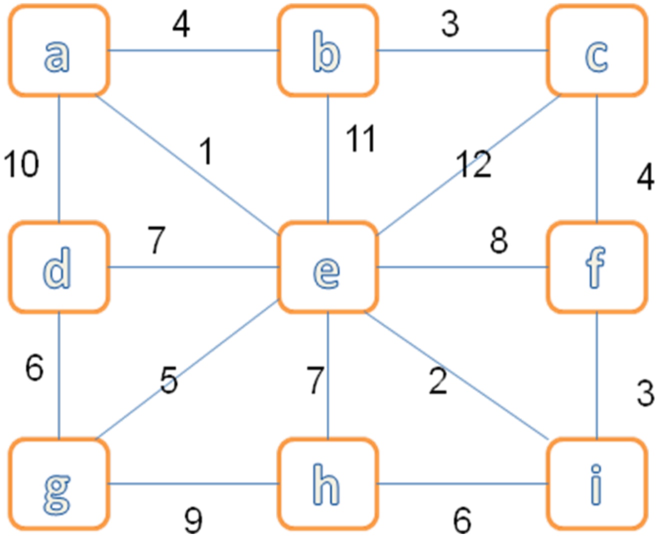
1. b.

	Average
Prim's MST	O(|E|log|V|)
Kruskal's MST	O(|E|log|V|)
Dijkstra's	O(|V|log|V| +|E|log|V|) Or |V|^2
Topological sort	O(|E| + |V|)

2.

a.



b. We can show the process in Prim's via a table. Start with an empty table.

Vertex	Known	Distance	Path
A	F		
B	F		
C	F		
D	F		
E	F		
F	F		
G	F		
H	F		
I	F		

Say we start at vertex A; explore ‘A’: mark as ‘Known’, update distance of any adjacent edges that are 1) not known, and 2) have edge weight < current distance value in table – mark the ‘Path’ of those updated nodes to be the node we are exploring.

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F		
D	F	10	a
E	F	1	A
F	F		
G	F		
H	F		
I	F		

Explore ‘E’

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F	12	E
D	F	7	E
E	T	1	A
F	F	8	E
G	F	5	E
H	F	7	E
I	F	2	E

Explore ‘I’

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F	12	E
D	F	7	E
E	T	1	A
F	F	3	I
G	F	5	E
H	F	6	I
I	T	2	E

Explore ‘F’

Vertex	Known	Distance	Path

A	T	0	-
B	F	4	A
C	F	4	F
D	F	7	E
E	T	1	A
F	T	3	I
G	F	5	E
H	F	6	I
I	T	2	E

Explore ‘B’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	F	3	B
D	F	7	E
E	T	1	A
F	T	3	I
G	F	5	E
H	F	6	I
I	T	2	E

Explore ‘C’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	3	B
D	F	7	E
E	T	1	A
F	T	3	I
G	F	5	E
H	F	6	I
I	T	2	E

Explore ‘G’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	3	B
D	F	6	G

E	T	1	A
F	T	3	I
G	T	5	E
H	F	6	I
I	T	2	E

Explore ‘D’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	3	B
D	T	6	G
E	T	1	A
F	T	3	I
G	T	5	E
H	F	6	I
I	T	2	E

Explore ‘H’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	3	B
D	T	6	G
E	T	1	A
F	T	3	I
G	T	5	E
H	T	6	I
I	T	2	E

The MST is then: AB:4, AE:1,BC:3, DG:6, EG:5, EI:2, FI:3, HI:6
Total cost: 30

c.

Kruskal’s: Order the list of edges of the graph; step through and accept an edge if it’s two vertices are not connected (we’d keep track of it using the union/find data structure).

Edge	Cost	Accepted
AE	1	Yes
EI	2	Yes
BC	3	Yes

FI	3	Yes
AB	4	Yes
CF	4	No
GE	5	Yes
HI	6	Yes
DG	6	Yes
DE	7	
EH	7	
EF	8	
GH	9	
AD	10	
BE	11	
CE	12	

The MST is then: AB:4, AE:1, BC:3, DG:6, EI:2, FI:3, GE:5, HI:6

Cost: 30

This turns out to be the same MST as we got with Prim's, but we can get a different one by swapping AB with CF; had our ordering of the edges for the above table been different, we may have ended up with this one instead.

d. We can use a table similar that used for Prim's to find the shortest distance from A to every other vertex.

Vertex	Known	Distance	Path
A	F		
B	F		
C	F		
D	F		
E	F		
F	F		
G	F		
H	F		
I	F		

Explore 'A'; very similar to Prim's, but here the 'Distance' is not merely the edge weight, but is the edge weight plus the current path length.

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F		
D	F	10	A
E	F	1	A

F	F		
G	F		
H	F		
I	F		

Explore ‘E’

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F	13	E
D	F	8	E
E	T	1	A
F	F	9	E
G	F	6	E
H	F	8	E
I	F	3	E

Explore ‘I’

Vertex	Known	Distance	Path
A	T	0	-
B	F	4	A
C	F	13	E
D	F	8	E
E	T	1	A
F	F	6	I
G	F	6	E
H	F	8	E
I	T	3	E

Explore ‘B’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	F	7	B
D	F	8	E
E	T	1	A
F	F	6	I
G	F	6	E
H	F	8	E
I	T	3	E

Explore ‘F’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	F	7	B
D	F	8	E
E	T	1	A
F	T	6	I
G	F	6	E
H	F	8	E
I	T	3	E

Explore ‘G’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	F	7	B
D	F	8	E
E	T	1	A
F	T	6	I
G	T	6	E
H	F	8	E
I	T	3	E

Explore ‘C’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	7	B
D	F	8	E
E	T	1	A
F	T	6	I
G	T	6	E
H	F	8	E
I	T	3	E

Explore ‘D’

Vertex	Known	Distance	Path
A	T	0	-

B	T	4	A
C	T	7	B
D	T	8	E
E	T	1	A
F	T	6	I
G	T	6	E
H	F	8	E
I	T	3	E

Explore ‘H’

Vertex	Known	Distance	Path
A	T	0	-
B	T	4	A
C	T	7	B
D	T	8	E
E	T	1	A
F	T	6	I
G	T	6	E
H	T	8	E
I	T	3	E

Now we’re done.

We can find the shortest path A to an arbitrary vertex X by starting at X and tracing it’s path backward using the ‘Path’ column of the table.

For instance, to find the path from A to F, start at F, look up its path value I; now check I’s, which is E; now check E’s, which is A. So the path is AEIF.