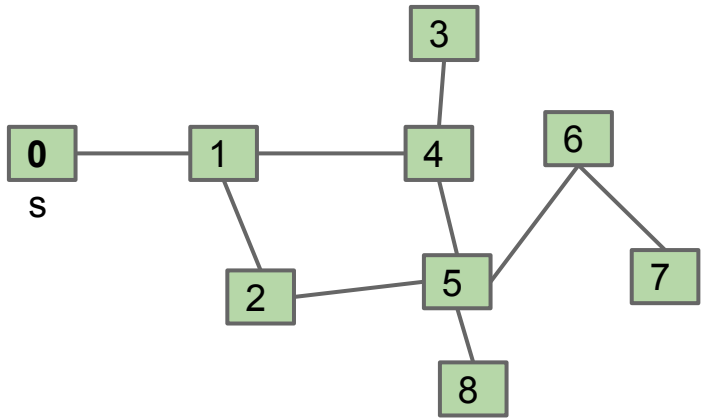
BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	F	-	0
1	F	-	-
2	F	-	-
3	F	-	-
4	F	-	-
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



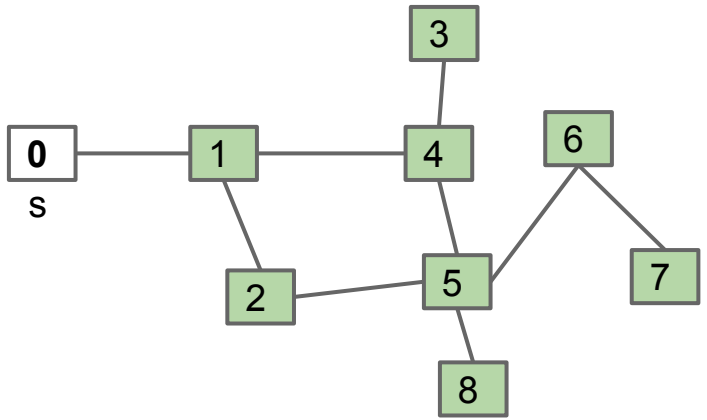
Queue: []

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	F	-	-
2	F	-	-
3	F	-	-
4	F	-	-
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



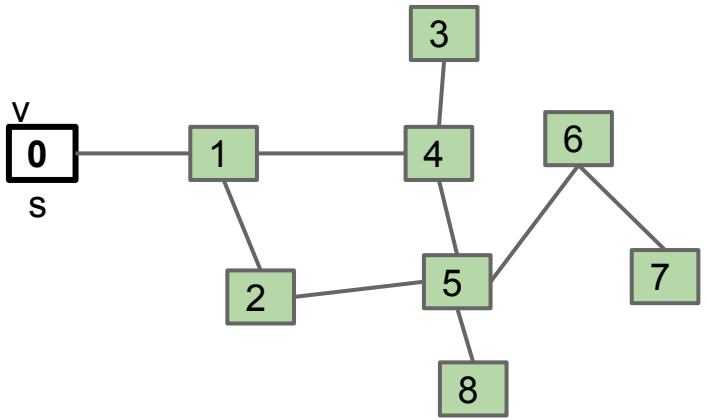
Queue: [0]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	F	-	-
2	F	-	-
3	F	-	-
4	F	-	-
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



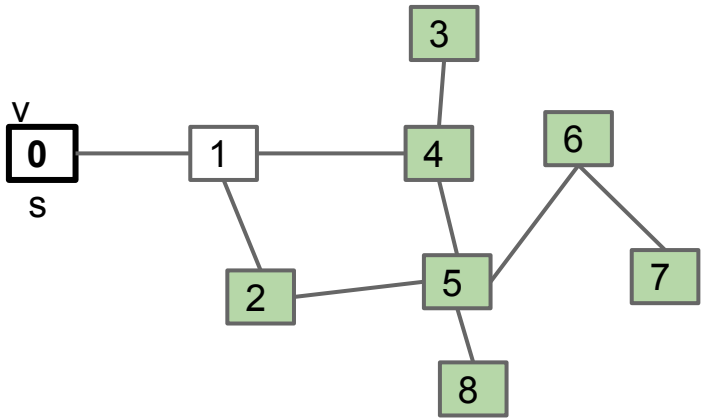
Queue: []

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	F	-	-
3	F	-	-
4	F	-	-
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



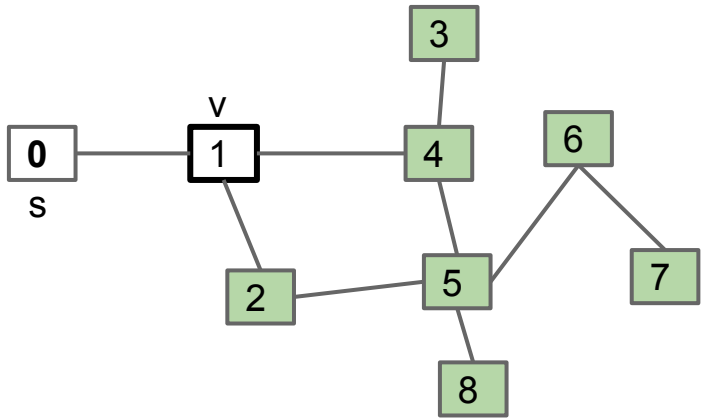
Queue: [1]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	F	-	-
3	F	-	-
4	F	-	-
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



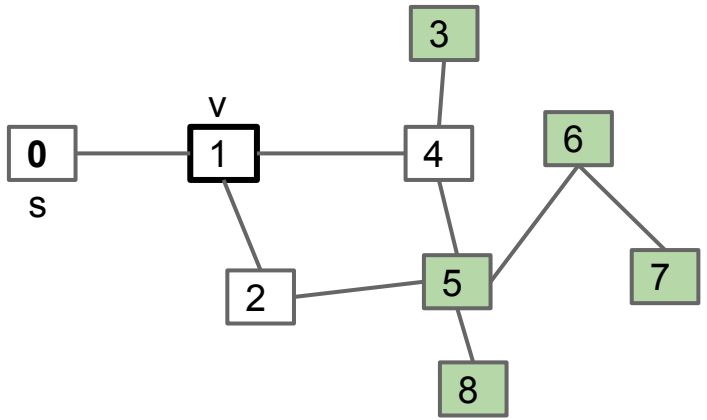
Queue: []

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	F	-	-
4	T	1	2
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



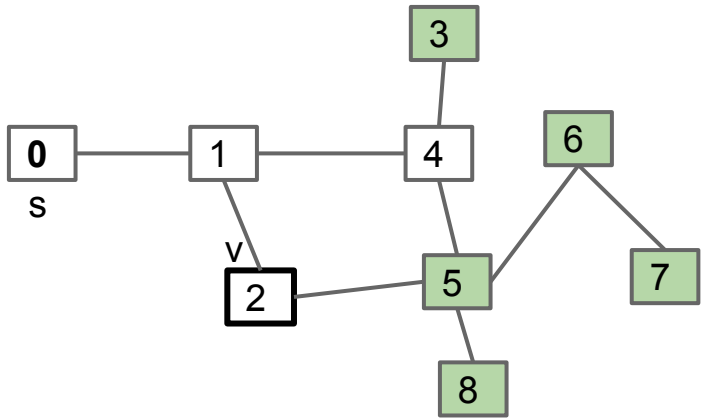
Queue: [2, 4]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	F	-	-
4	T	1	2
5	F	-	-
6	F	-	-
7	F	-	-
8	F	-	-



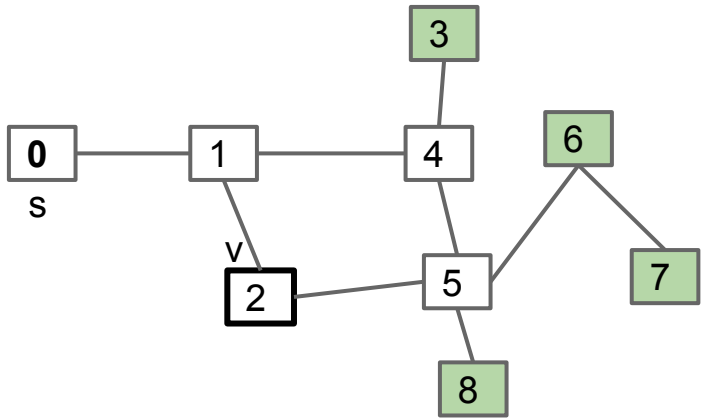
Queue: [4]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	F	-	-
4	T	1	2
5	T	2	3
6	F	-	-
7	F	-	-
8	F	-	-



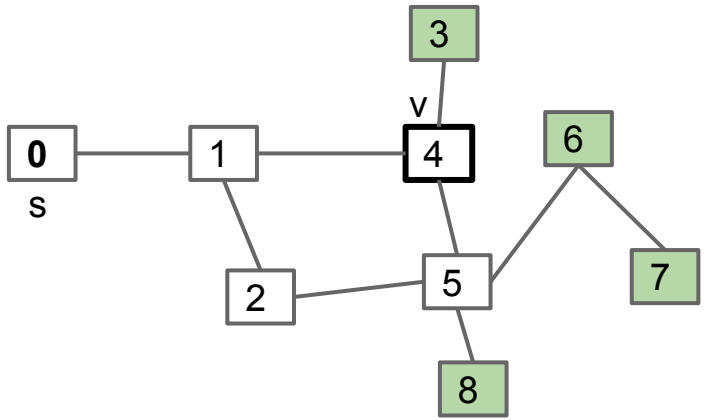
Queue: [4, 5]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	F	-	-
4	T	1	2
5	T	2	3
6	F	-	-
7	F	-	-
8	F	-	-



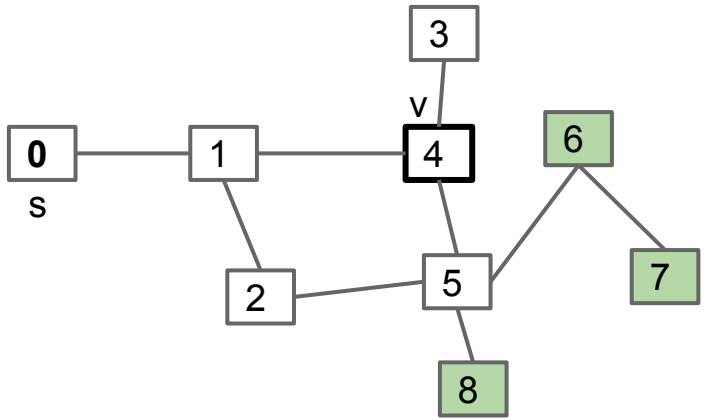
Queue: [5]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	F	-	-
7	F	-	-
8	F	-	-



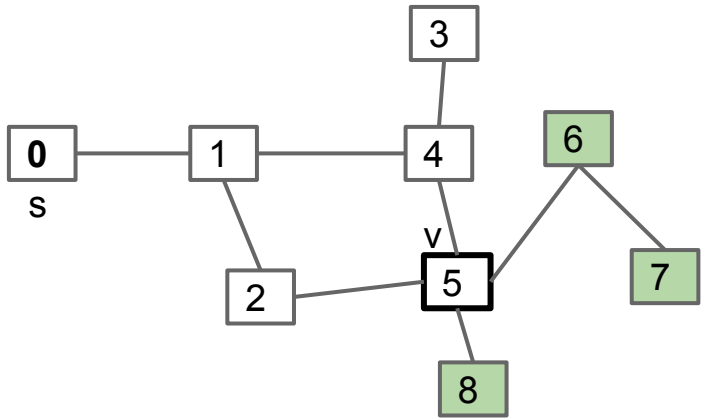
Queue: [5, 3]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	F	-	-
7	F	-	-
8	F	-	-



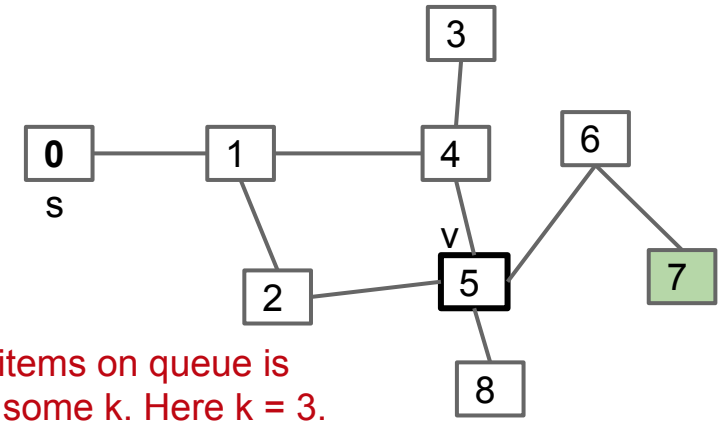
Queue: [3]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	F	-	-
8	T	5	4



Note: distance to all items on queue is always k or k + 1 for some k. Here k = 3.

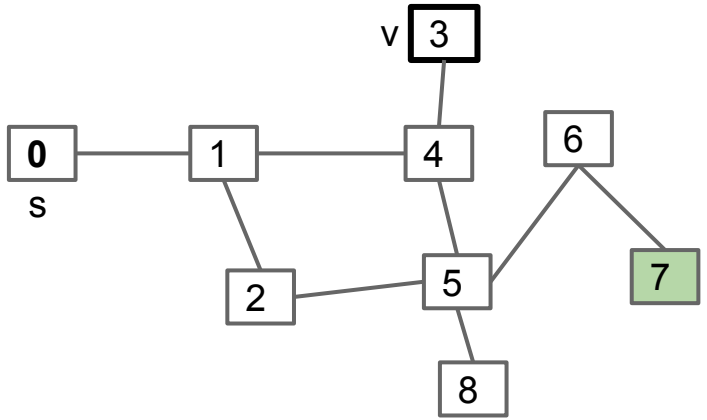
Queue: [3, 6, 8]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	F	-	-
8	T	5	4



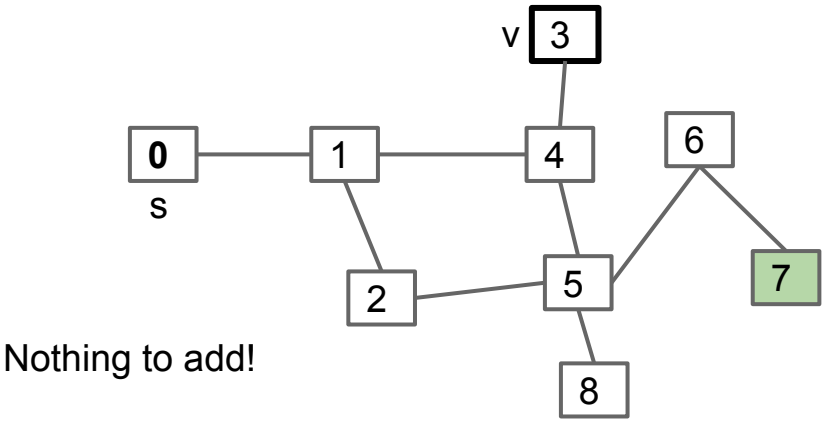
Queue: [6, 8]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	F	-	-
8	T	5	4



Nothing to add!

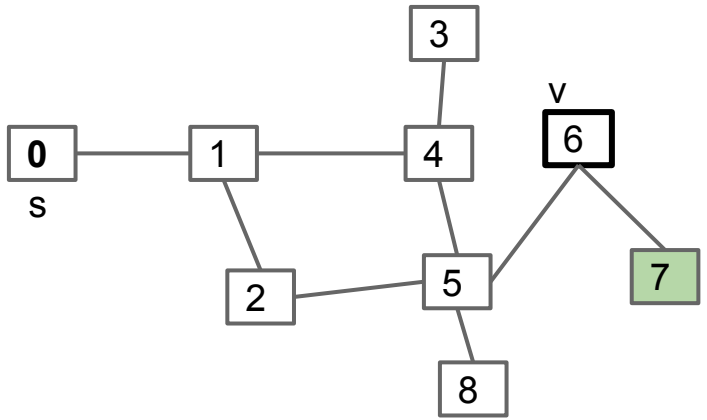
Queue: [6, 8]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	F	-	-
8	T	5	4



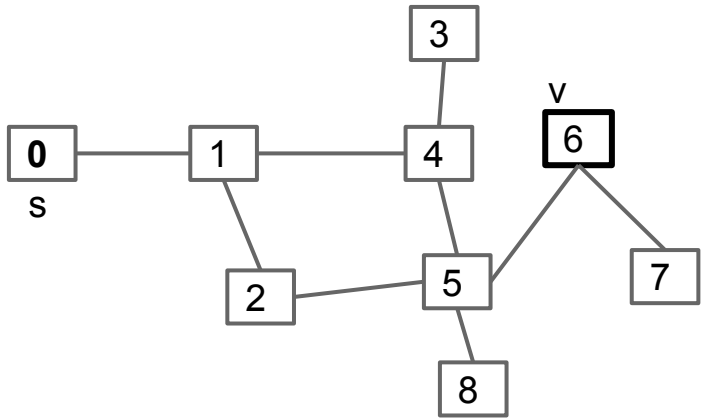
Queue: [8]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	T	6	5
8	T	5	4



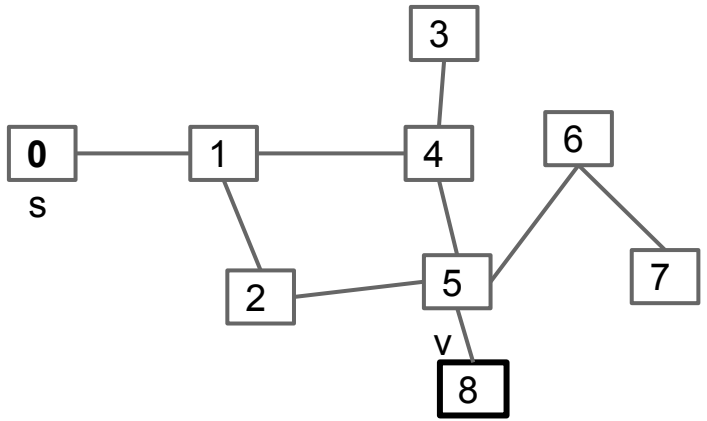
Queue: [8, 7]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	T	6	5
8	T	5	4



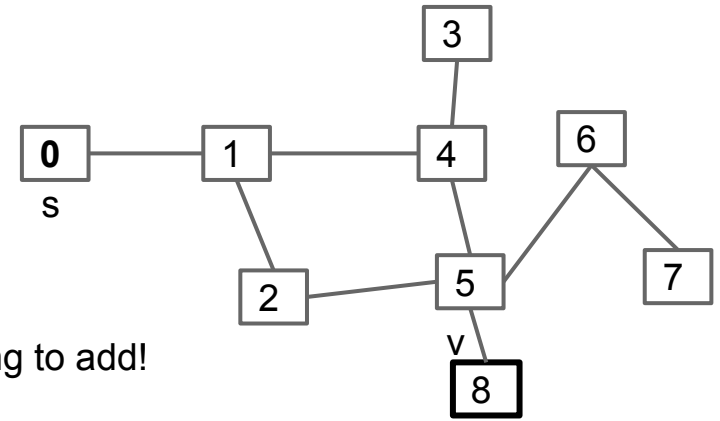
Queue: [7]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	T	6	5
8	T	5	4



Nothing to add!

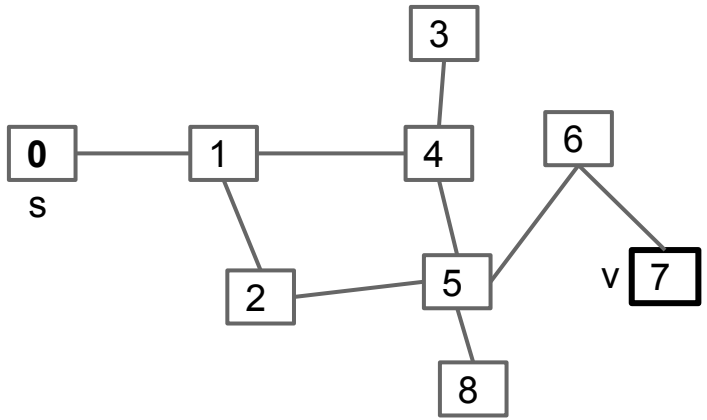
Queue: [7]

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	T	6	5
8	T	5	4



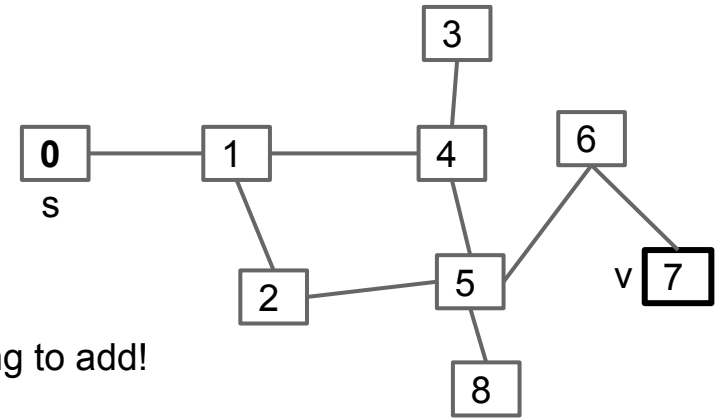
Queue: []

BreadthFirstPaths Demo

Goal: Find shortest path between s and every other vertex.

- **Initialize the fringe (a queue with a starting vertex s) and mark that vertex.**
- Repeat until fringe is empty:
  - Remove vertex v from fringe.
  - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

#	marked	edgeTo	distTo
0	T	-	0
1	T	0	1
2	T	1	2
3	T	4	3
4	T	1	2
5	T	2	3
6	T	5	4
7	T	6	5
8	T	5	4



Nothing to add!

Queue: []