Minimum Spanning Trees

Problem 1
LMNST!

(Recall) Dijkstra's algorithm pseudocode

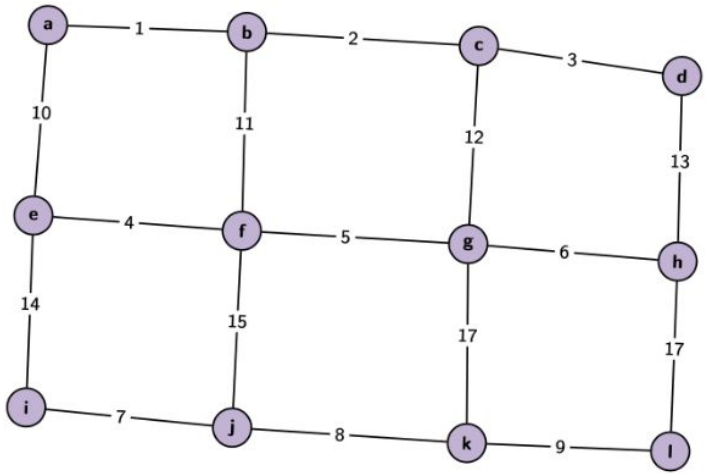
```
dijkstra(Graph G, Node source) {
	Set<Node> processed = new HashSet<>();
	for each Node n in G.nodes() { n.dist = infinity; } // initialize distances (from source) to ∞
	PriorityQueue<Node> fringe = new PriorityQueue<>(); // order by min distance
	source.dist = 0 // it takes 0 step to go from source to itself
	fringe.add(source, 0); // push source in fringe
	while (!fringe.isEmpty()) {
		Node curr = fringe.removeMin();
		if (processed.contains(curr)) continue; // once a node is explored, its STP is settled
		for each Edge (curr,neighbor,w) in curr.outEdges() {
			if (!processed.contains(neighbor) && curr.dist + w < neighbor.dist) {
				fringe.updateOrAddPriority(neighbor, curr.dist + w);
				neighbor.predecessor = curr;
			}
		}
		processed.add(curr); // mark curr node as processed
	}
}
```

Prim's algorithm pseudocode

```
primMST(Graph G) {
	Graph MST = new Graph<>();
	Node start = any Node n in G.nodes()
	Set<Node> processed = new HashSet<>();
	for each Node n in G.nodes() { n.dist = infinity; } // dist here is just the cost of edge
	PriorityQueue<Node> fringe = new PriorityQueue<>(); // order by min distance
	start.dist = 0 // 0 weight to come to the start
	fringe.add(start, 0); // push start in fringe
	while (!fringe.isEmpty()) {
		Node curr = fringe.removeMin();
		if (processed.contains(curr)) continue; // once a node is explored, its best edge is settled and is in MST
		if (!curr.equals(start)) MST.addEdge(curr.bestEdge);
		for each Edge (curr,neighbor,w) in curr.outEdges() {
			if (!processed.contains(neighbor) && w < neighbor.dist) {
				fringe.updateOrAddPriority(neighbor, w);
				neighbor.bestEdge = (curr,neighbor,w);
			}
		}
		processed.add(curr); // mark curr node as processed
	}}
```

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



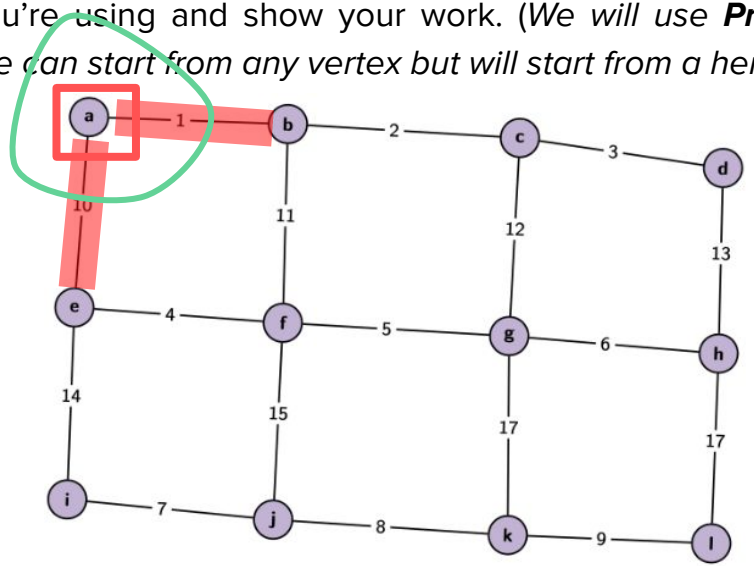
Vertex	Processed?	Cost of Edge	Best Edge
a	N	0	-
b	N	∞	
c	N	∞	
d	N	∞	
e	N	∞	
f	N	∞	
g	N	∞	
h	N	∞	
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

(a,0)							

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	N	~~∞~~ 1	(a,b)
c	N	∞	
d	N	∞	
e	N	~~∞~~ 10	(a,e)
f	N	∞	
g	N	∞	
h	N	∞	
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	**(b,1)**	(e,10)					

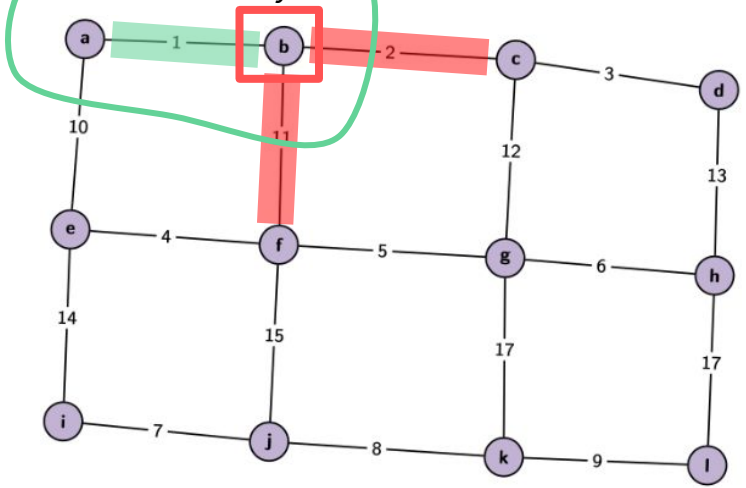
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	N	~~∞~~ 2	(b,c)
d	N	∞	
e	N	~~∞~~ 10	(a,e)
f	N	~~∞~~ 11	(b,f)
g	N	∞	
h	N	∞	
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	(e,10)	**(c,2)**	(f,11)			

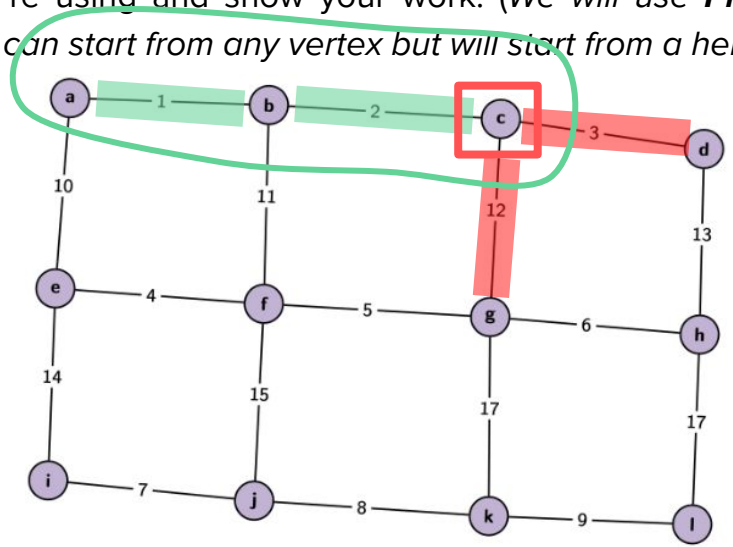
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	N	~~∞~~ 3	(c,d)
e	N	~~∞~~ 10	(a,e)
f	N	~~∞~~ 11	(b,f)
g	N	~~∞~~ 12	(c,g)
h	N	∞	
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	(e,10)	~~(c,2)~~	(f,11)	**(d,3)**	(g,12)	

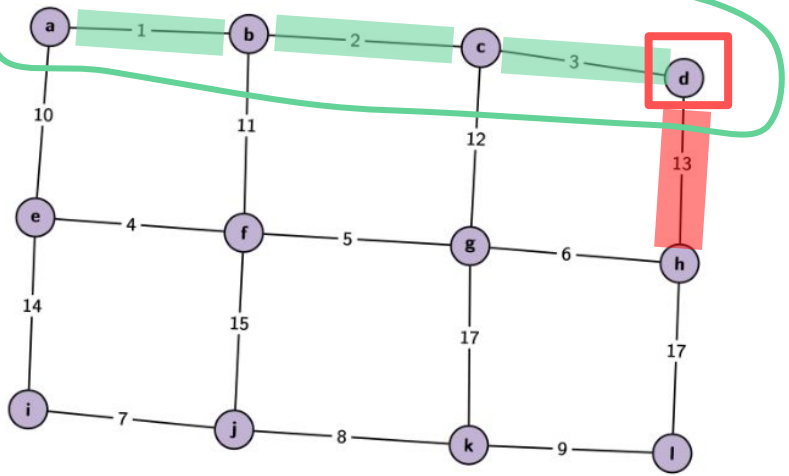
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	N	~~∞~~ 10	(a,e)
f	N	~~∞~~ 11	(b,f)
g	N	~~∞~~ 12	(c,g)
h	N	~~∞~~ 13	(d,h)
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	**(e,10)**	~~(c,2)~~	(f,11)	~~(d,3)~~	(g,12)	(h,13)

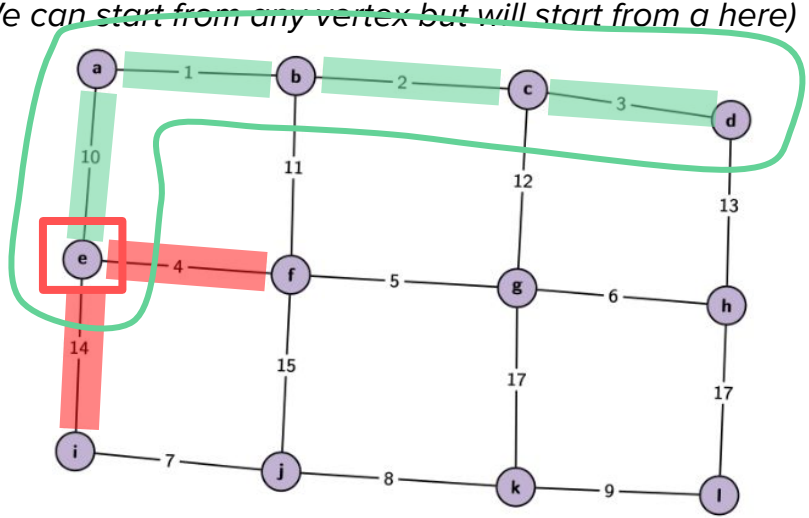
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	N	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	N	~~∞~~ 12	(c,g)
h	N	~~∞~~ 13	(d,h)
i	N	~~∞~~ 14	(e,i)
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(e,2)~~	**(f,4)**	~~(d,3)~~	(g,12)	(h,13)
(i,14)							

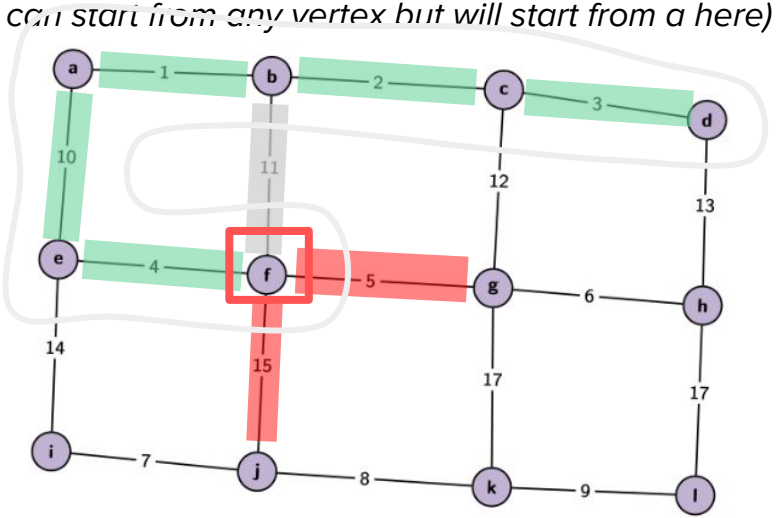
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	N	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	N	~~∞~~ 13	(d,h)
i	N	~~∞~~ 14	(e,i)
j	N	~~∞~~ 15	(f,j)
k	N	∞	
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(c,2)~~	~~(f,4)~~	~~(d,3)~~	**(g,5)**	(h,13)
(i,14)	(j,15)						

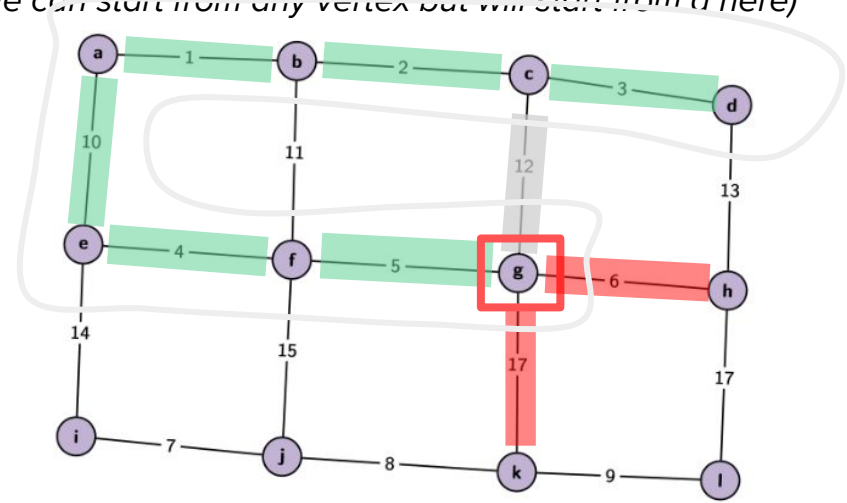
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	N	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	N	~~∞~~ 14	(e,i)
j	N	~~∞~~ 15	(f,j)
k	N	~~∞~~ 17	(g,k)
l	N	∞	

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(e,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	**(h,6)**
(i,14)	(j,15)	(k,17)					

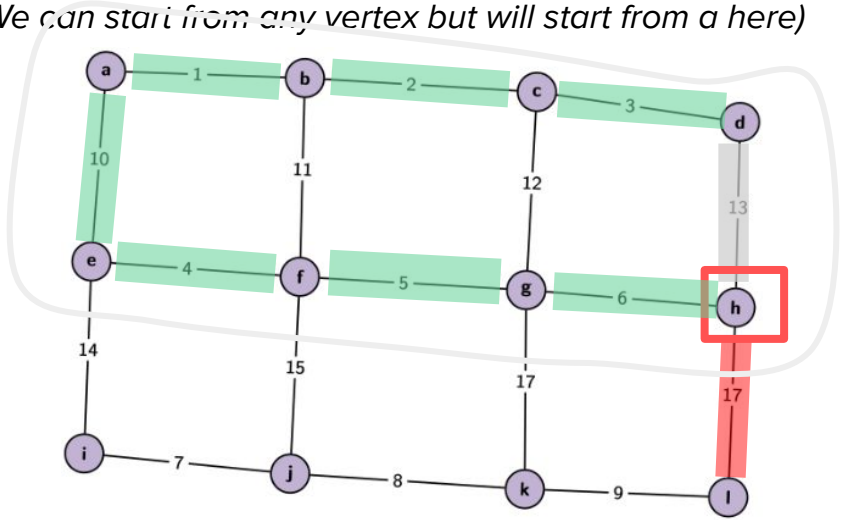
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	N	~~∞~~ 14	(e,i)
j	N	~~∞~~ 15	(f,j)
k	N	~~∞~~ 17	(g,k)
l	N	~~∞~~ 17	(h,l)

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(c,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	~~(h,6)~~
(i,14)	(j,15)	(k,17)	(l,17)				

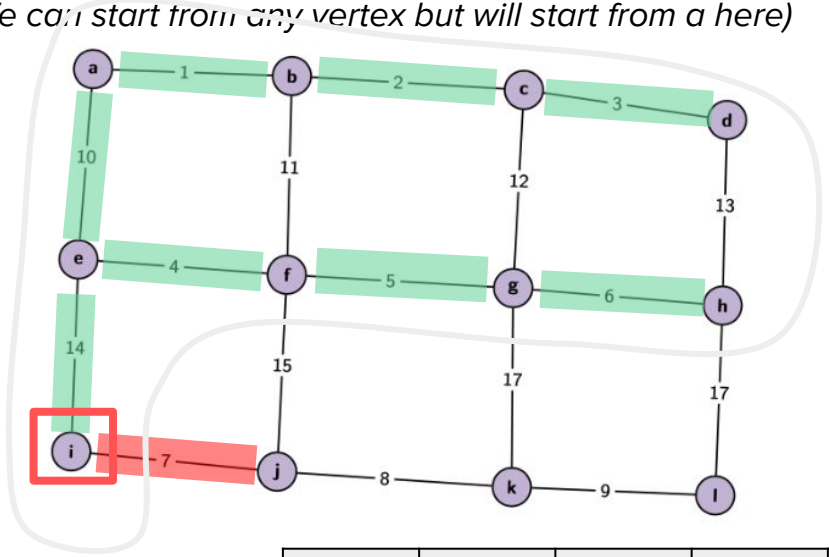
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	N	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	N	~~∞~~ 17	(g,k)
l	N	~~∞~~ 17	(h,l)

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(c,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	~~(h,6)~~
~~(i,14)~~	**(j,7)**	(k,17)	(l,17)				

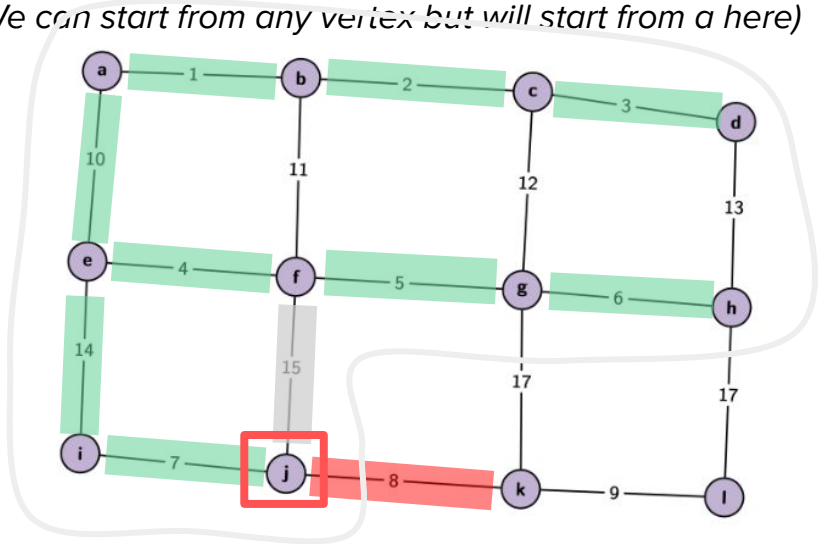
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	N	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	N	~~∞~~ 17	(h,l)

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(e,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	~~(h,6)~~
~~(i,14)~~	~~(j,7)~~	**(k,8)**	(l,17)				

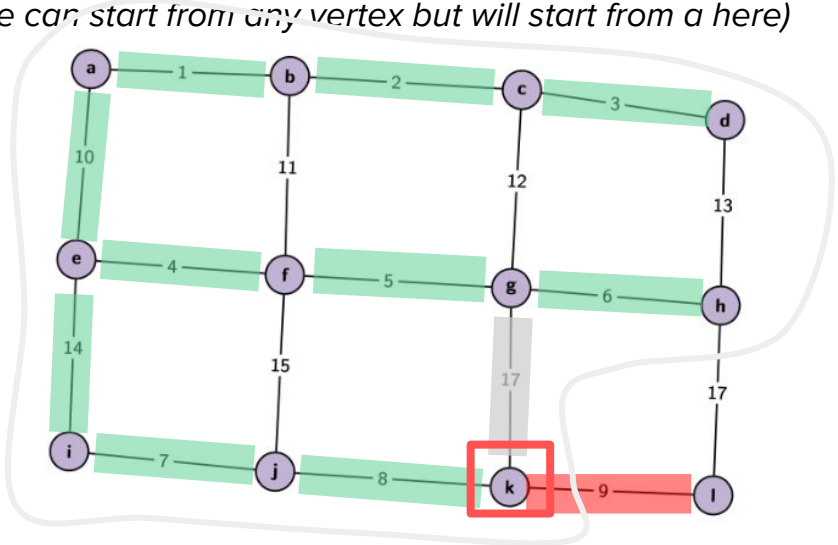
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	~~N~~ Y	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	N	~~∞~~ ~~17~~ 9	~~(h,l)~~ (k,l)

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(c,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	~~(h,6)~~
~~(i,14)~~	~~(j,7)~~	~~(k,8)~~	**(l,9)**				

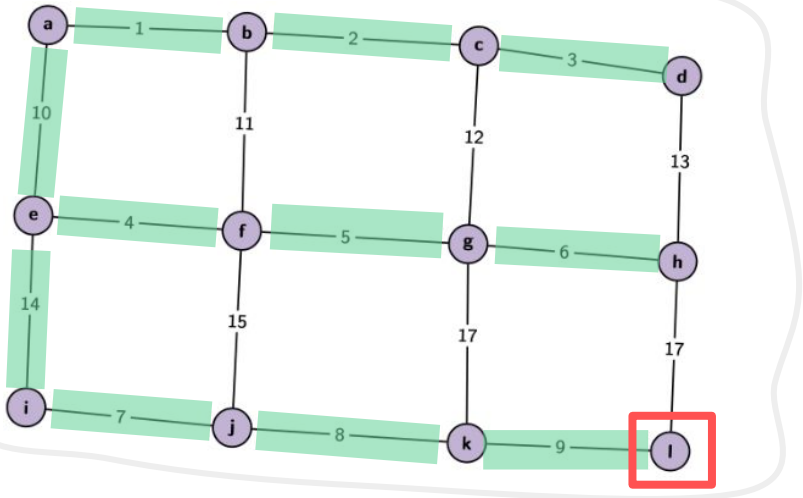
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	~~N~~ Y	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	~~N~~ Y	~~∞~~ ~~17~~ 9	~~(h,l)~~ (k,l)

Priority Queue:

~~(a,0)~~	~~(b,1)~~	~~(e,10)~~	~~(c,2)~~	~~(f,4)~~	~~(d,3)~~	~~(g,5)~~	~~(h,6)~~
~~(i,14)~~	~~(j,7)~~	~~(k,8)~~	~~(l,9)~~				

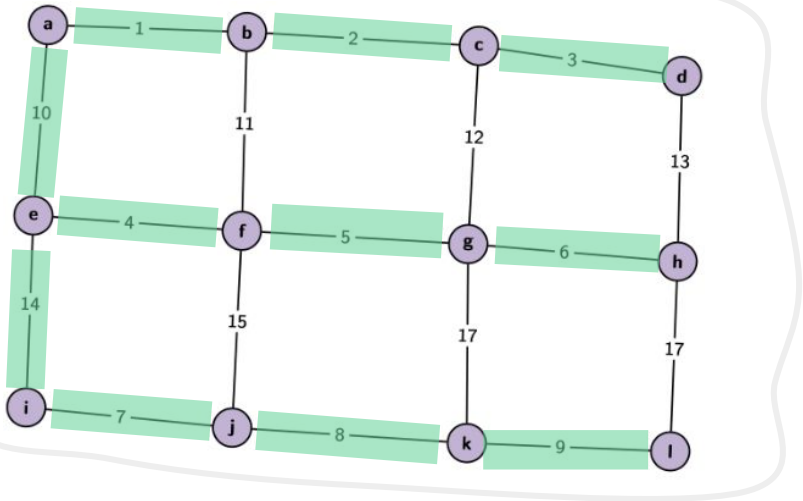
Add/Update priority

No need to update priority

Pop the node and settle the best edge coming into it

Q1) LMNST!

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*

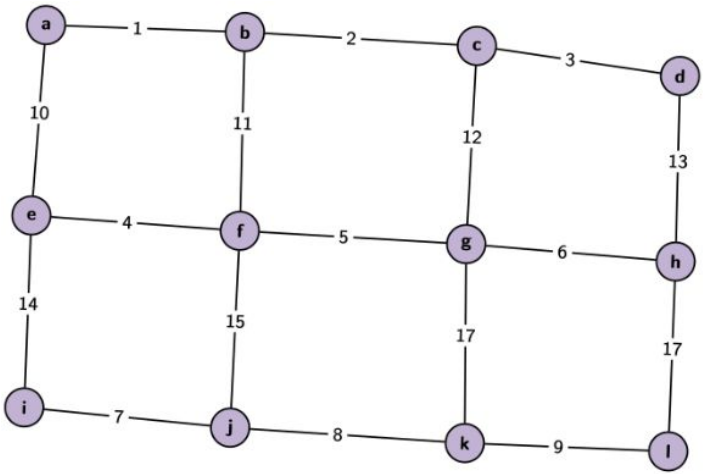


Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	~~N~~ Y	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	~~N~~ Y	~~∞~~ ~~17~~ 9	~~(h,l)~~ (k,l)

Total Weight of the MST: 1 + 2 + 3 + 10 + 4 + 5 + 6 + 14 + 7 + 8 + 9 = 69

Q1) LMNST! - Breakout 1

Problem: Find an MST of this graph. Say which algorithm you're using and show your work. (*We will use **Prim's**! We can start from any vertex but will start from a here)*



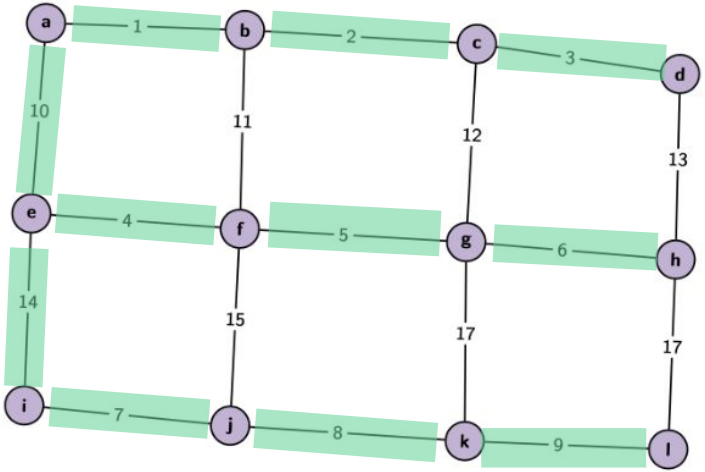
Vertex	Processed?	Cost of Edge	Best Edge
a	N	0	-
b	N	∞	
c	N	∞	
d	N	∞	
e	N	∞	
f	N	∞	
g	N	∞	
h	N	∞	
i	N	∞	
j	N	∞	
k	N	∞	
l	N	∞	

Priority Queue:

(a,0)							

Q1b) LMNST!

b) Using just the graph, how can you determine if it's possible that there are multiple MSTs of the graph? Does this graph have multiple MSTs?

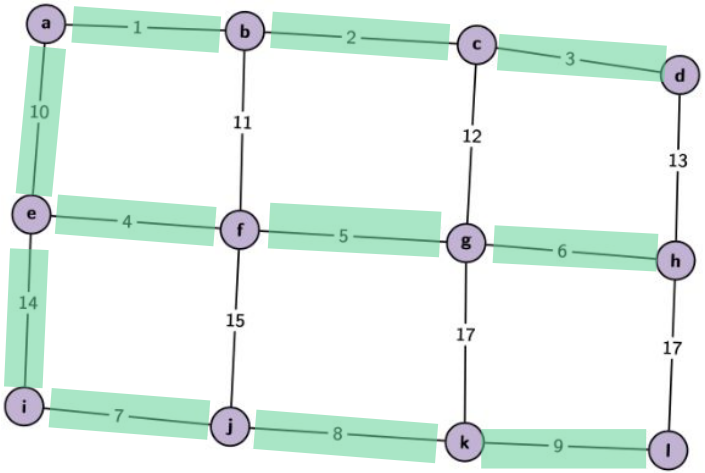


Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	~~N~~ Y	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	~~N~~ Y	~~∞~~ ~~17~~ 9	~~(h,l)~~ (k,l)

A graph can only have multiple MSTs if it has **multiple edges of the same weight**. This graph has two 17's, but neither of them are used in the MST. So, there's only one MST here.

Q1c) LMNST!

c) What is the asymptotic runtime of the algorithms that you used to compute the MSTs?



Vertex	Processed?	Cost of Edge	Best Edge
a	~~N~~ Y	0	-
b	~~N~~ Y	~~∞~~ 1	(a,b)
c	~~N~~ Y	~~∞~~ 2	(b,c)
d	~~N~~ Y	~~∞~~ 3	(c,d)
e	~~N~~ Y	~~∞~~ 10	(a,e)
f	~~N~~ Y	~~∞~~ ~~11~~ 4	~~(b,f)~~ (e,f)
g	~~N~~ Y	~~∞~~ ~~12~~ 5	~~(c,g)~~ (f,g)
h	~~N~~ Y	~~∞~~ ~~13~~ 6	~~(d,h)~~ (g,h)
i	~~N~~ Y	~~∞~~ 14	(e,i)
j	~~N~~ Y	~~∞~~ ~~15~~ 7	~~(f,j)~~ (i,j)
k	~~N~~ Y	~~∞~~ ~~17~~ 8	~~(g,k)~~ (j,k)
l	~~N~~ Y	~~∞~~ ~~17~~ 9	~~(h,l)~~ (k,l)

Prim's algorithm - similar to Dijkstra's, it takes ***O(\|V\|log\|V\| + \|E\|log\|V\|)***
Kruskal's algorithm - sort edges by weights and use union-find to detect cycle while adding edges into MST. It takes ***O(\|E\|log\|E\| + E*O(log*(\|V\|)) + V*O(log*(\|V\|)))*** ≃ ***O(\|E\|log\|E\|)***, assuming path-compression UF is used.