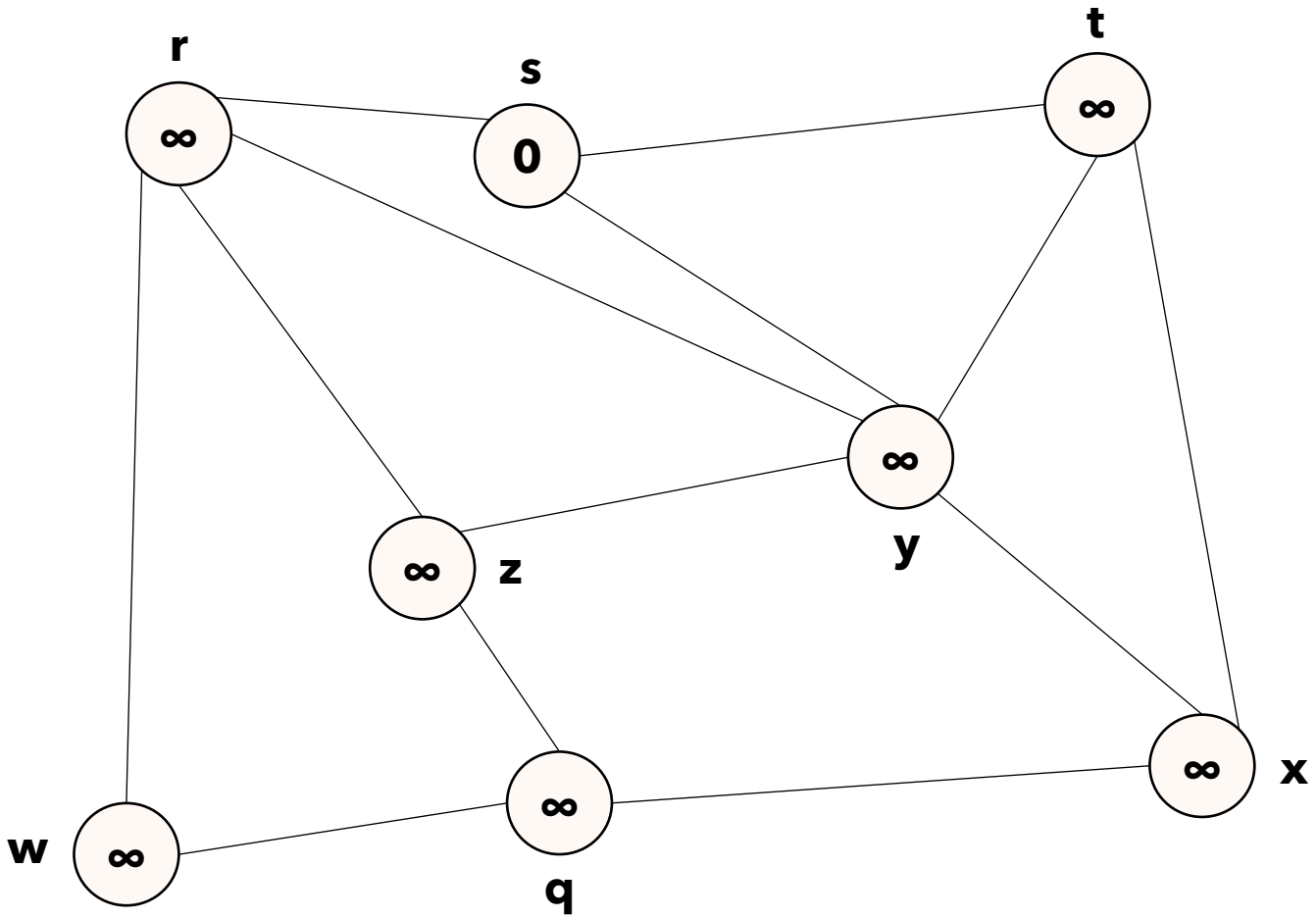
Algoritmi sui grafi
Breadth-first search

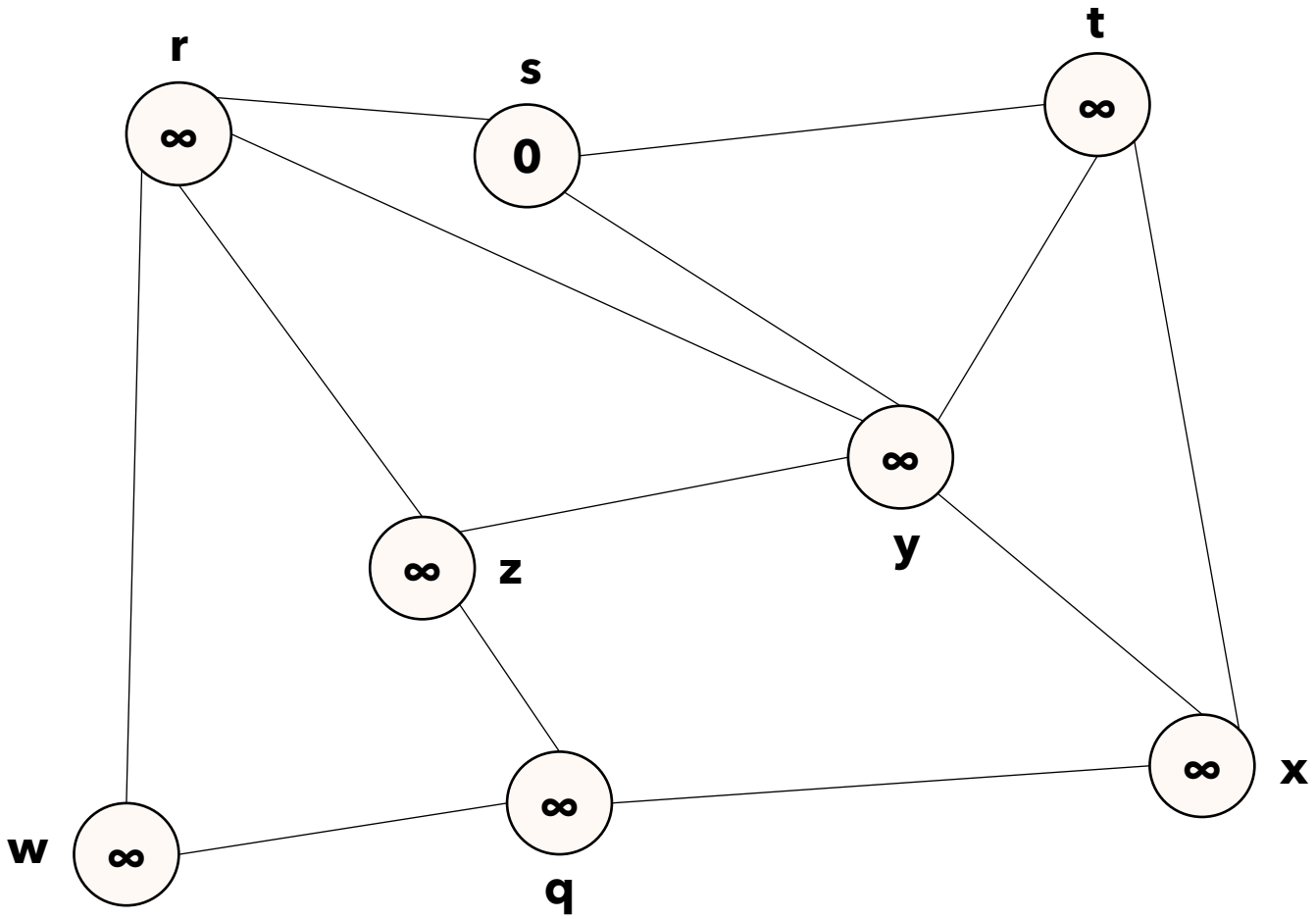
Liceo G.B. Brocchi – Bassano del Grappa (VI)
Liceo Scientifico – opzione scienze applicate
Giovanni Mazzocchin

Breadth-first search



- la **BFS** opera su un grafo non diretto e non pesato
- all'interno di ciascun nodo scriviamo la distanza minima dal nodo sorgente s
- essendo il grafo non pesato, la distanza minima tra 2 nodi x e y è definita come il numero minimo di archi che portano da x a y

Breadth-first search

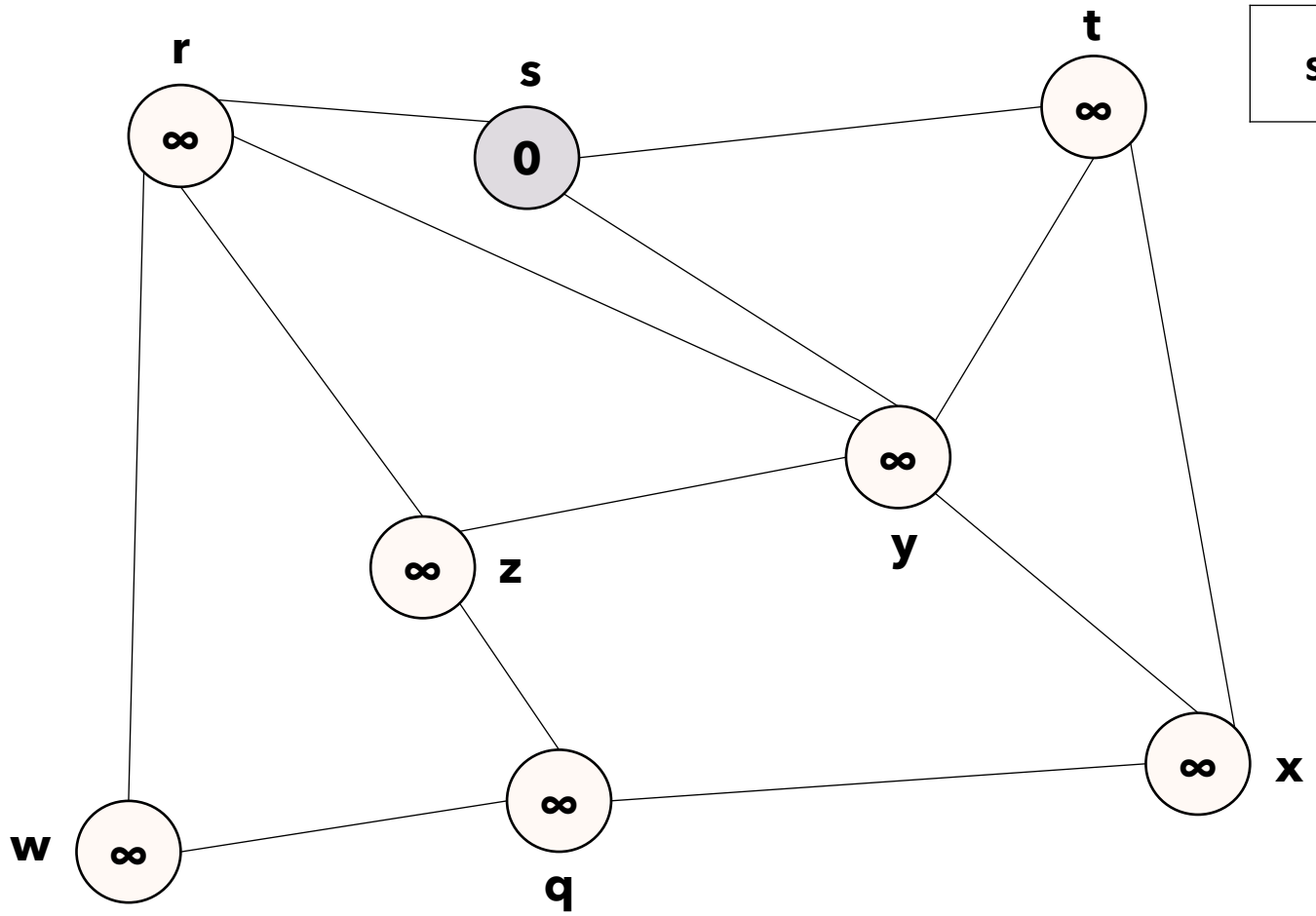


ciascun nodo può essere:

1. white: non ancora scoperto
2. gray: scoperto, ma non tutti i suoi successori sono stati scoperti
3. black: scoperto, e scoperti anche tutti i suoi successori

la BFS richiede una queue come struttura dati di appoggio

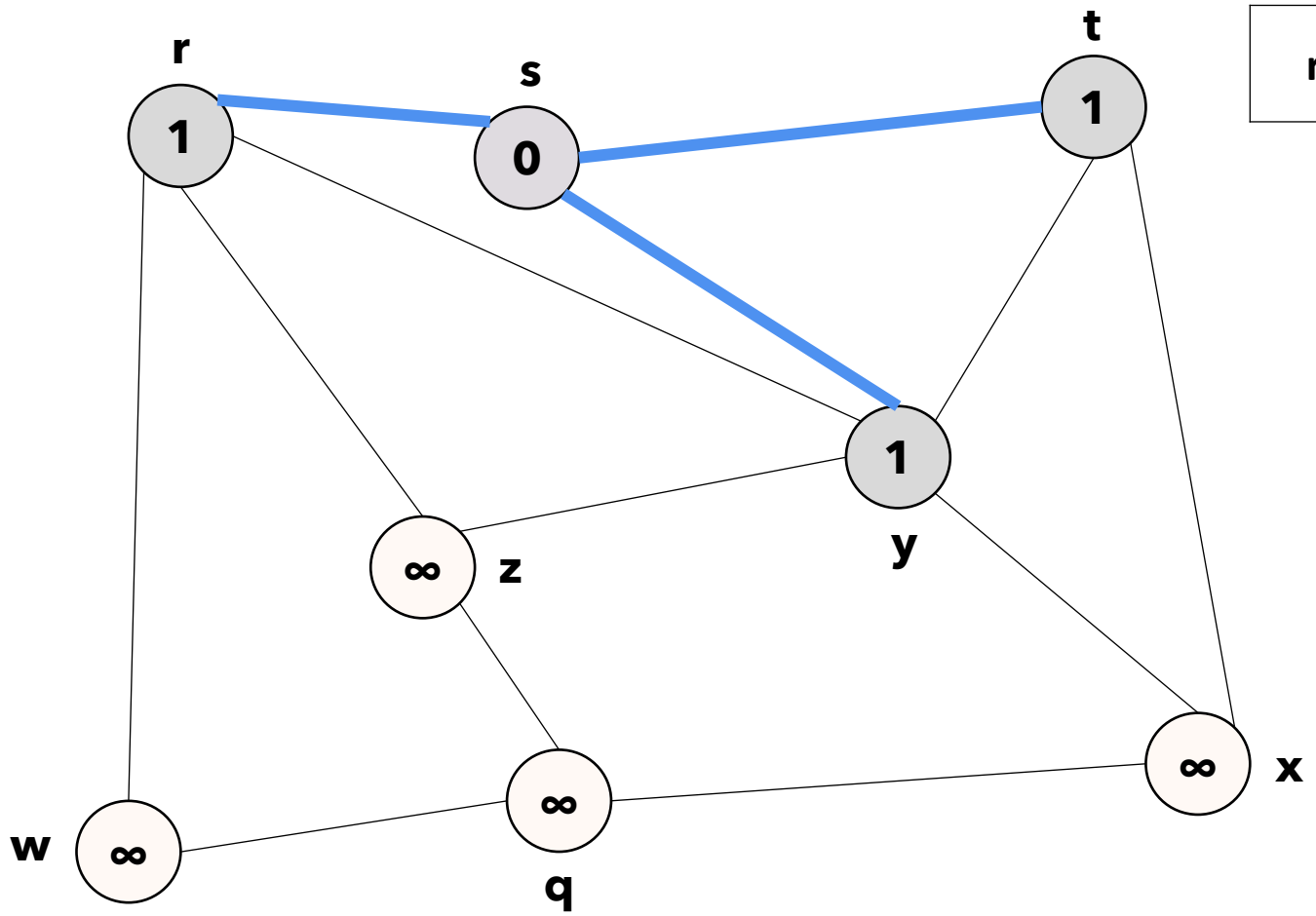
Breadth-first search



s							

```
enqueue(s)

colour s gray
```

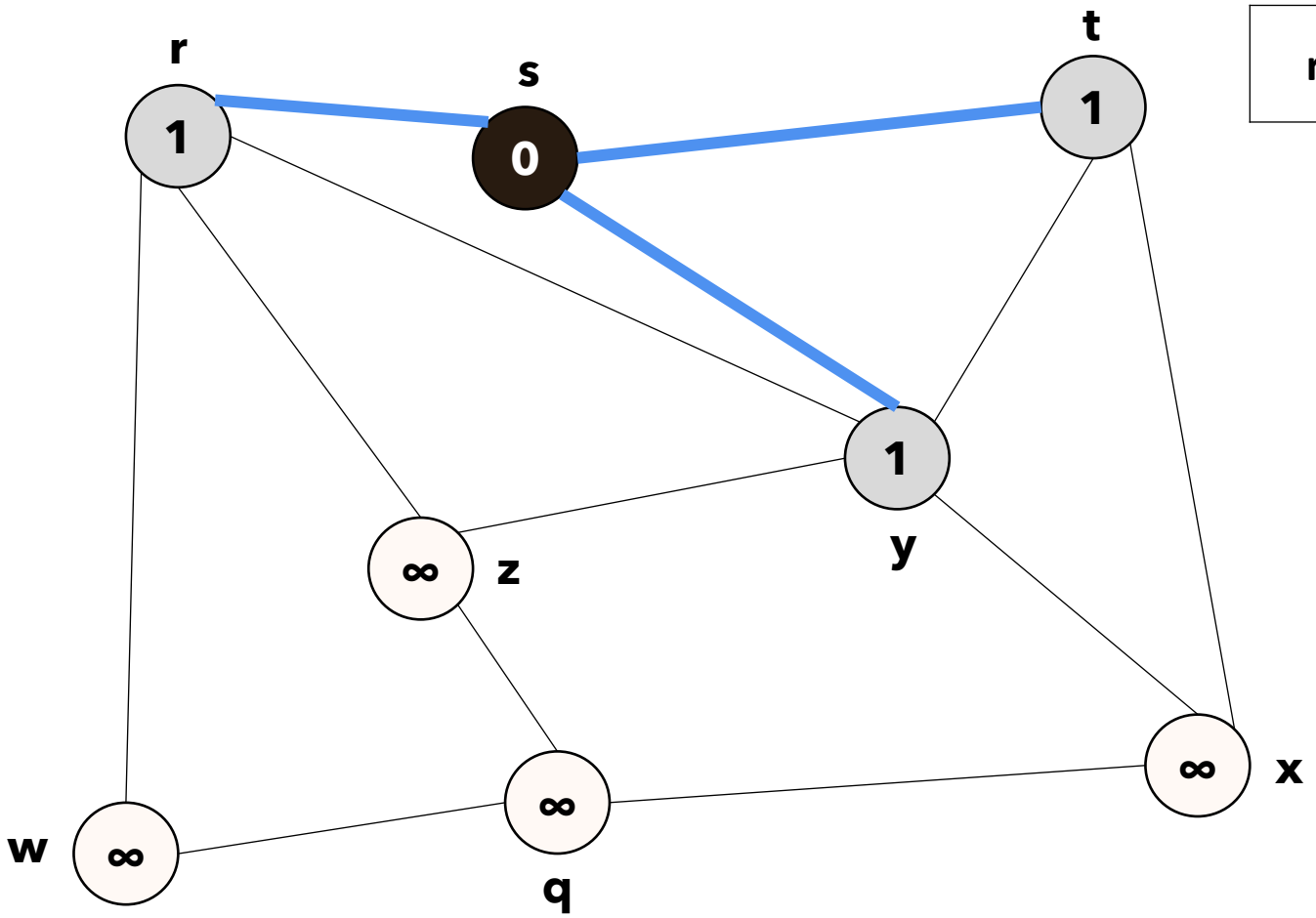
Breadth-first search



r	y	t					

```
dequeue -> s

enqueue s' white neighbours
colour them gray

r.predecessor = s
y.predecessor = s
t.predecessor = s

r.distance = s.distance + 1 = 1
y.distance = s.distance + 1 = 1
t.distance = s.distance + 1 = 1

set tree edges (blue)
```

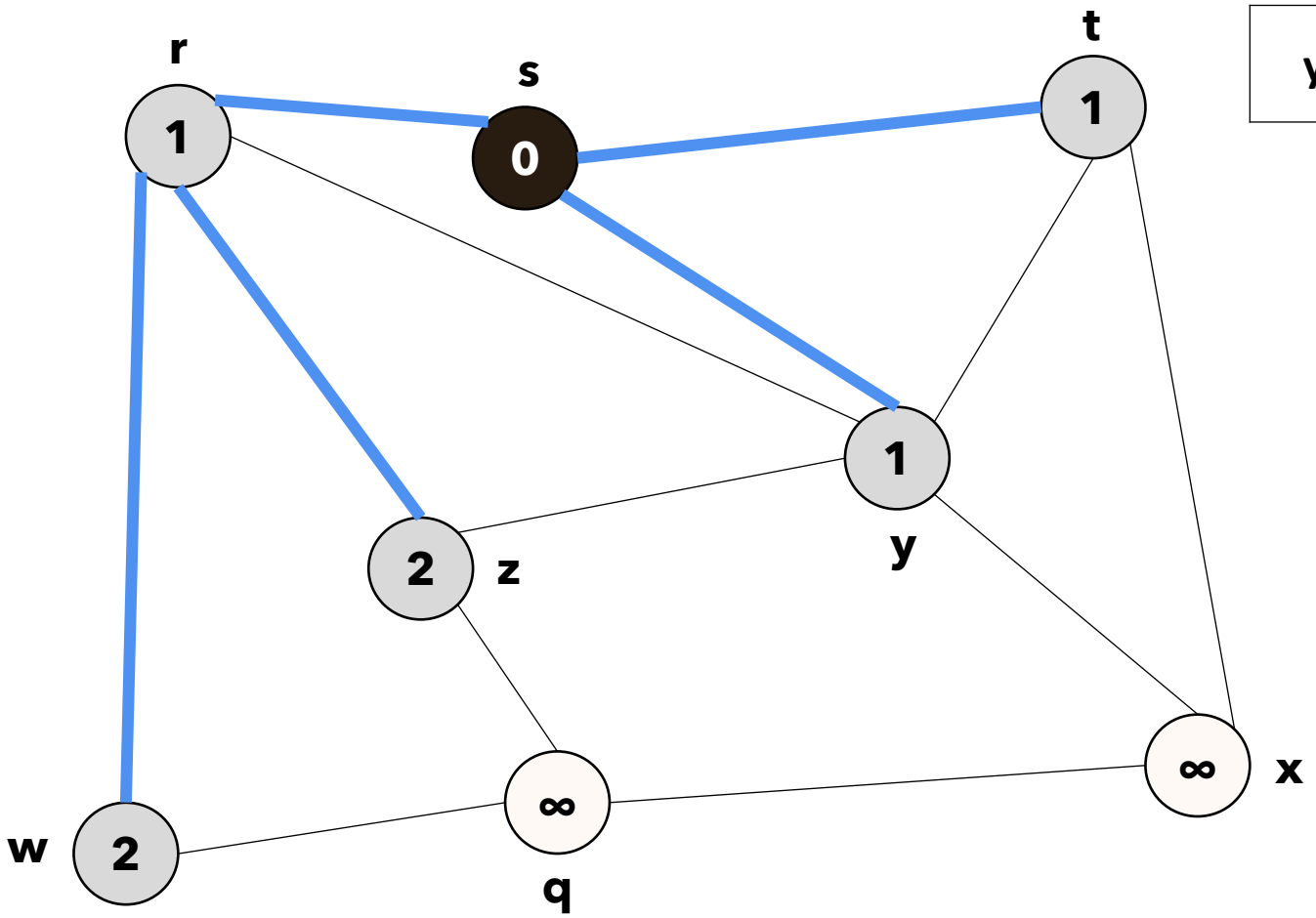
Breadth-first search



r	y	t					

colour s black

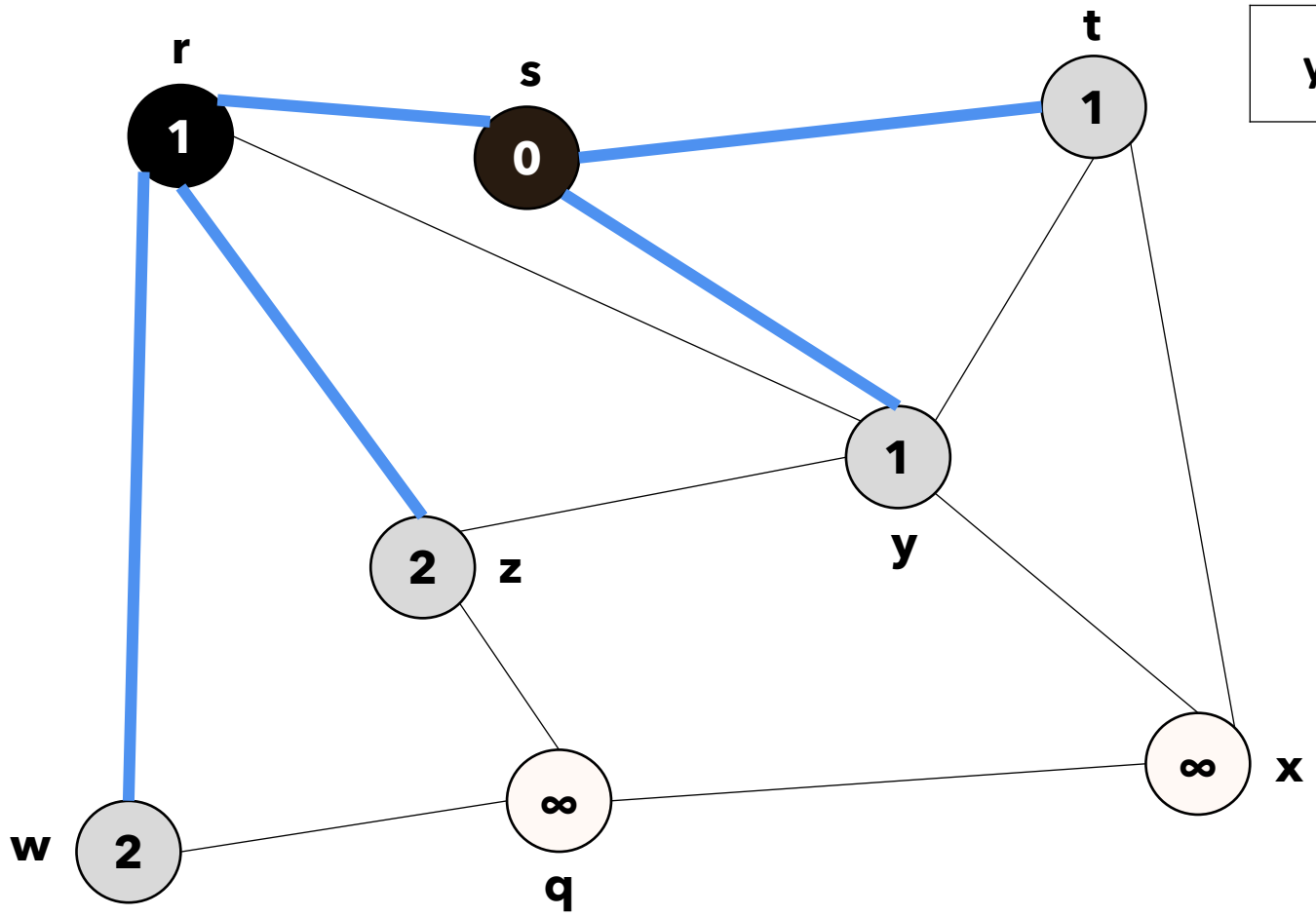
Breadth-first search



y	t	w	z				

```
dequeue -> r

enqueue r' white neighbours
colour them gray

w.predecessor = r
z.predecessor = r

w.distance = r.distance + 1 = 2
z.distance = r.distance + 1 = 2

set tree edges (blue)
```

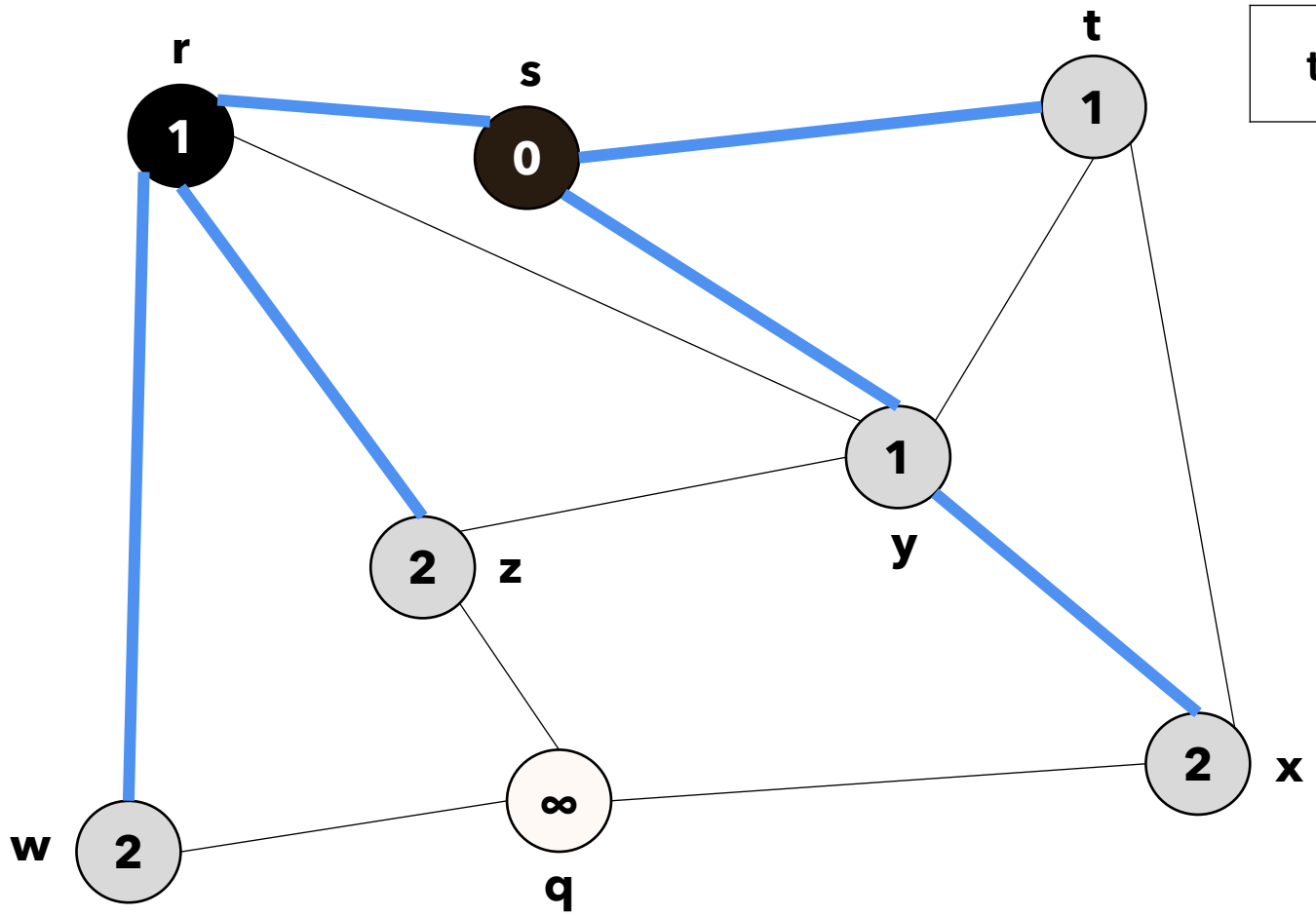
Breadth-first search



y	t	w	z				

colour r black

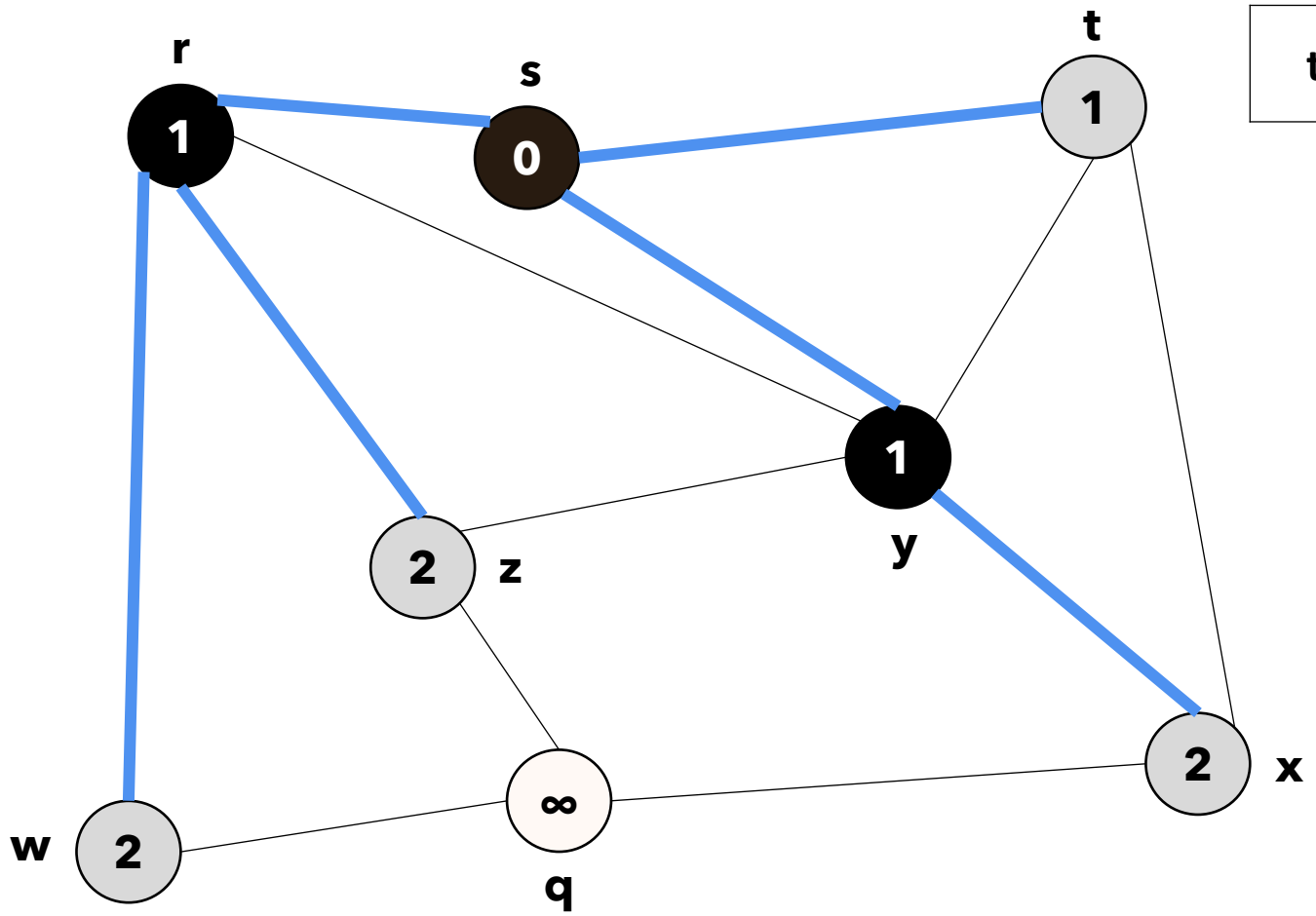
Breadth-first search



t	w	z	x				

```
dequeue -> y

enqueue y' white neighbours
colour them gray

x.predecessor = y

x.distance = y.distance + 1 = 2

set tree edges (blue)
```

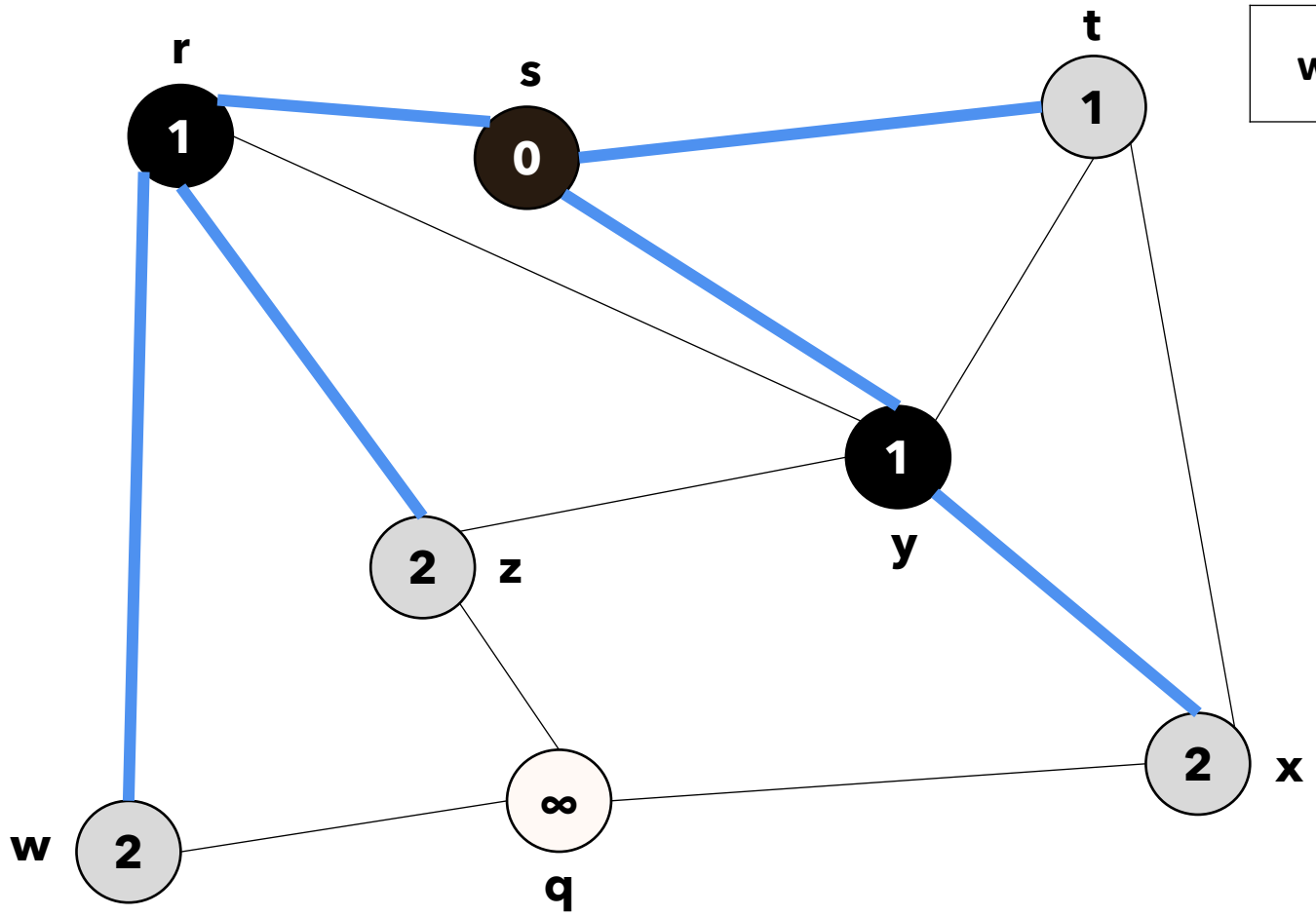
Breadth-first search



t	w	z	x				

colour y black

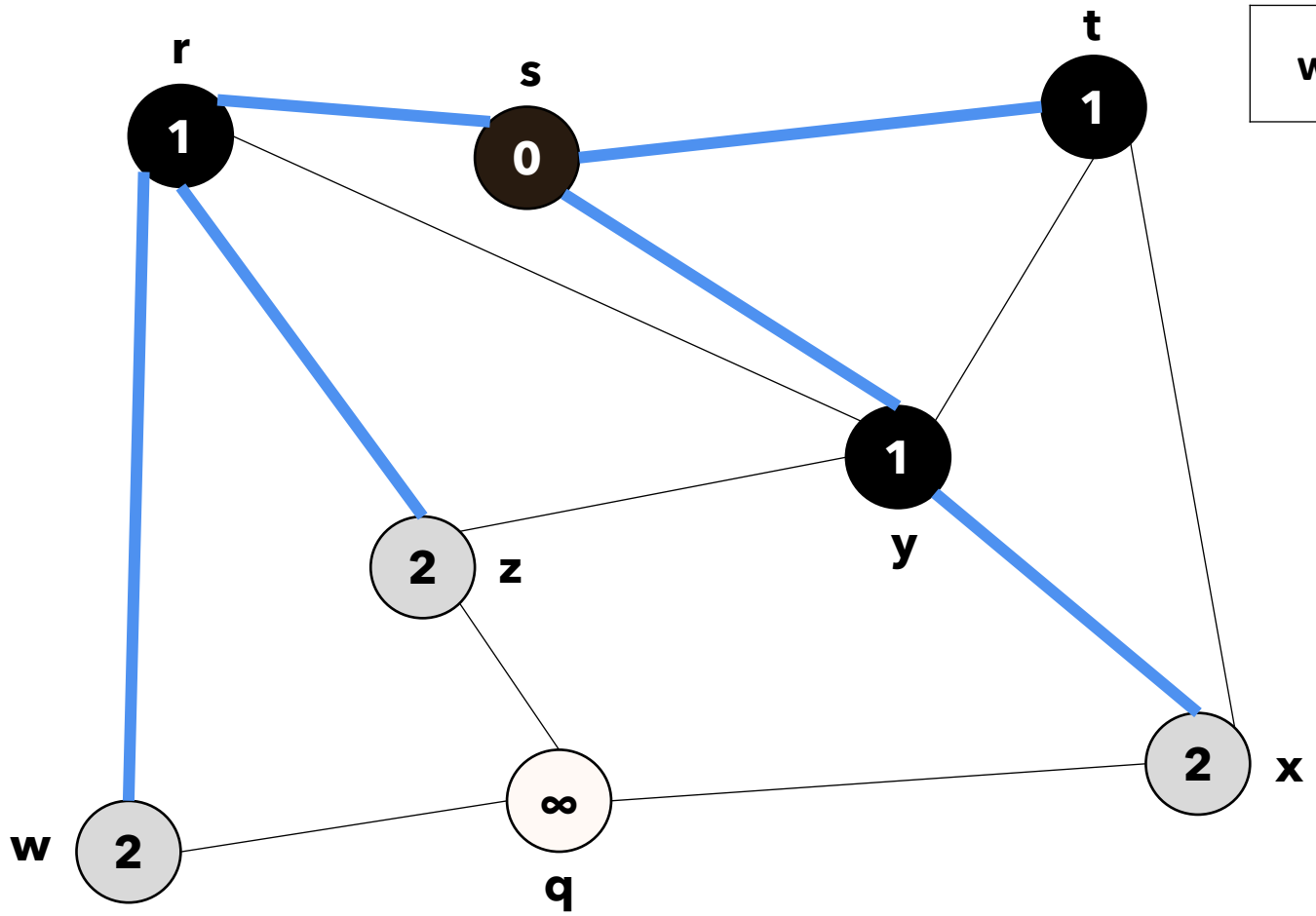
Breadth-first search



w	z	x					

```
dequeue -> t
```

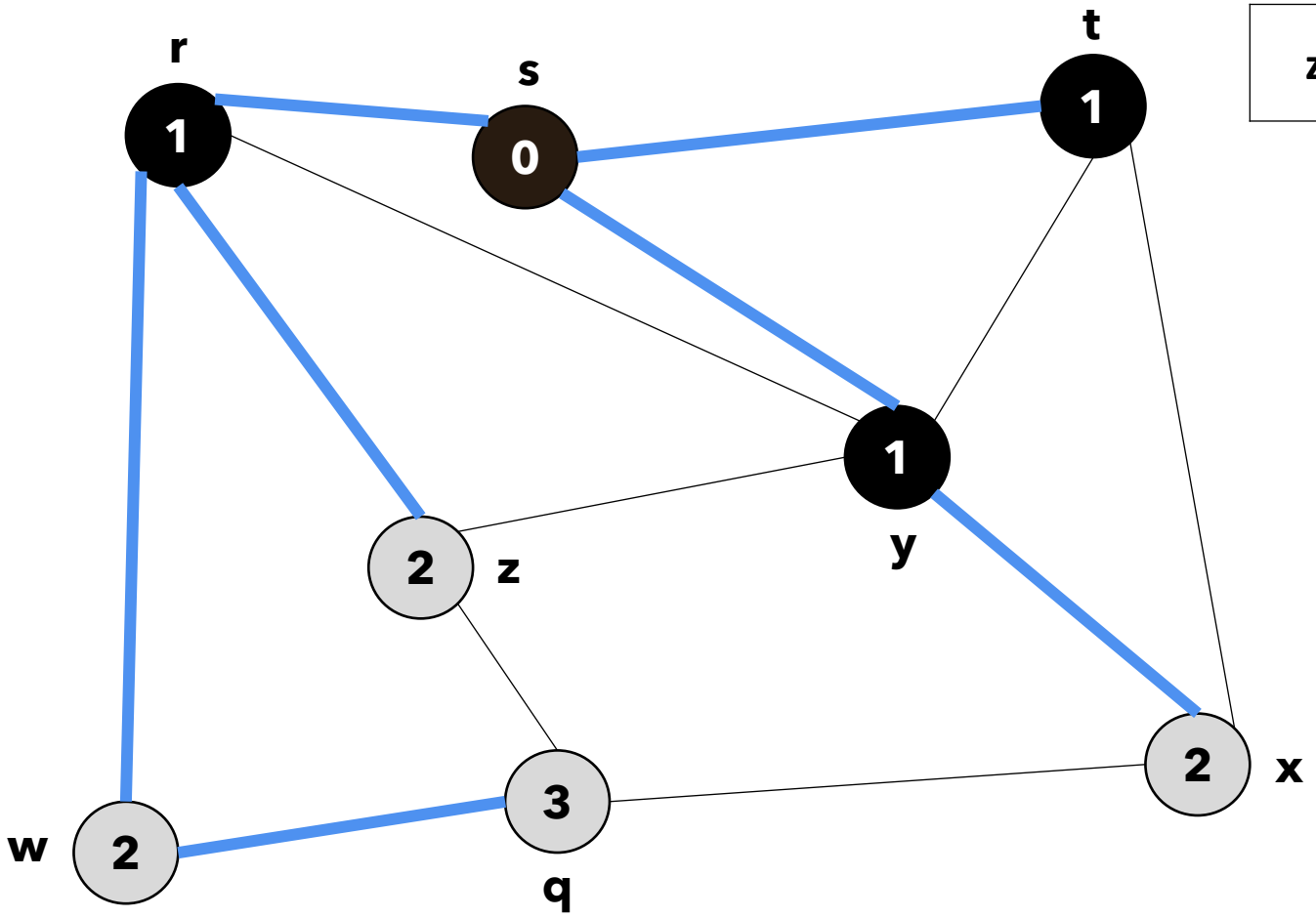
Breadth-first search



w	z	x					

```
colour t black
```

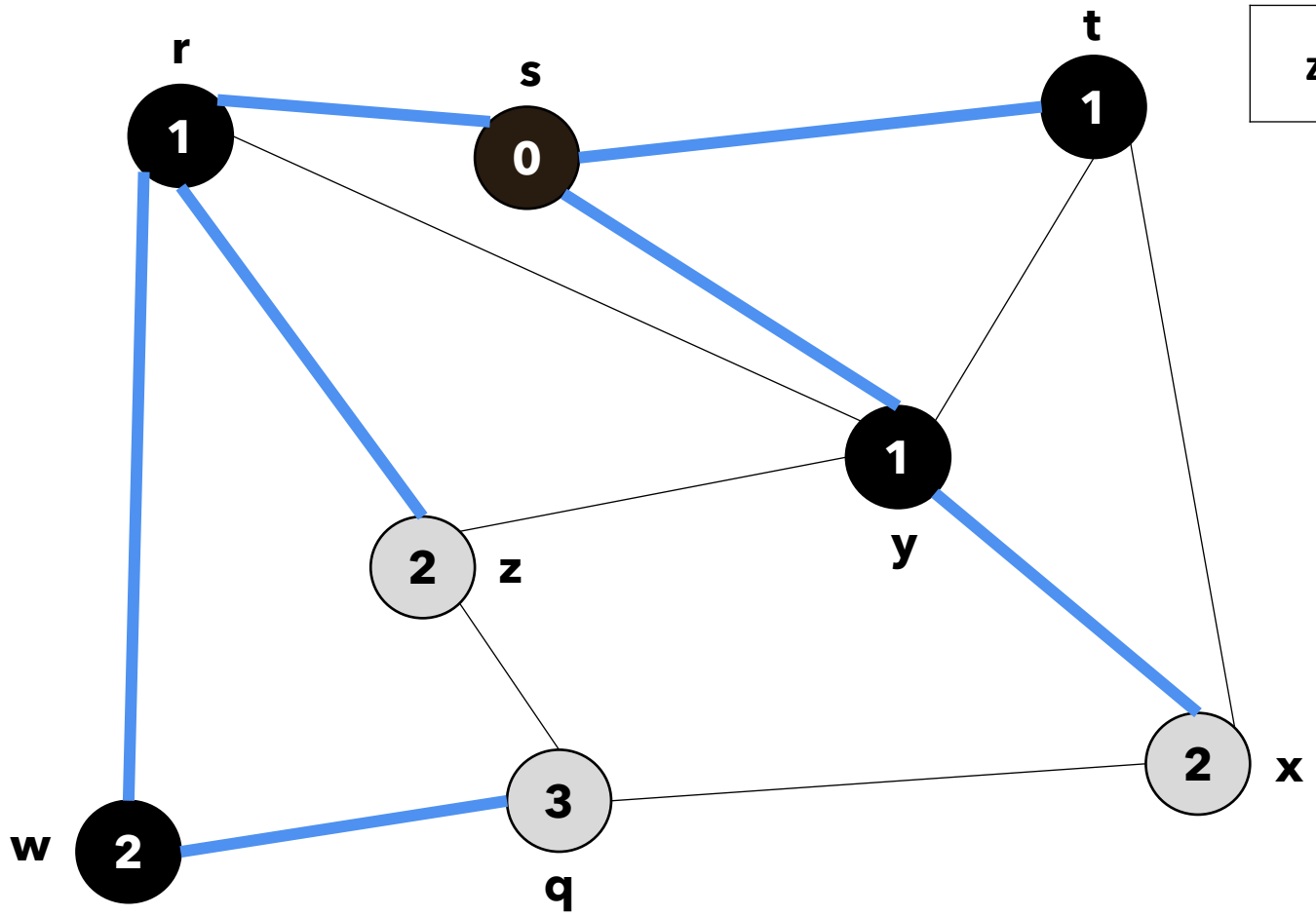
Breadth-first search



z	x	q					

```
dequeue -> w

enqueue w' white neighbours
colour them gray

q.predecessor = w

q.distance = w.distance + 1 = 3

set tree edges (blue)
```

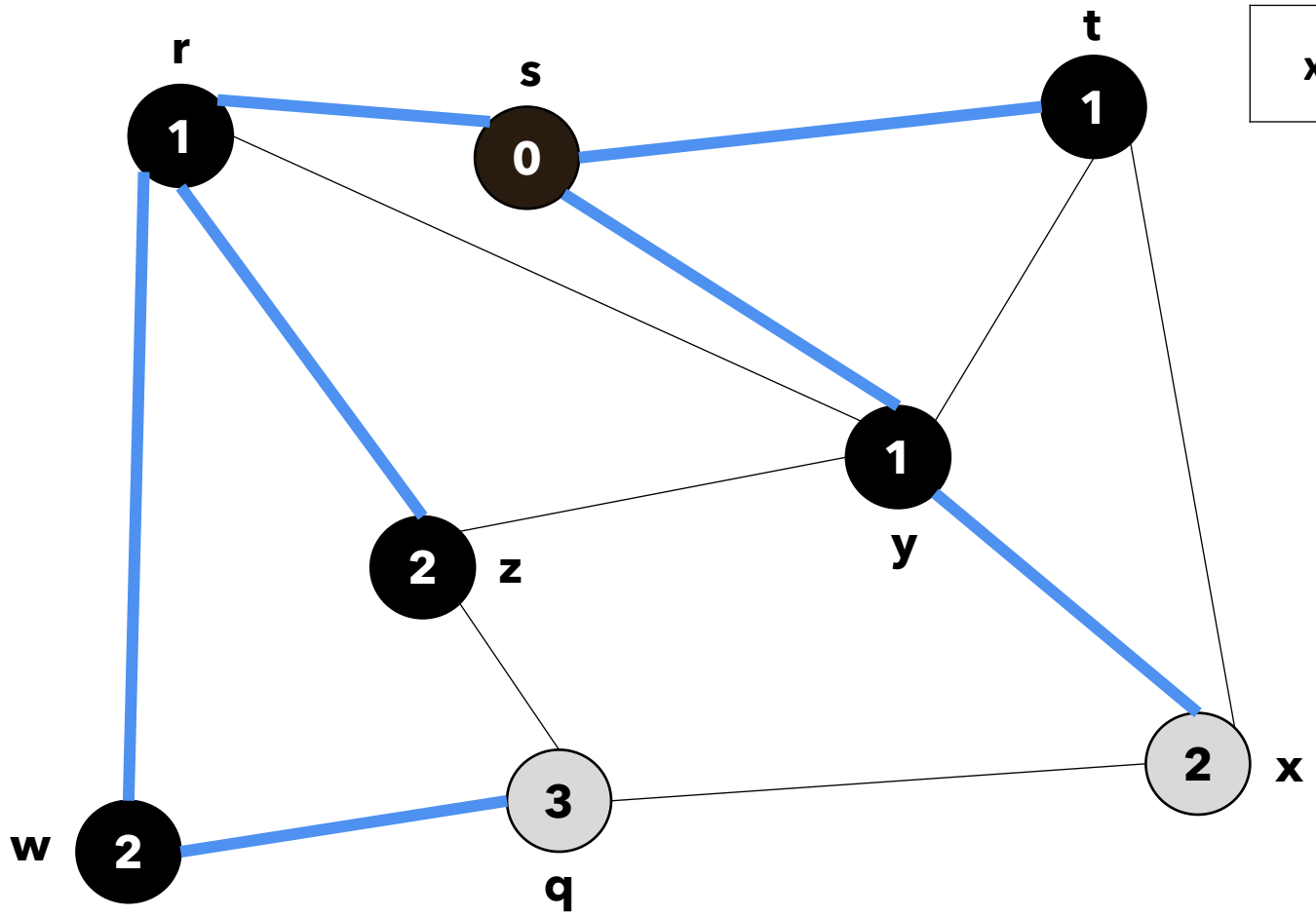
Breadth-first search



z	x	q					

colour w black

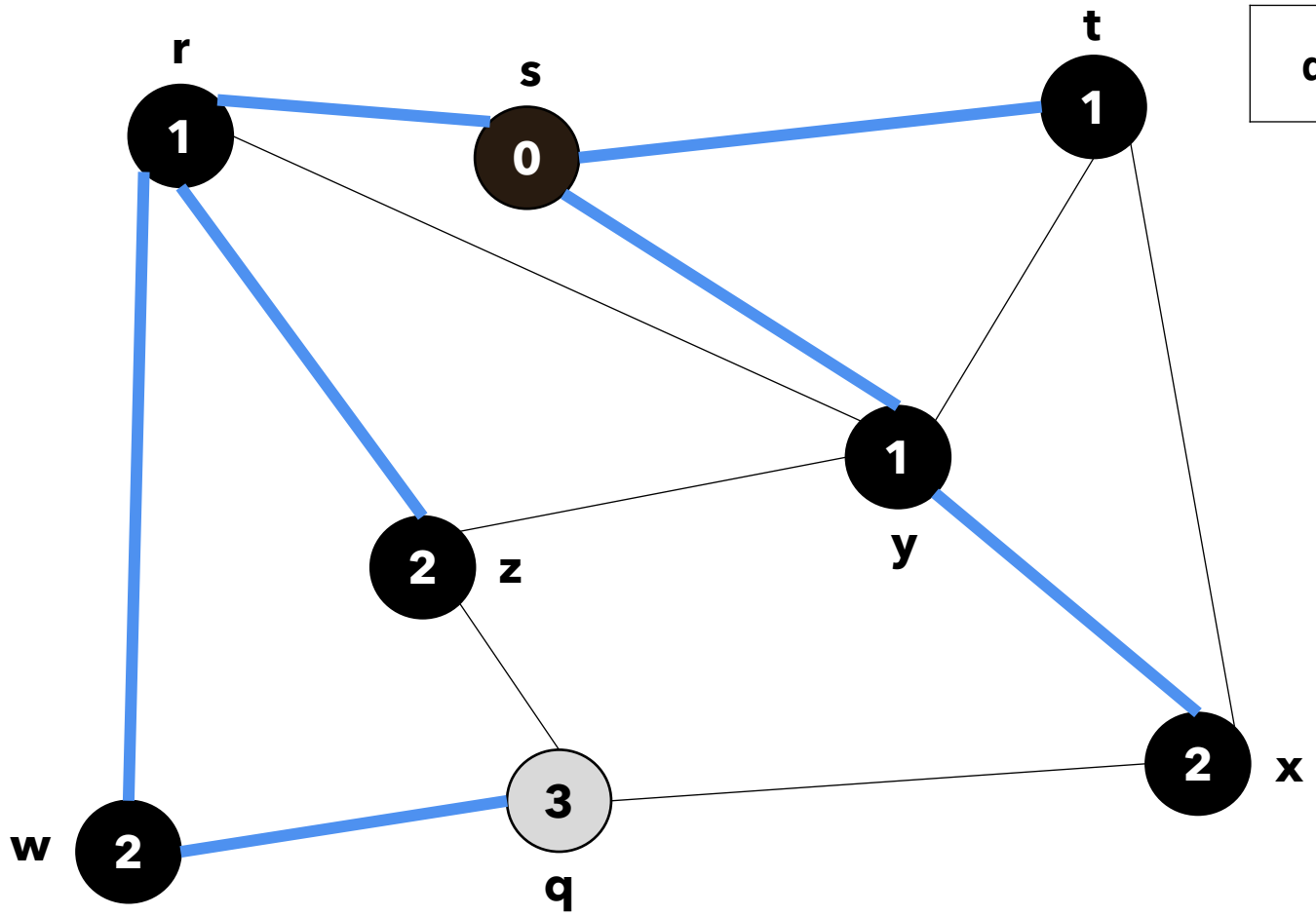
Breadth-first search



x	q						

```
dequeue -> z

colour z black
```

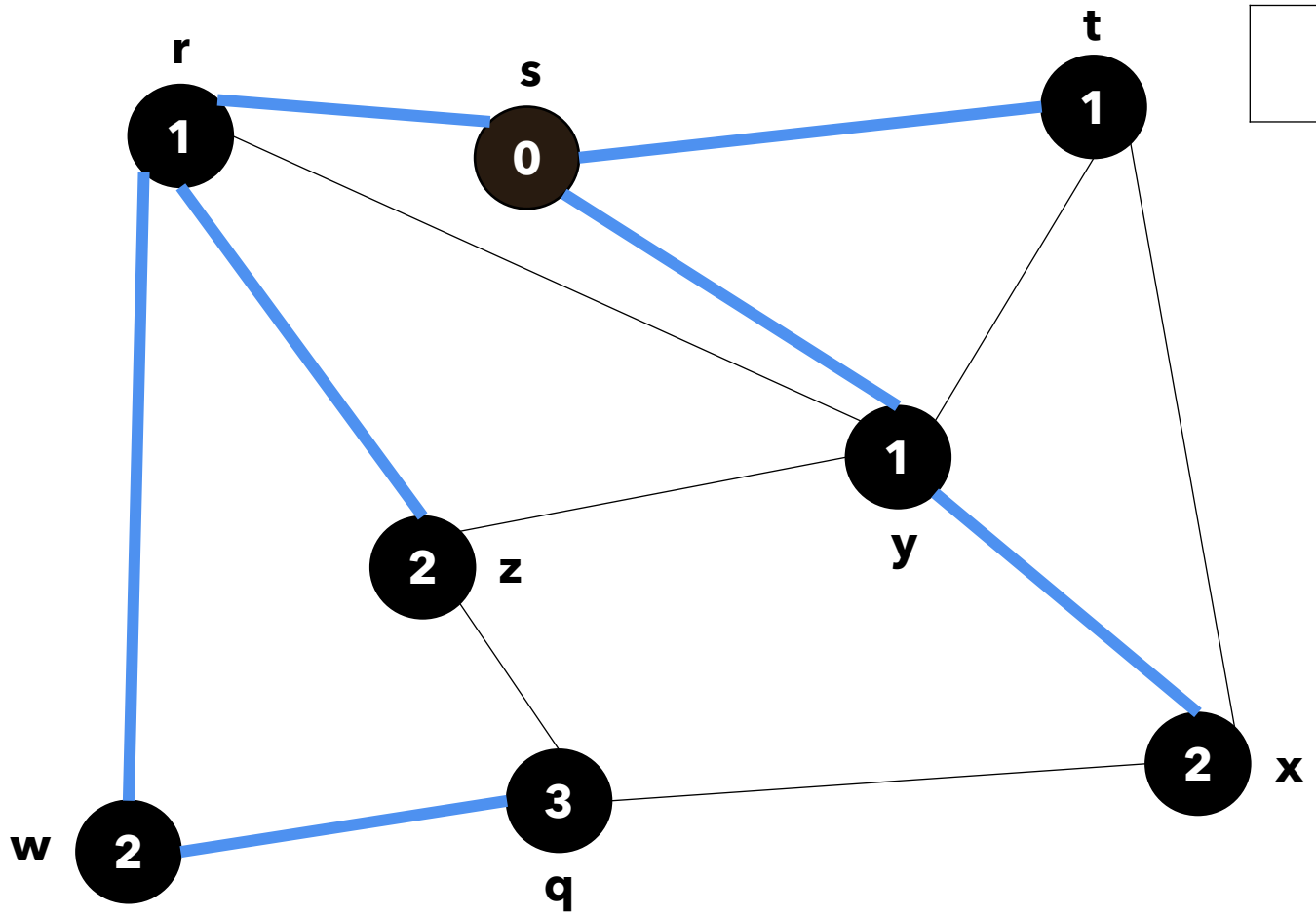
Breadth-first search

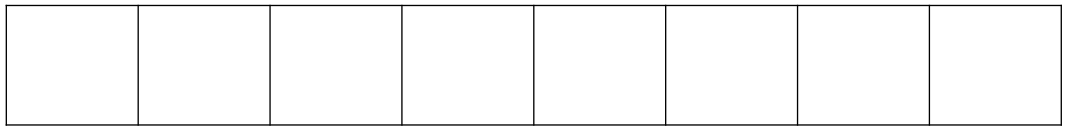


q							

```
dequeue -> x

colour x black
```

Breadth-first search





```
dequeue -> q

colour q black
```

Breadth-first search

- Implementa la BFS in Python
- Sfrutta le strutture dati che conosci (liste, tuple, dizionari etc…) per rappresentare il grafo e la coda