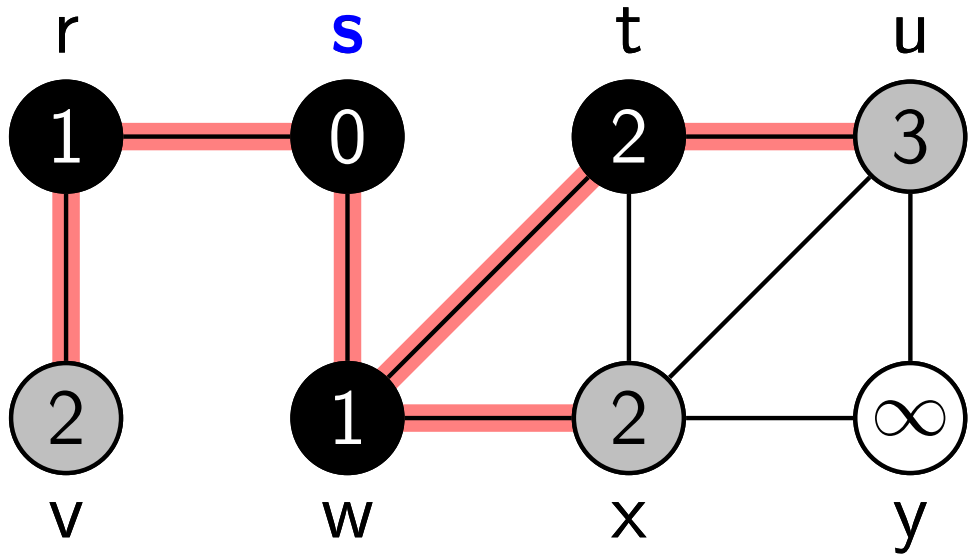
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← t;
    Para cada x ∈ Adj[t]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[t] + 1;
            π[x] ← t;
            ENQUEUE (Q,x);
▷   cor[t] ← preto;
Retorne d, π;
```



Q = | **x** | **v** | **u** |

Busca-em-largura (G, s)

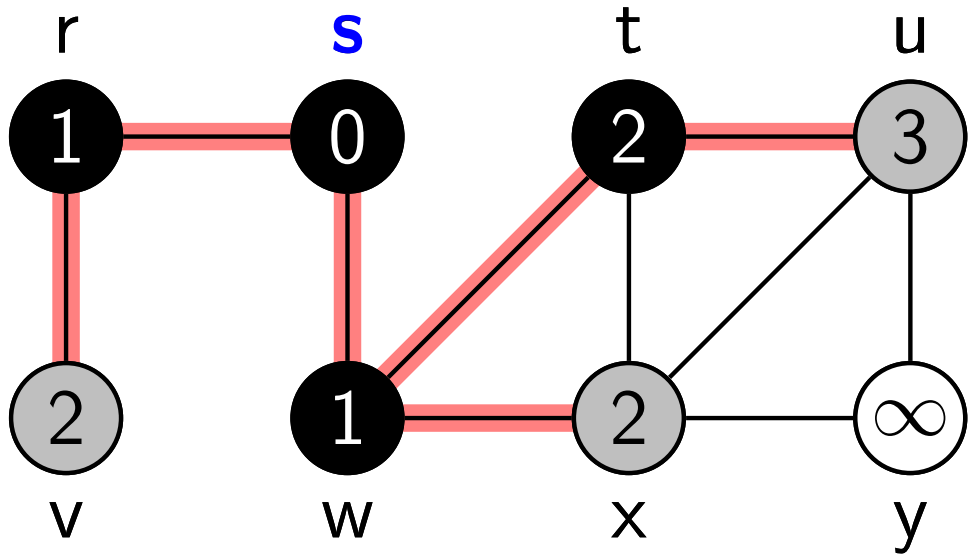
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada t ∈ Adj[x]
            Se cor[t] = branco;
                  cor[t] ← cinza;
                  d[t] ← d[x] + 1;
                  π[t] ← x;
                  ENQUEUE (Q,t);
      cor[x] ← preto;
Retorne d, π;
```



Q = | x | v | u |

Busca-em-largura (G, s)

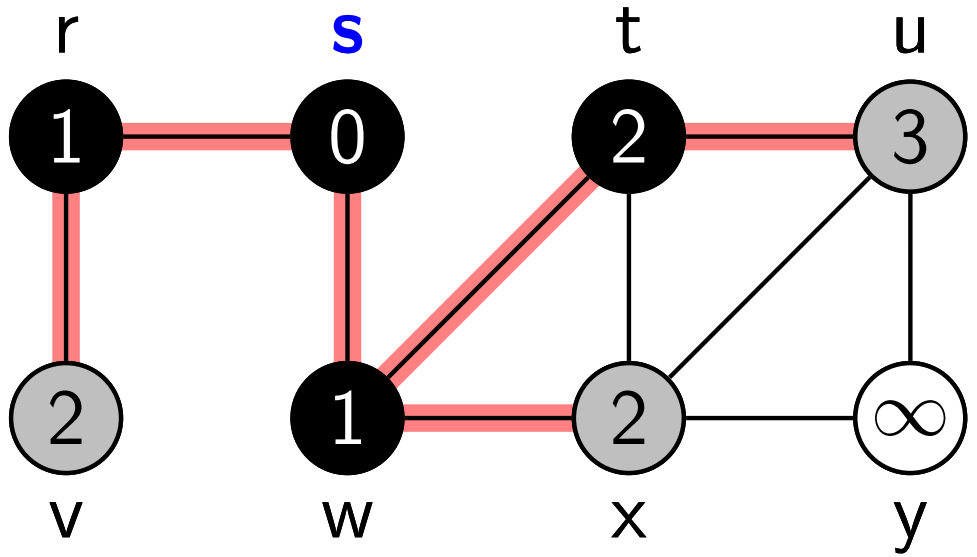
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← x;
    Para cada t ∈ Adj[x]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[x] + 1;
            π[t] ← x;
            ENQUEUE (Q,t);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

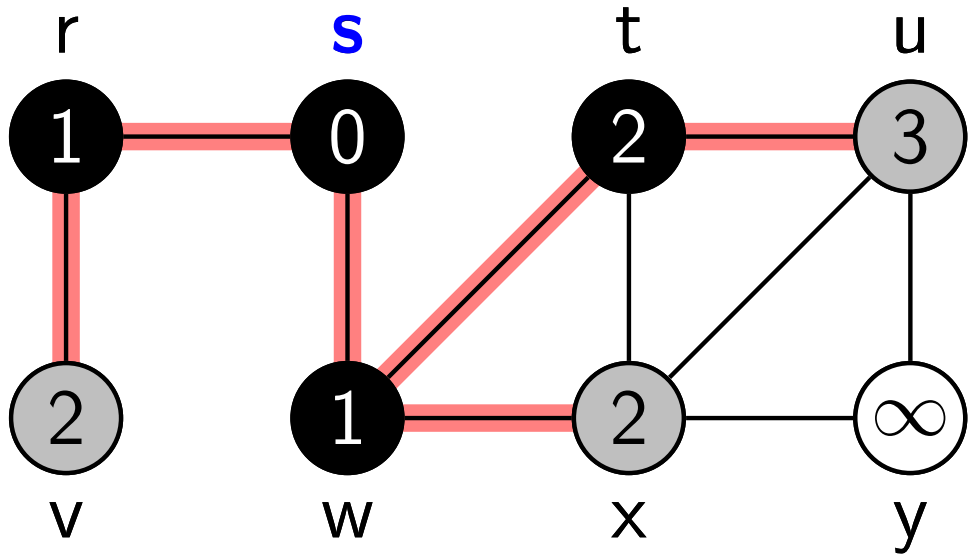
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
▷   Para cada t ∈ Adj[x]
      Se cor[t] = branco;
          cor[t] ← cinza;
          d[t] ← d[x] + 1;
          π[t] ← x;
          ENQUEUE (Q,t);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

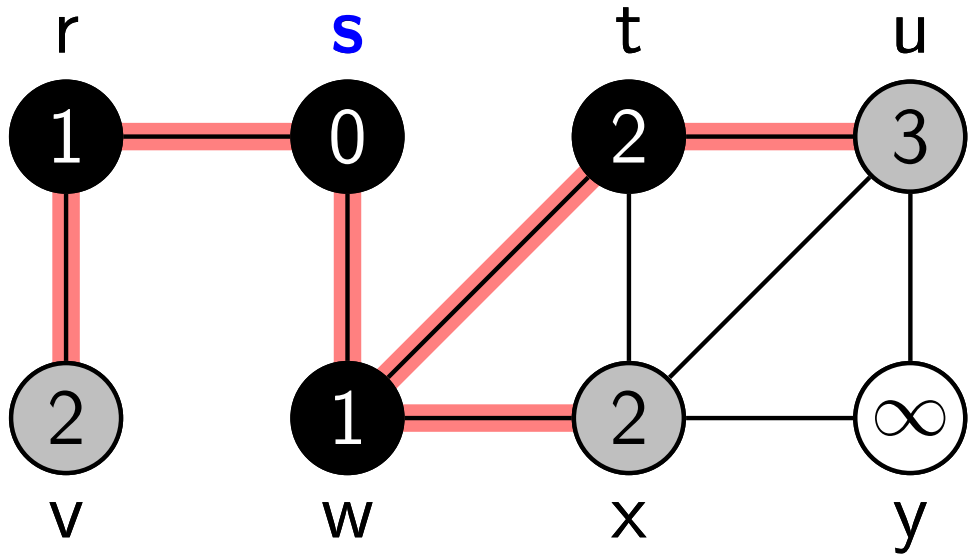
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada t ∈ Adj[x]
▷       Se cor[t] = branco;
          cor[t] ← cinza;
          d[t] ← d[x] + 1;
          π[t] ← x;
          ENQUEUE (Q,t);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **x**;
▷ **Para cada u** $\in Adj$[**x**]
        Se cor[**u**] $=$ branco;
            cor[**u**] $\leftarrow$ cinza;
            d[**u**] $\leftarrow$ d[**x**] $+ 1$;
            π[**u**] $\leftarrow$ **x**;
            ENQUEUE (Q,**u**);
    cor[**x**] $\leftarrow$ preto;
Retorne d, π;

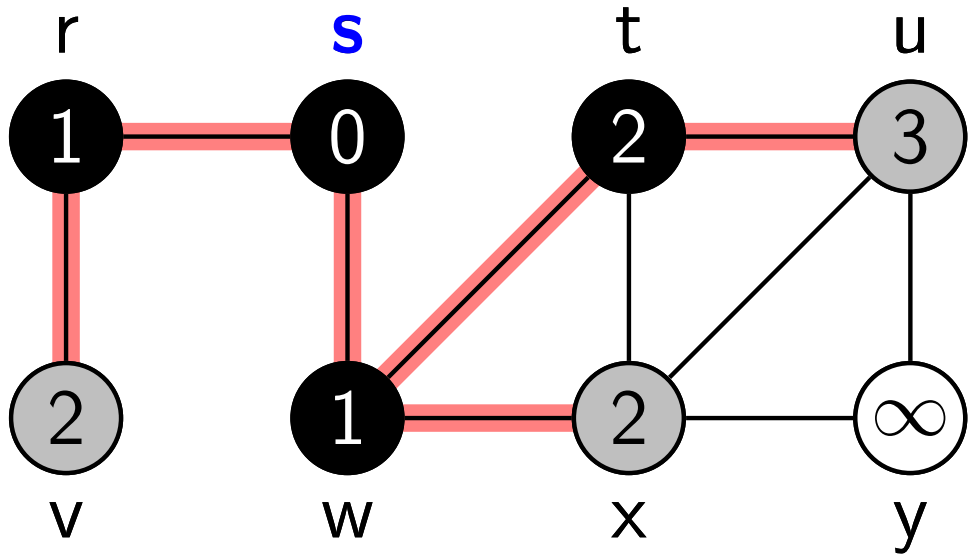


Q $=$

v	**u**	

Busca-em-largura (G, s)

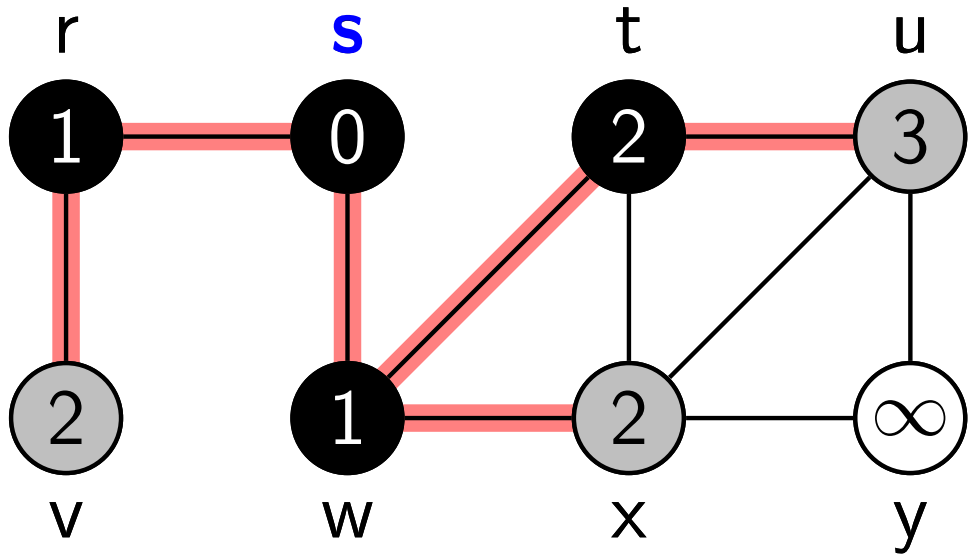
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada u ∈ Adj[x]
▷       Se cor[u] = branco;
          cor[u] ← cinza;
          d[u] ← d[x] + 1;
          π[u] ← x;
          ENQUEUE (Q,u);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

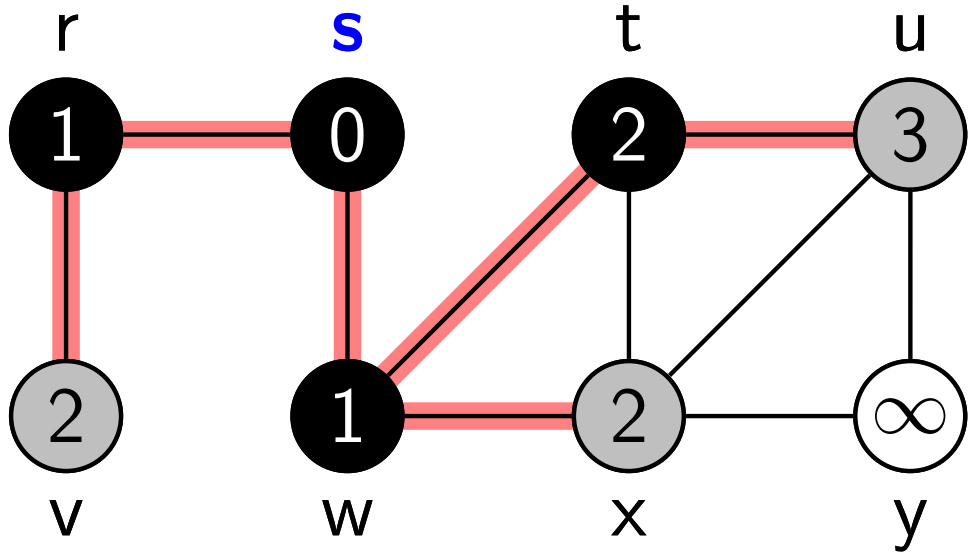
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
▷     Para cada y ∈ Adj[x]
        Se cor[y] = branco;
            cor[y] ← cinza;
            d[y] ← d[x] + 1;
            π[y] ← x;
            ENQUEUE (Q,y);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

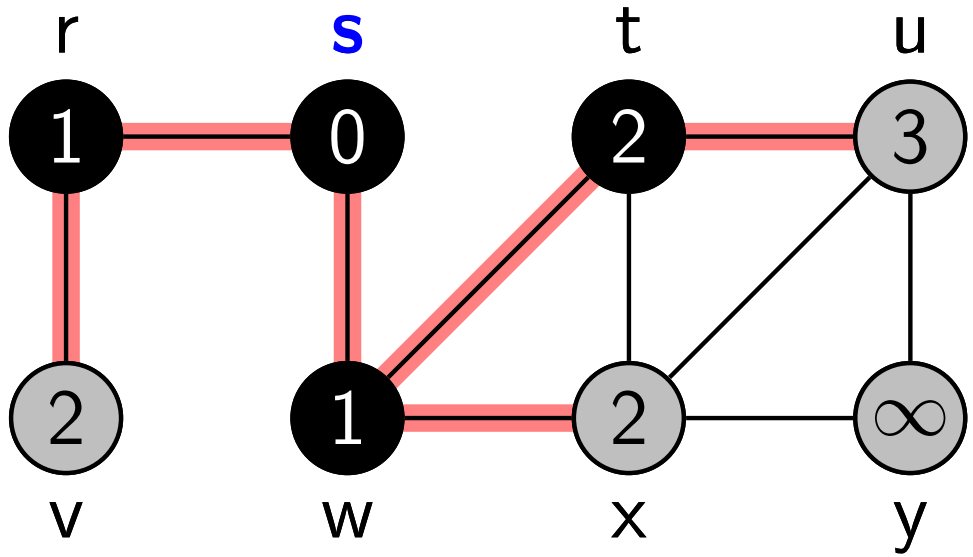
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada y ∈ Adj[x]
▷       Se cor[y] = branco;
          cor[y] ← cinza;
          d[y] ← d[x] + 1;
          π[y] ← x;
          ENQUEUE (Q,y);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

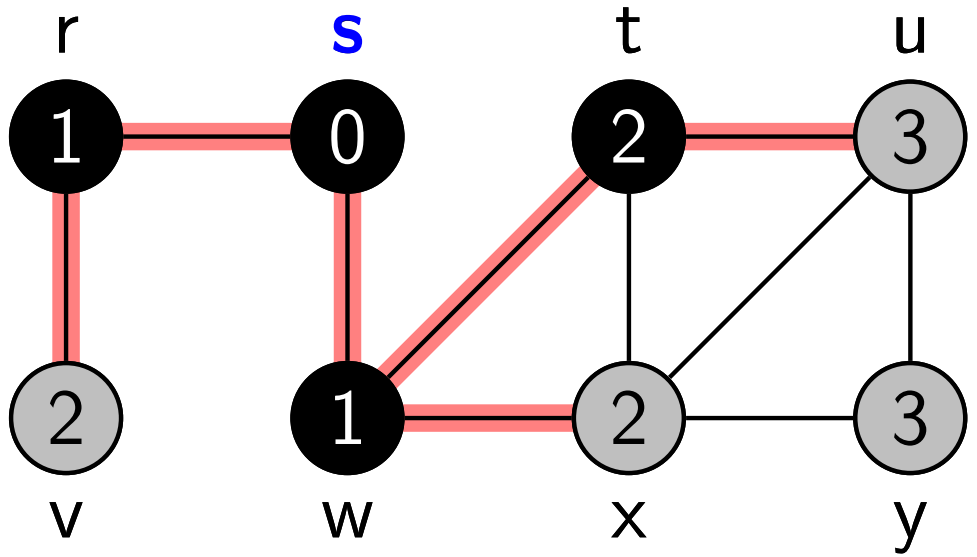
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **x**;
    Para cada **y** $\in Adj$[**x**]
        Se cor[**y**] = branco;
$\triangleright$ cor[**y**] $\leftarrow$ cinza;
d[**y**] $\leftarrow$ d[**x**] + 1;
π[**y**] $\leftarrow$ **x**;
ENQUEUE (Q,**y**);
    cor[**x**] $\leftarrow$ preto;
Retorne d, π;



Q = | **v** | **u** | |

Busca-em-largura (G, s)

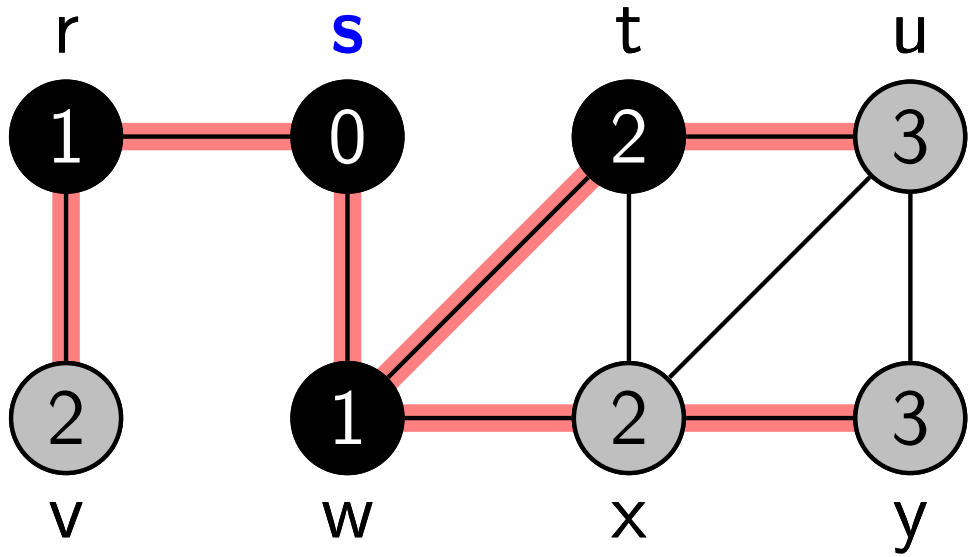
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada y ∈ Adj[x]
        Se cor[y] = branco;
            cor[y] ← cinza;
▷               d[y] ← d[x] + 1;
            π[y] ← x;
            ENQUEUE (Q,y);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | |

Busca-em-largura (G, s)

Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **x**;
    Para cada **y** $\in Adj$[**x**]
        Se cor[**y**] $=$ branco;
            cor[**y**] $\leftarrow$ cinza;
            d[**y**] $\leftarrow$ d[**x**] $+ 1$;
$\triangleright$ π[**y**] $\leftarrow$ **x**;
            ENQUEUE (Q,**y**);
    cor[**x**] $\leftarrow$ preto;
Retorne d, π;

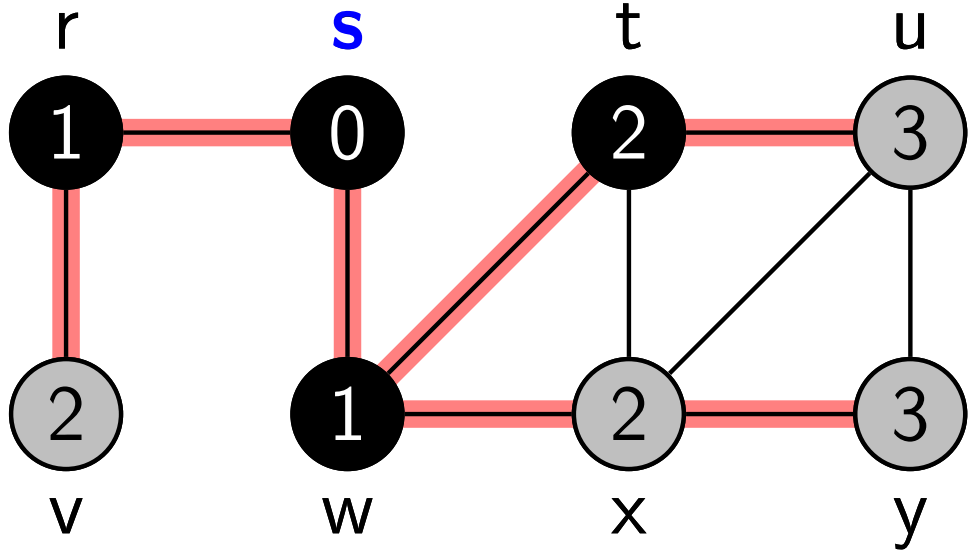


Q $=$

v	**u**	

Busca-em-largura (G, s)

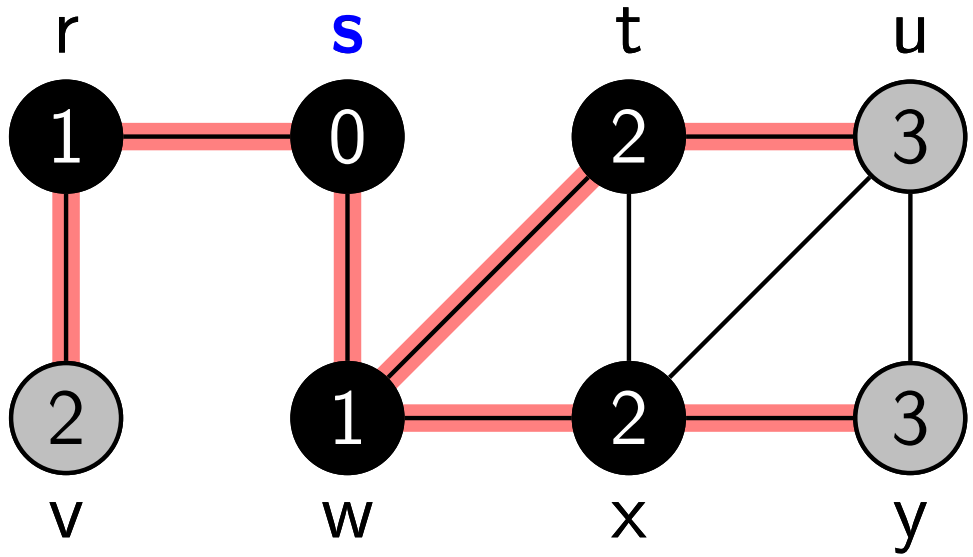
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada y ∈ Adj[x]
        Se cor[y] = branco;
            cor[y] ← cinza;
            d[y] ← d[x] + 1;
            π[y] ← x;
▷               ENQUEUE (Q,y);
    cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | **y** |

Busca-em-largura (G, s)

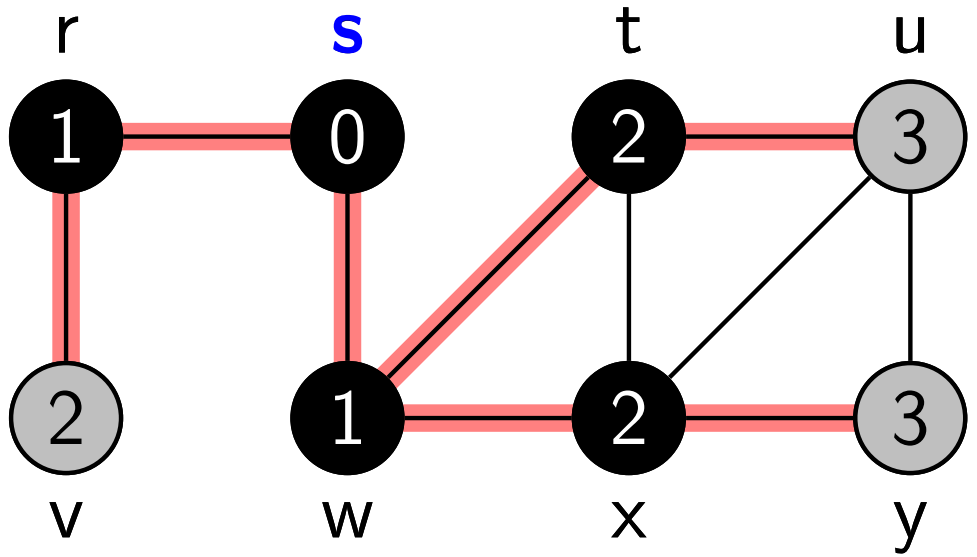
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← x;
    Para cada y ∈ Adj[x]
        Se cor[y] = branco;
            cor[y] ← cinza;
            d[y] ← d[x] + 1;
            π[y] ← x;
            ENQUEUE (Q,y);
▷   cor[x] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | **y** |

Busca-em-largura (G, s)

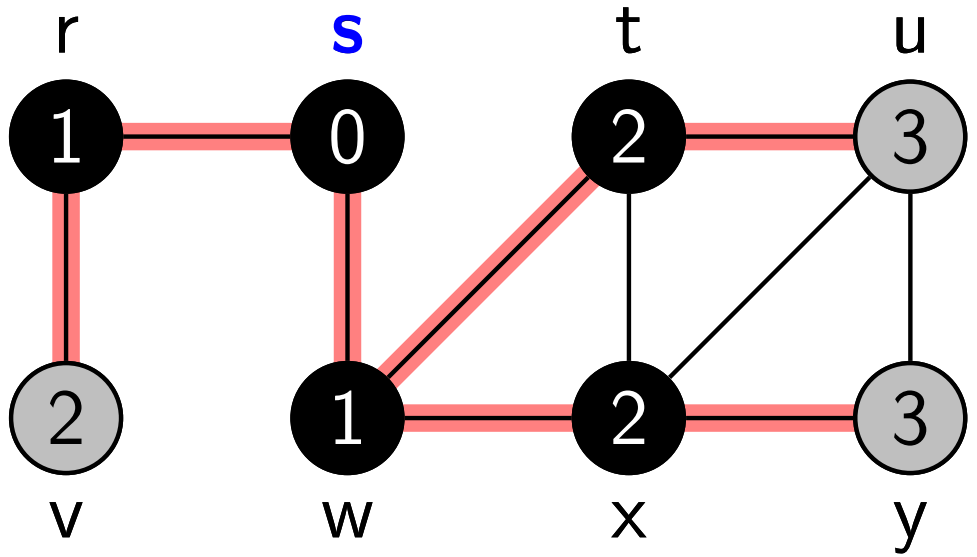
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada r ∈ Adj[v]
            Se cor[r] = branco;
                  cor[r] ← cinza;
                  d[r] ← d[v] + 1;
                  π[r] ← v;
                  ENQUEUE (Q,r);
      cor[v] ← preto;
Retorne d, π;
```



Q = | **v** | **u** | **y** |

Busca-em-largura (G, s)

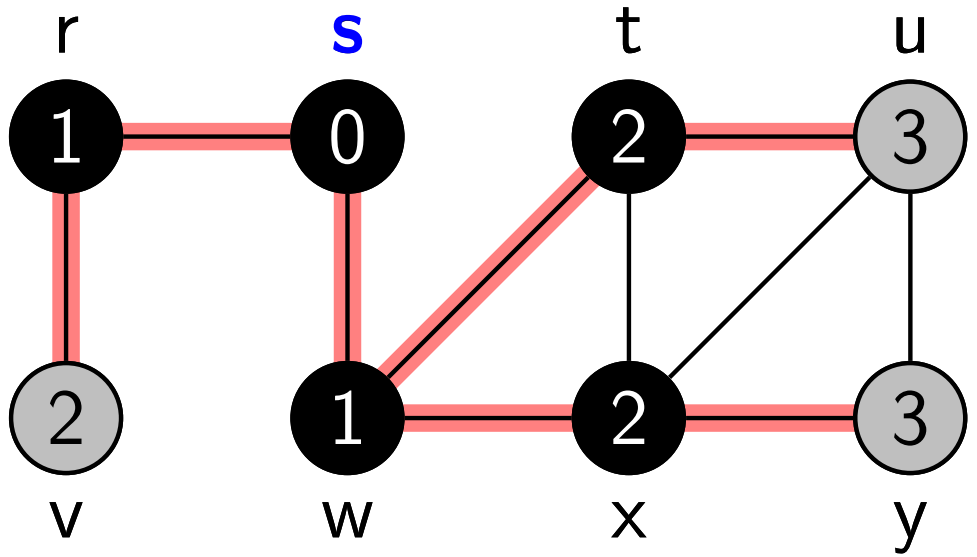
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← v;
    Para cada r ∈ Adj[v]
        Se cor[r] = branco;
            cor[r] ← cinza;
            d[r] ← d[v] + 1;
            π[r] ← v;
            ENQUEUE (Q,r);
    cor[v] ← preto;
Retorne d, π;
```



Q = | **u** | **y** | |

Busca-em-largura (G, s)

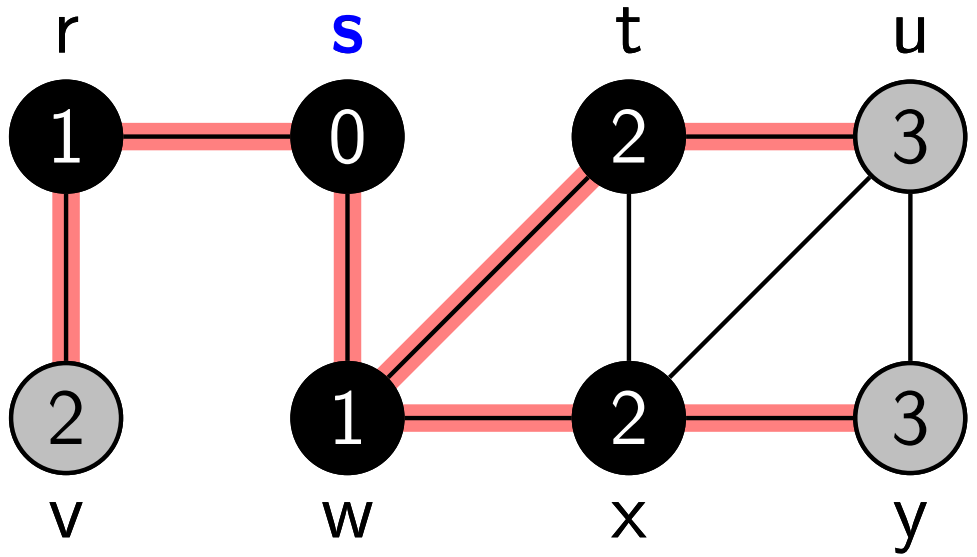
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← v;
▷   Para cada r ∈ Adj[v]
      Se cor[r] = branco;
          cor[r] ← cinza;
          d[r] ← d[v] + 1;
          π[r] ← v;
          ENQUEUE (Q,r);
    cor[v] ← preto;
Retorne d, π;
```



Q = | **u** | **y** | |

Busca-em-largura (G, s)

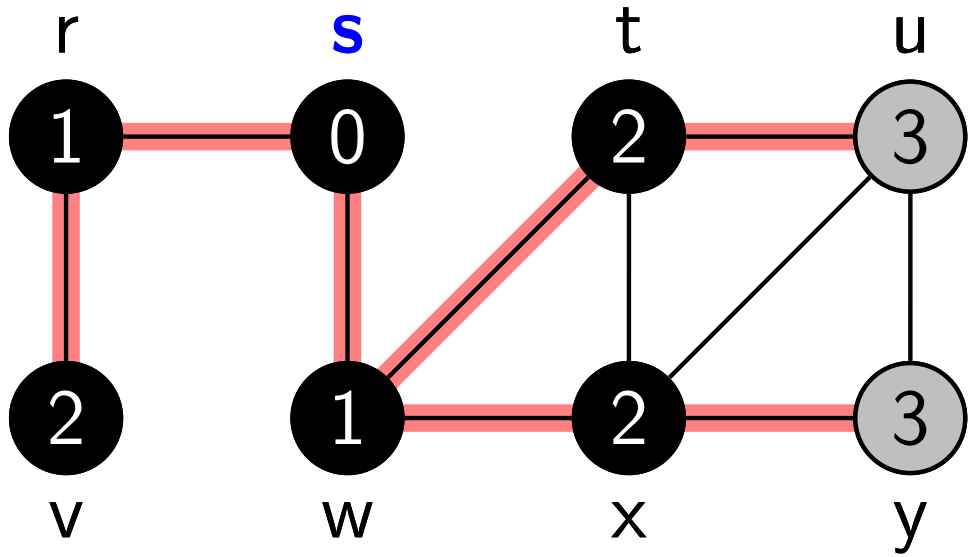
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← v;
    Para cada r ∈ Adj[v]
▷       Se cor[r] = branco;
          cor[r] ← cinza;
          d[r] ← d[v] + 1;
          π[r] ← v;
          ENQUEUE (Q,r);
    cor[v] ← preto;
Retorne d, π;
```



Q = | **u** | **y** | |

Busca-em-largura (G, s)

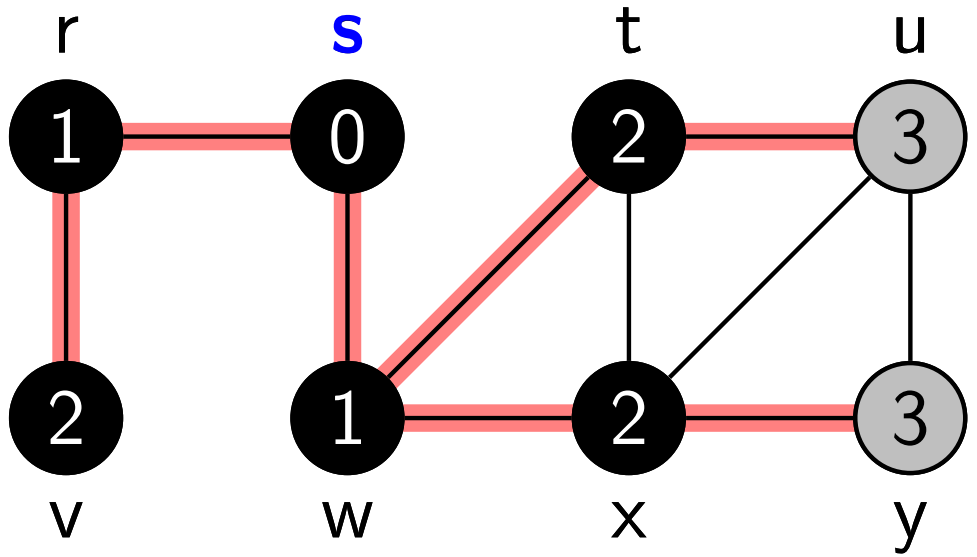
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← v;
    Para cada r ∈ Adj[v]
        Se cor[r] = branco;
            cor[r] ← cinza;
            d[r] ← d[v] + 1;
            π[r] ← v;
            ENQUEUE (Q,r);
▷   cor[v] ← preto;
Retorne d, π;
```



Q = | **u** | **y** | |

Busca-em-largura (G, s)

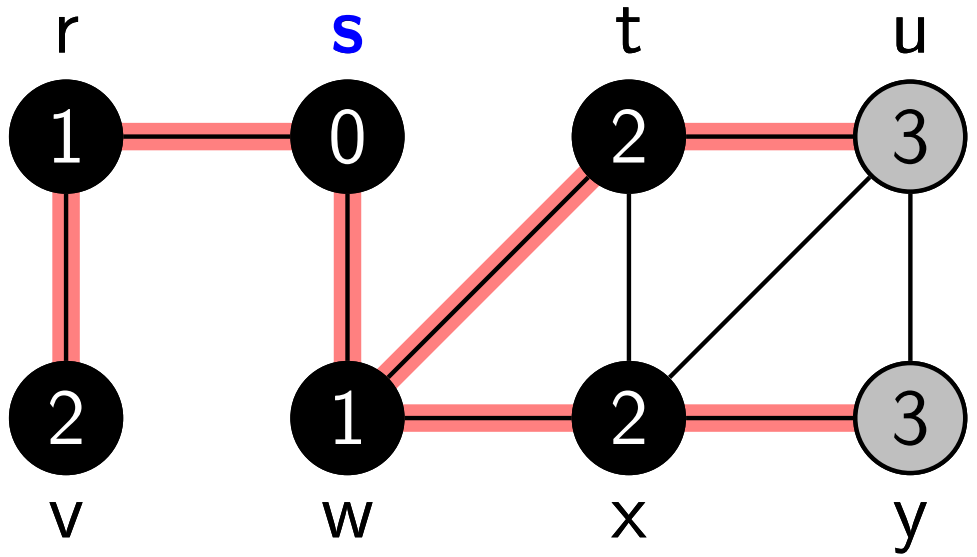
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
    Para cada t ∈ Adj[u]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[u] + 1;
            π[t] ← u;
            ENQUEUE (Q,t);
    cor[u] ← preto;
Retorne d, π;
```



Q = | u | y | |

Busca-em-largura (G, s)

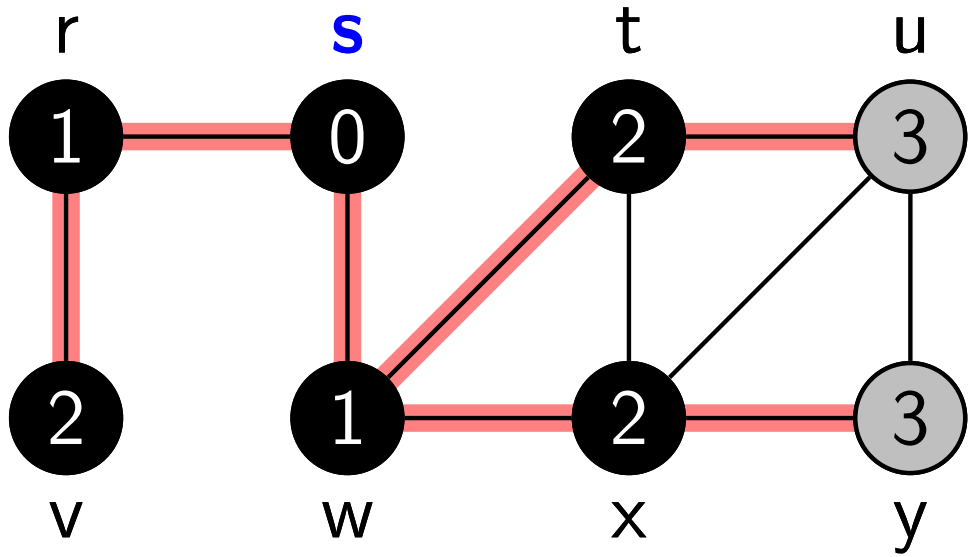
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← u;
    Para cada t ∈ Adj[u]
        Se cor[t] = branco;
            cor[t] ← cinza;
            d[t] ← d[u] + 1;
            π[t] ← u;
            ENQUEUE (Q,t);
    cor[u] ← preto;
Retorne d, π;
```



Q = | **y** | | |

Busca-em-largura (G, s)

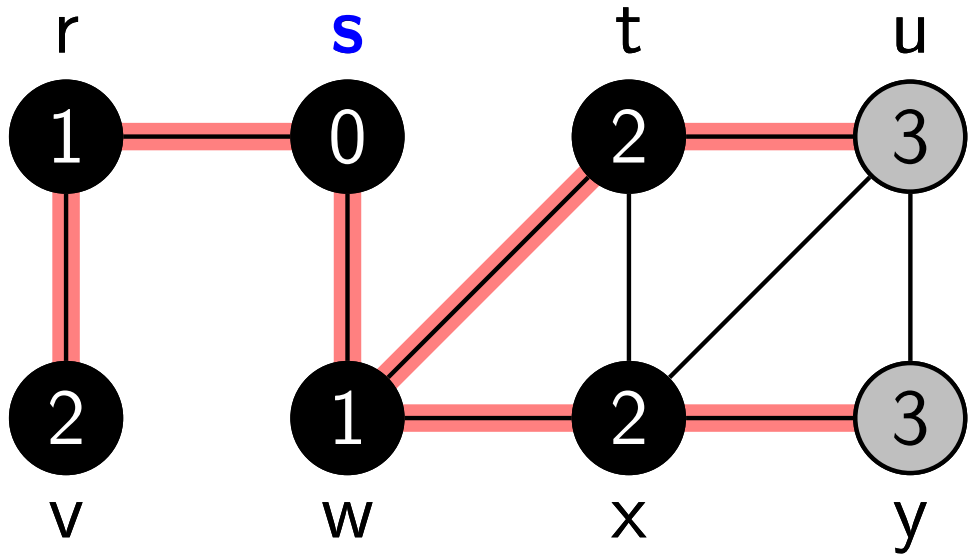
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **u**;
▷   **Para cada t** $\in Adj$[**u**]
        Se cor[**t**] = branco;
            cor[**t**] $\leftarrow$ cinza;
            d[**t**] $\leftarrow$ d[**u**] + 1;
            π[**t**] $\leftarrow$ **u**;
            ENQUEUE (Q,**t**);
    cor[**u**] $\leftarrow$ preto;
Retorne d, π;



Q = | **y** | | |

Busca-em-largura (G, s)

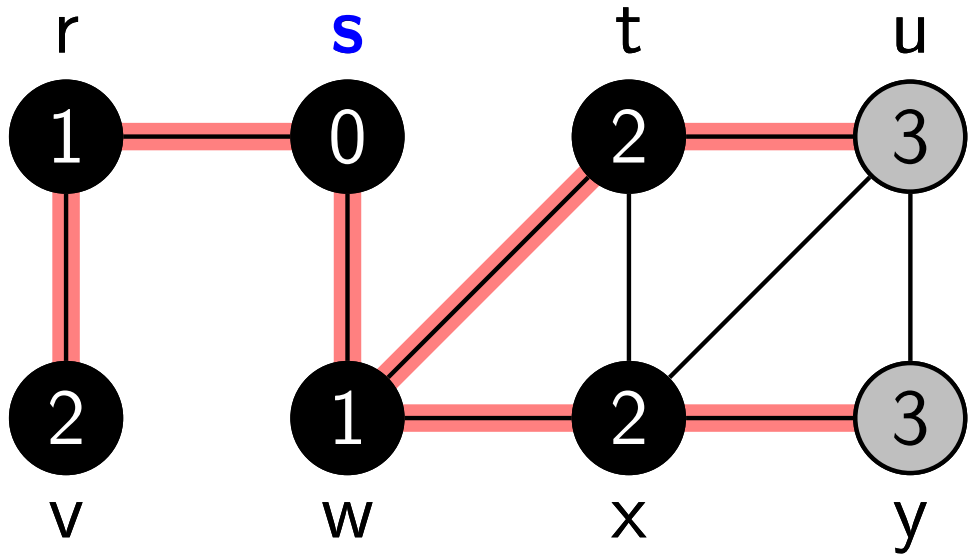
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← u;
    Para cada t ∈ Adj[u]
▷       Se cor[t] = branco;
          cor[t] ← cinza;
          d[t] ← d[u] + 1;
          π[t] ← u;
          ENQUEUE (Q,t);
    cor[u] ← preto;
Retorne d, π;
```



Q = | **y** | | |

Busca-em-largura (G, s)

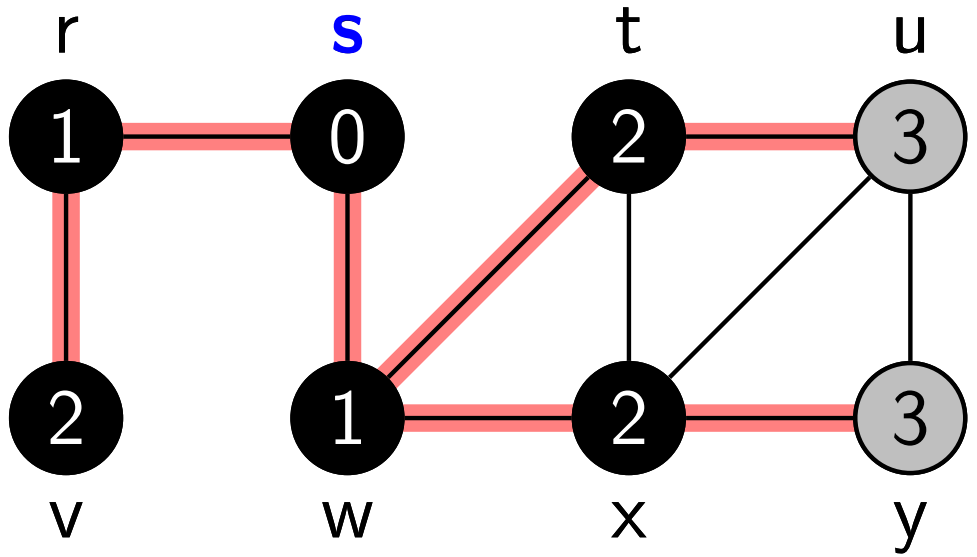
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$**u**;
▷   **Para cada x** $\in Adj$[**u**]
        Se cor[**x**] = branco;
            cor[**x**] $\leftarrow$ cinza;
            d[**x**] $\leftarrow$ d[**u**] + 1;
            π[**x**] $\leftarrow$ **u**;
            ENQUEUE (Q,**x**);
    cor[**u**] $\leftarrow$ preto;
Retorne d, π;



Q = | **y** | | |

Busca-em-largura (G, s)

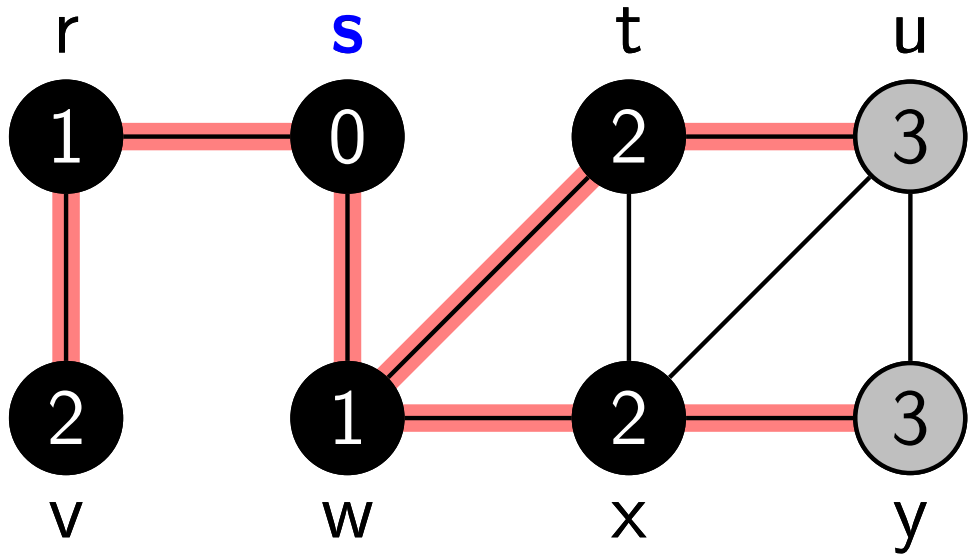
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ←u;
    Para cada x ∈ Adj[u]
▷       Se cor[x] = branco;
          cor[x] ← cinza;
          d[x] ← d[u] + 1;
          π[x] ← u;
          ENQUEUE (Q,x);
    cor[u] ← preto;
Retorne d, π;
```



Q = | **y** | | |

Busca-em-largura (G, s)

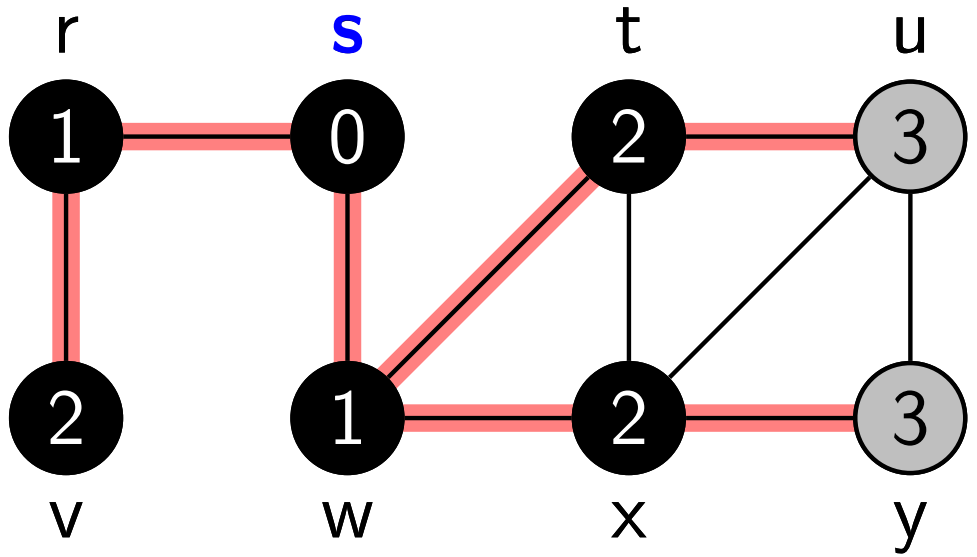
Q $\leftarrow \emptyset$;
ENQUEUE (Q,*s*);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$**u**;
▷   **Para cada y** $\in Adj$[**u**]
        Se cor[**y**] = branco;
            cor[**y**] $\leftarrow$ cinza;
            d[**y**] $\leftarrow$ d[**u**] + 1;
            π[**y**] $\leftarrow$ **u**;
            ENQUEUE (Q,**y**);
    cor[**u**] $\leftarrow$ preto;
Retorne d, π;



Q = | **y** | | |

Busca-em-largura (G, s)

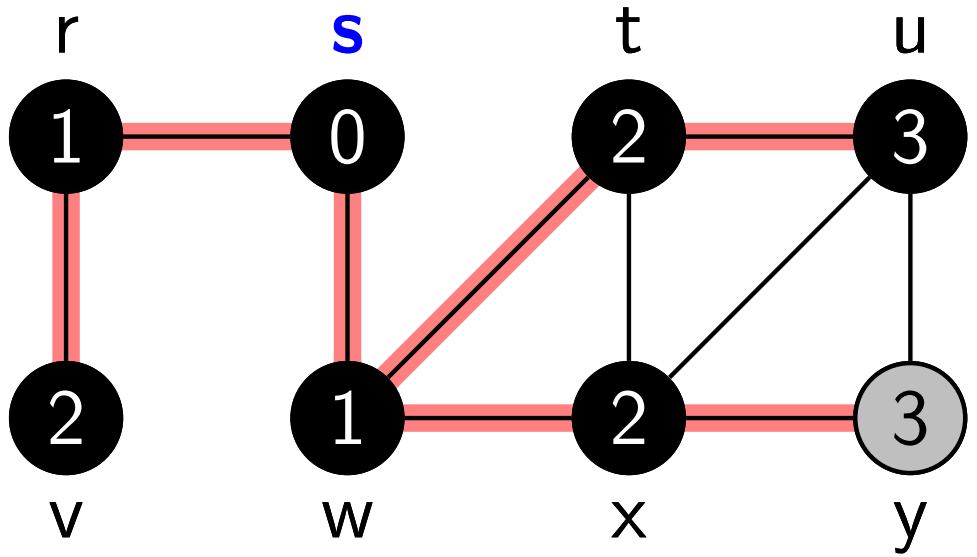
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ←u;
    Para cada y ∈ Adj[u]
▷       Se cor[y] = branco;
          cor[y] ← cinza;
          d[y] ← d[u] + 1;
          π[y] ← u;
          ENQUEUE (Q,y);
    cor[u] ← preto;
Retorne d, π;
```



Q = | **y** | | |

Busca-em-largura (G, s)

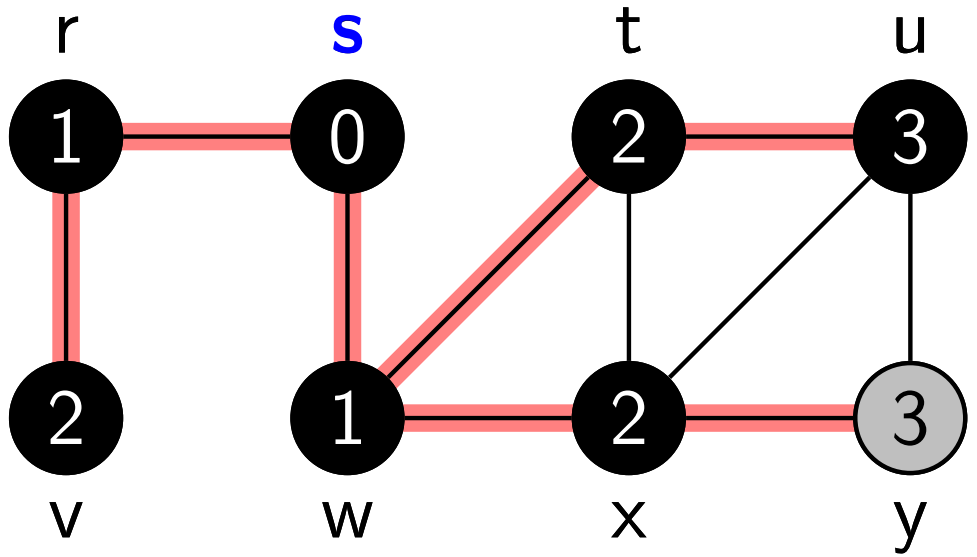
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$**u**;
    Para cada y $\in Adj$[**u**]
        Se cor[**y**] = branco;
            cor[**y**] $\leftarrow$ cinza;
            d[**y**] $\leftarrow$ d[**u**] + 1;
            π[**y**] $\leftarrow$ **u**;
            ENQUEUE (Q,**y**);
$\triangleright$ cor[**u**] $\leftarrow$ preto;
Retorne d, π;



Q = | **y** | | |

Busca-em-largura (G, s)

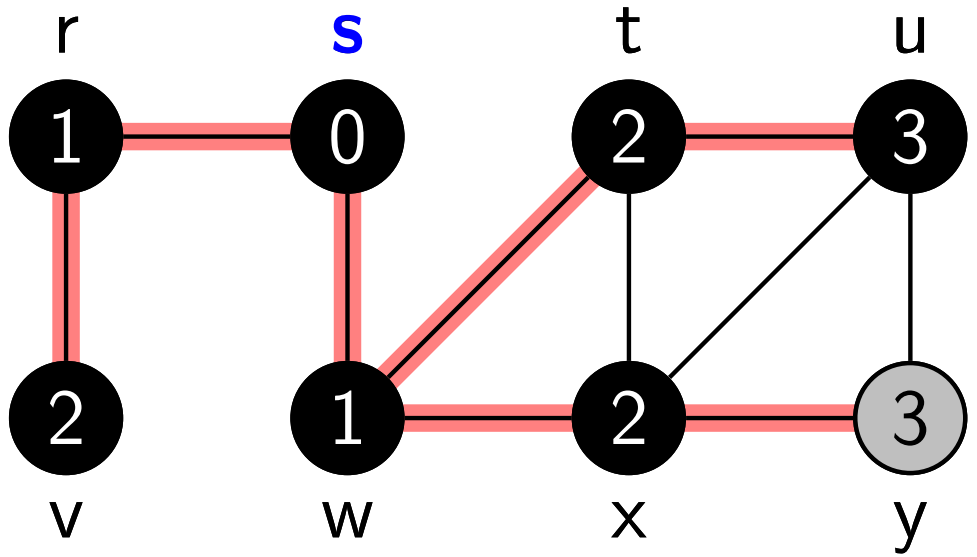
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← DEQUEUE (Q);
      Para cada x ∈ Adj[y]
            Se cor[x] = branco;
                  cor[x] ← cinza;
                  d[x] ← d[y] + 1;
                  π[x] ← y;
                  ENQUEUE (Q,x);
      cor[y] ← preto;
Retorne d, π;
```

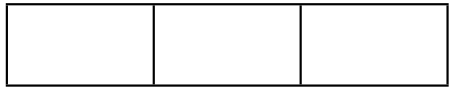


Q = | **y** | | |

Busca-em-largura (G, s)

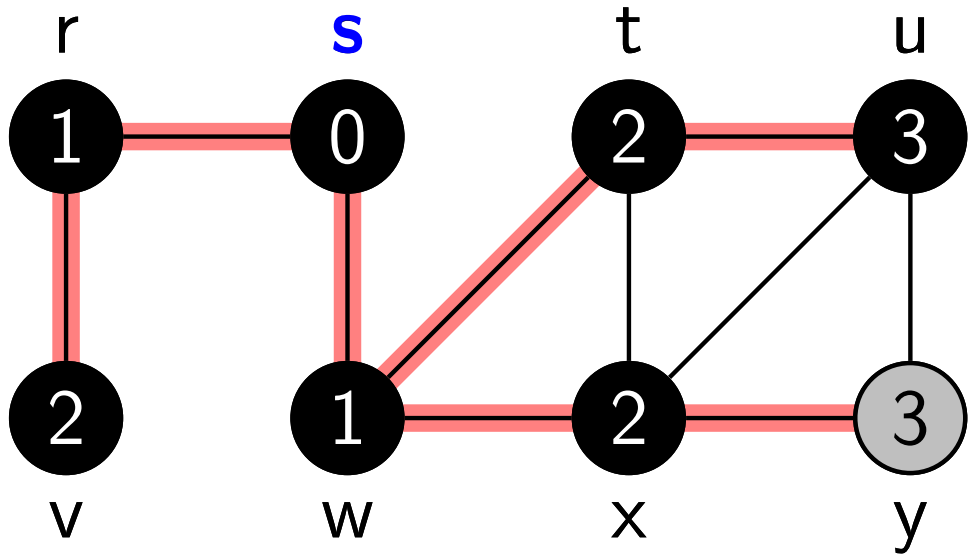
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
▷     u ← y;
    Para cada x ∈ Adj[y]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[y] + 1;
            π[x] ← y;
            ENQUEUE (Q,x);
    cor[y] ← preto;
Retorne d, π;
```

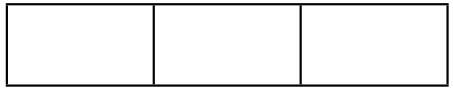


Q = 

Busca-em-largura (G, s)

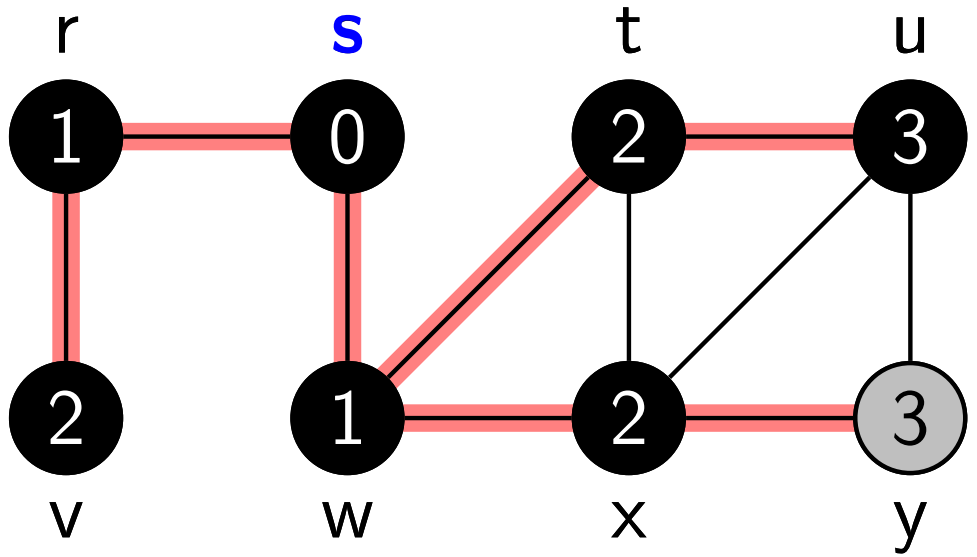
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
▷     Para cada x ∈ Adj[y]
        Se cor[x] = branco;
            cor[x] ← cinza;
            d[x] ← d[y] + 1;
            π[x] ← y;
            ENQUEUE (Q,x);
    cor[y] ← preto;
Retorne d, π;
```

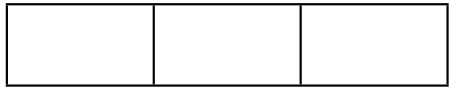


Q = 

Busca-em-largura (G, s)

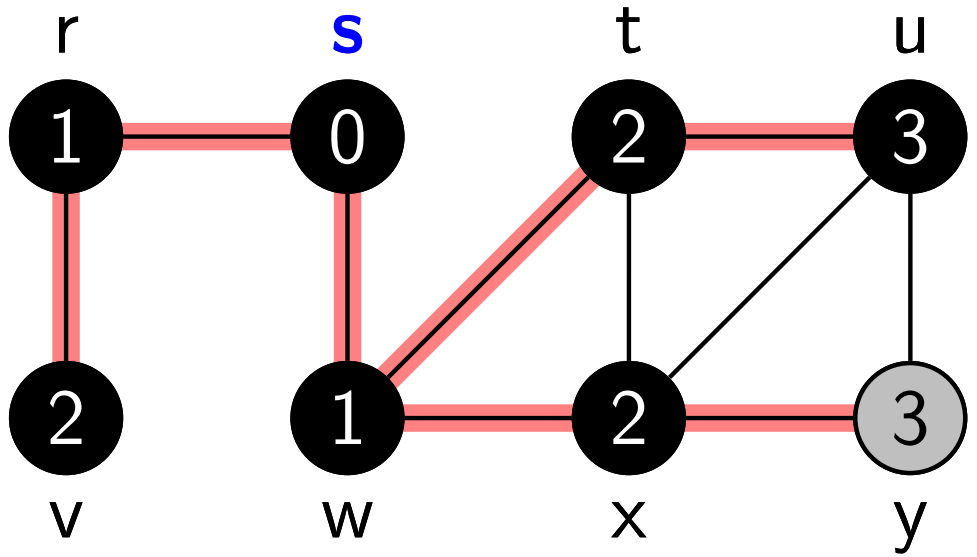
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
    Para cada x ∈ Adj[y]
▷       Se cor[x] = branco;
          cor[x] ← cinza;
          d[x] ← d[y] + 1;
          π[x] ← y;
          ENQUEUE (Q,x);
    cor[y] ← preto;
Retorne d, π;
```

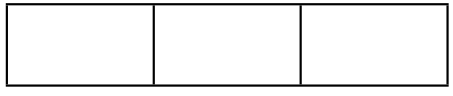


Q = 

Busca-em-largura (G, s)

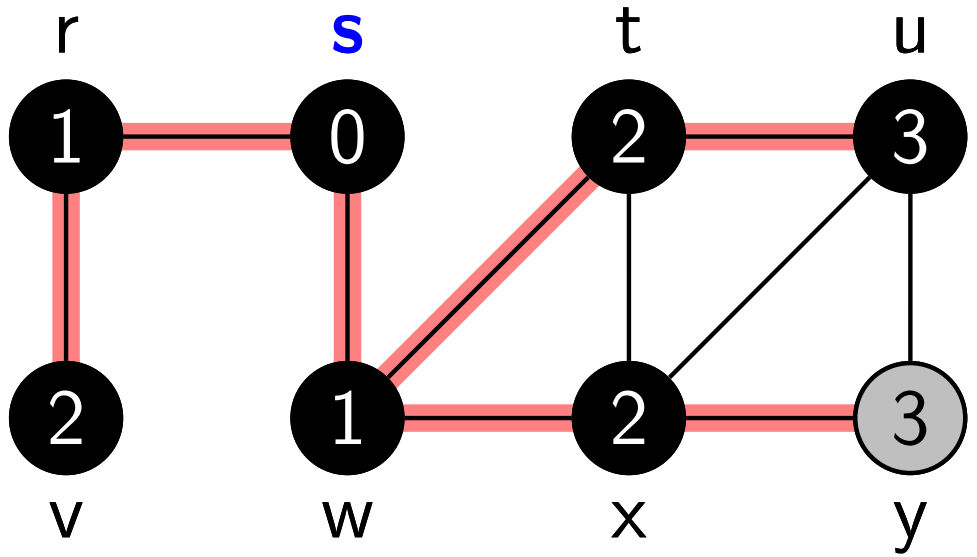
Q $\leftarrow \emptyset$;
ENQUEUE (Q,s);
Enquanto $Q \neq \emptyset$
    u $\leftarrow$ **y**;
▷   **Para cada u** $\in Adj$[**y**]
        Se cor[**u**] $=$ branco;
            cor[**u**] $\leftarrow$ cinza;
            d[**u**] $\leftarrow$ d[**y**] $+ 1$;
            π[**u**] $\leftarrow$ **y**;
            ENQUEUE (Q,**u**);
    cor[**y**] $\leftarrow$ preto;
Retorne d, π;

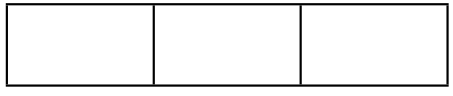


Q $=$ 

Busca-em-largura (G, s)

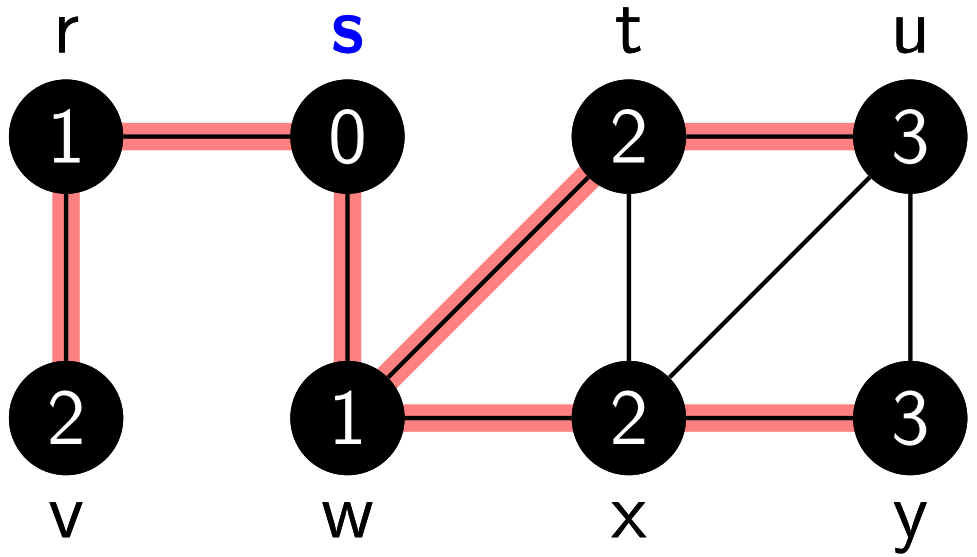
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
    Para cada u ∈ Adj[y]
▷       Se cor[u] = branco;
          cor[u] ← cinza;
          d[u] ← d[y] + 1;
          π[u] ← y;
          ENQUEUE (Q,u);
    cor[y] ← preto;
Retorne d, π;
```

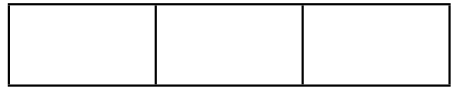


Q = 

Busca-em-largura (G, s)

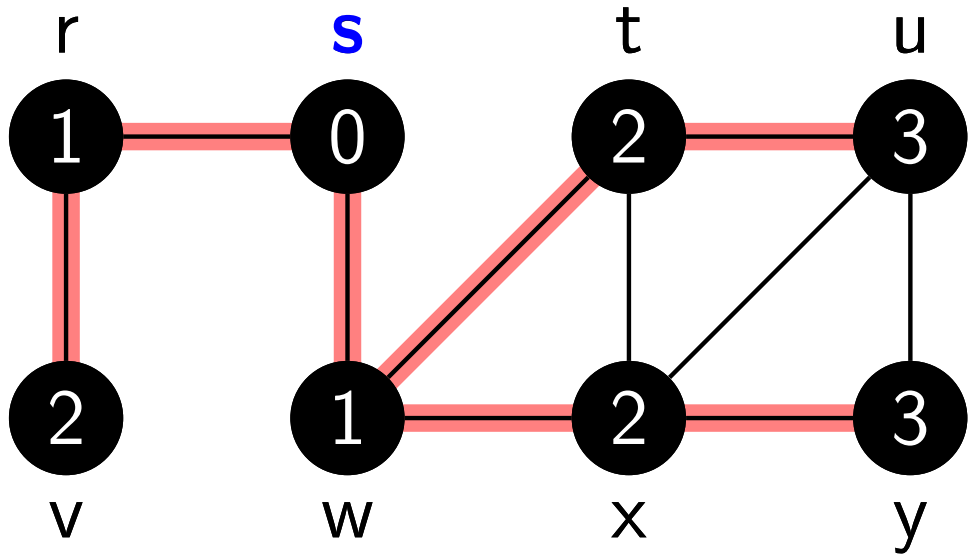
```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
    Para cada u ∈ Adj[y]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[y] + 1;
            π[u] ← y;
            ENQUEUE (Q,u);
▷   cor[y] ← preto;
Retorne d, π;
```

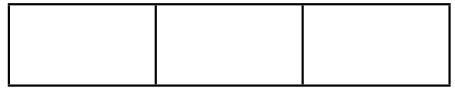


Q = 

Busca-em-largura (G, s)

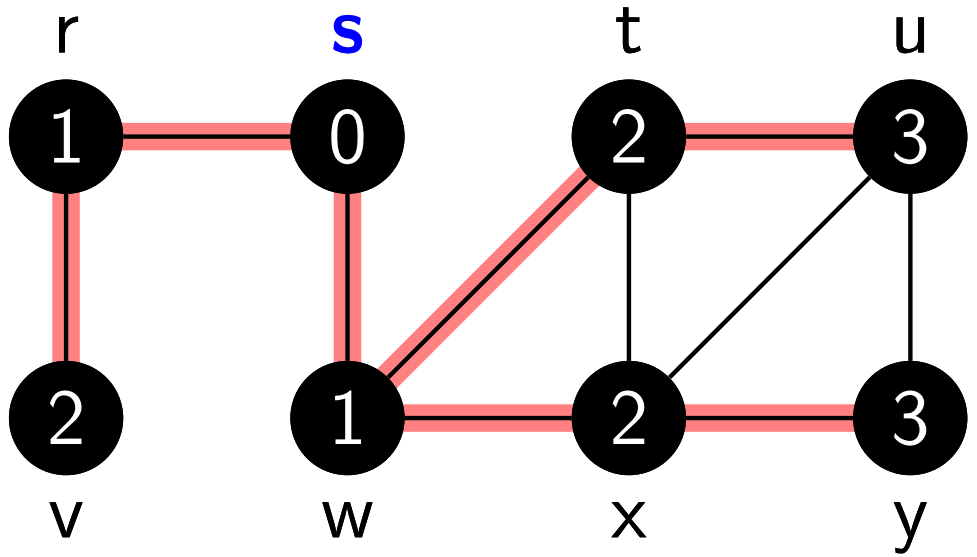
```
Q ← ∅;
ENQUEUE (Q,s);
▷Enquanto Q ≠ ∅
    u ← y;
    Para cada u ∈ Adj[y]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[y] + 1;
            π[u] ← y;
            ENQUEUE (Q,u);
    cor[y] ← preto;
Retorne d, π;
```

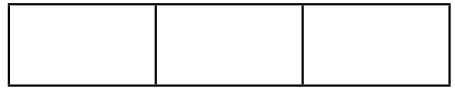


Q = 

Busca-em-largura (G, s)

```
Q ← ∅;
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
    Para cada u ∈ Adj[y]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[y] + 1;
            π[u] ← y;
            ENQUEUE (Q,u);
    cor[y] ← preto;
▷Retorne d, π;
```



Q = 

Busca em largura - Complexidade

A inicialização consome tempo $\Theta(V)$.

Para cada **u** $\in V[G] - \{s\}$
    cor[**u**] $\leftarrow$ branco;
    d[**u**] $\leftarrow \infty$;
    π[**u**] $\leftarrow$ NIL;
cor[**s**] $\leftarrow$ cinza;
d[**s**] $\leftarrow 0$;
π[**s**] $\leftarrow$ NIL;

Busca em largura - Complexidade

Depois que um vértice deixa de ser branco, ele não volta a ser branco novamente. Assim, cada vértice é inserido na fila Q no máximo uma vez. Cada operação sobre a fila consome tempo $\Theta(1)$ resultando em um total de $\mathcal{O}(V)$.

```
ENQUEUE (Q,s);
Enquanto Q ≠ ∅
    u ← y;
    Para cada u ∈ Adj[y]
        Se cor[u] = branco;
            cor[u] ← cinza;
            d[u] ← d[y] + 1;
            π[u] ← y;
            ENQUEUE (Q,u);
    cor[y] ← preto;
Retorne d, π;
```

Busca em largura - Consumo de tempo

- Em uma lista de adjacência, cada vértice é percorrido apenas uma vez. A soma dos comprimentos das listas é $\Theta(E)$. Assim, o tempo gasto para percorrer as listas é $\mathcal{O}(E)$.
- **Conclusão**: A complexidade de tempo de BUSCA-EM-LARGURA é $\mathcal{O}(V + E)$.