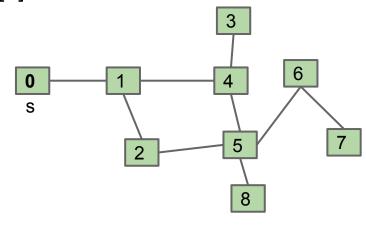
Goal: Find shortest path between s and every other vertex.

- 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.

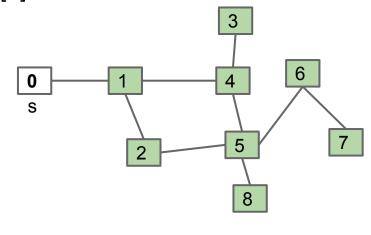
#	marked	edgeTo	distTo	
0	F	-	0	
1	F	-	-	
2	F	-	-	
3	F	-	-	
4	F	-	_	
5	F	-	-	
6	F	-	-	
7	F	-	_	
8	F	-	-	



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.

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

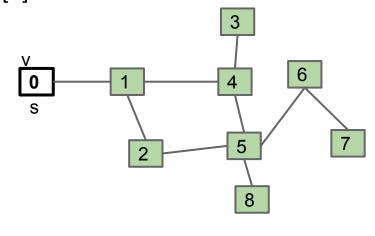




Goal: Find shortest path between s and every other vertex.

- 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.

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

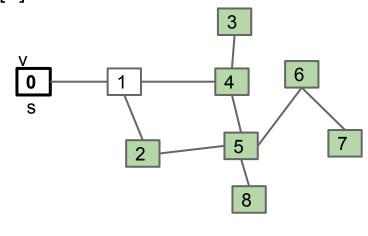




Goal: Find shortest path between s and every other vertex.

- 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.

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

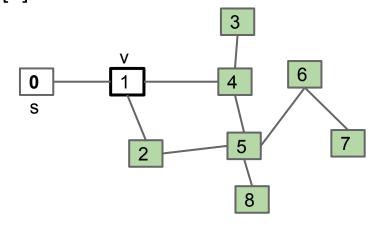




Goal: Find shortest path between s and every other vertex.

- 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.

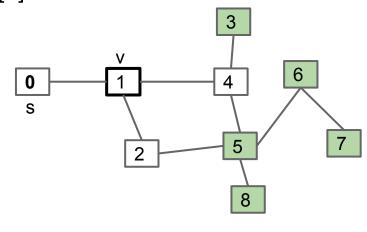
#	marked	edgeTo	distTo	
0	Т	-	0	
1	Т	0	1	
2	F	-	-	
3	F	-	-	
4	F		-	
5	F	-	-	
6	F	-	-	
7	F	-	-	
8	F	-	-	





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.



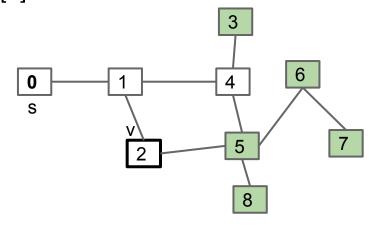
Queue: [2, 4]

datastructur.e

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.

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

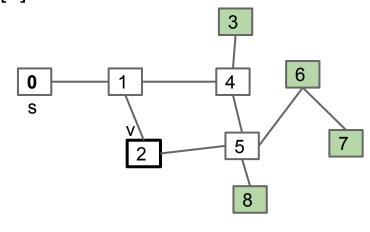




Goal: Find shortest path between s and every other vertex.

- 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.

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

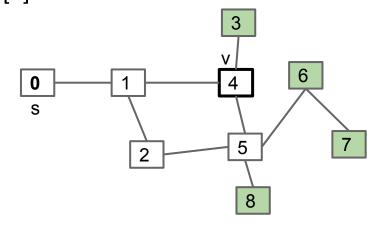


Queue: [4, 5]



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

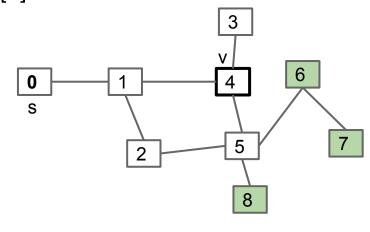
- Initialize the fringe (a queue with a starting vertex s) and mark that vertex.
- Repeat until fringe is empty:
 - Remove vertex v from fringe.
 - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.





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.



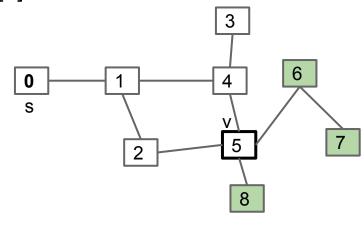
Queue: [5, 3]



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

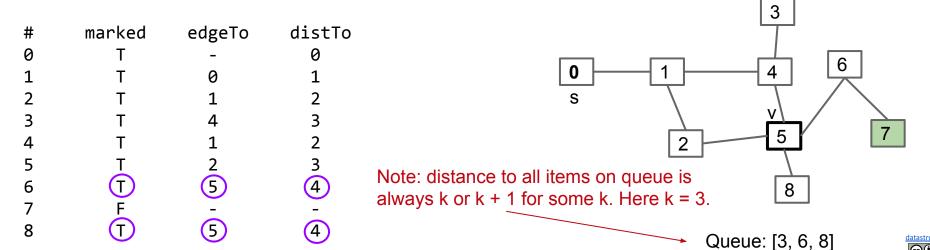
- Initialize the fringe (a queue with a starting vertex s) and mark that vertex.
- Repeat until fringe is empty:
 - Remove vertex v from fringe.
 - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

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





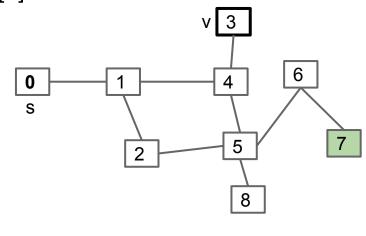
- Initialize the fringe (a queue with a starting vertex s) and mark that vertex.
- Repeat until fringe is empty:
 - Remove vertex v from fringe.
 - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.



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	Т	0	1	
2	Т	1	2	
3	Т	4	3	
4	Т	1	2	
5	Т	2	3	
6	Т	5	4	
7	F	-	-	
8	Т	5	4	



Queue: [6, 8]

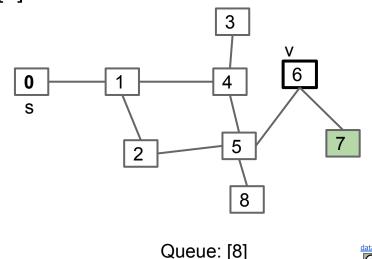


- Initialize the fringe (a queue with a starting vertex s) and mark that vertex.
- Repeat until fringe is empty:
 - Remove vertex v from fringe.
 - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

				v 3
#	marked	edgeTo	distTo	7
0	T	-	0	6
1	Т	0	1	0 1 4 0
2	Т	1	2	s \ \ \ \
3	Т	4	3	
4	Т	1	2	2 5 7
5	Т	2	3	Nothing to addl
6	Т	5	4	Nothing to add!
7	F	-	-	
8	Т	5	4	Queue: [6, 8]
				Chene ip of

- 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.

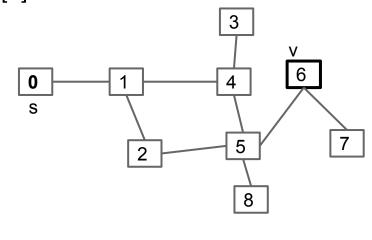
# 0 1 2 3 4 5 6 7	marked T T T T T T T	edgeTo - 0 1 4 1 2 5	distTo 0 1 2 3 2 3 4	
7 8	F T	- 5	- 4	



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.

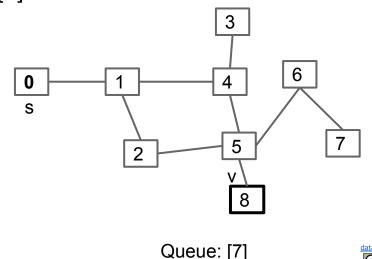
#	marked	edgeTo	distTo	
0	T	-	0	
1	T	0	1	
2	T	1	2	
3	T	4	3	
4	T	1	2	
5	T	2	3	
6	T	5	4	
7	T	<u>6</u>	(5)	
8	Ť	5	4	



Queue: [8, 7]

- 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.

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



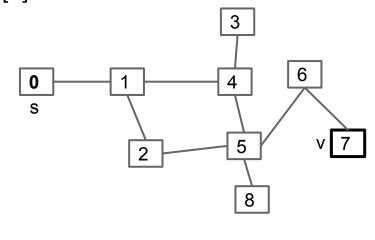
- Initialize the fringe (a queue with a starting vertex s) and mark that vertex.
- Repeat until fringe is empty:
 - Remove vertex v from fringe.
 - For each unmarked neighbor n of v: mark n, add n to fringe, set edgeTo[n] = v, set distTo[n] = distTo[v] + 1.

					3
#	marked	edgeTo	distTo		T
0	Т	-	0		6
1	Т	0	1	0	4 0
2	T	1	2	S	
3	T	4	3		
4	T	1	2		2 5 7
5	T	2	3	Nothing to add!	<u>v\</u>
6	T	5	4	Nothing to add:	8
7	T	6	5		_
8	Т	5	4		Queue: [7]
					GUCUC. III

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.

# 0 1 2 3 4 5 6 7	marked T T T T T T	edgeTo - 0 1 4 1 2 5	distTo 0 1 2 3 2 3 4 5	
	<u>T</u>	•	5	
8	T	5	4	



- 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.

				3
#	marked	edgeTo	distTo	一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一
0	T	-	0	
1	Т	0	1	0 1 4 0
2	Т	1	2	s \
3	T	4	3	
4	T	1	2	2 5 V 7
5	T	2	3	Nothing to addl
6	T	5	4	Nothing to add!
7	T	6	5	
8	Т	5	4	Queue: []
				Queue. []