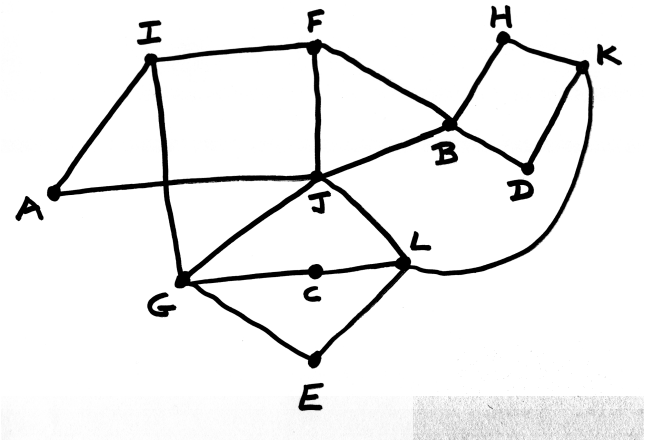
Shortest Paths (Dijkstra's Algorithm)

1. For each of the graphs below (one undirected, the second directed) find the shortest distances from vertex A to all other vertices. (Note that the edges $\{I, G\}$ and $\{A, J\}$ cross each other, but there is not a vertex at the point of intersection). For each graph, draw the subgraph that consist of edges that are used in the shortest paths.

   You should find *both* the shortest distances *and* the predecessor array which will allow us to reconstruct a path joining A to any vertex.

   For each of these two graphs, the weight of every edge is 1 (and hence, that's why I haven't included the weights in the diagram).



   As for any problem using Dijkstra's algorithm, I will maintain a table for the shortest distances. In fact, I will maintain two elements in the table, the (current) shortest distance and the predecessor of a vertex. Both of these items could be updated in each step of the algorithm. The predecessor array lets us reconstruct the shortest path from vertex A to any other one, by tracing backwards through those values.

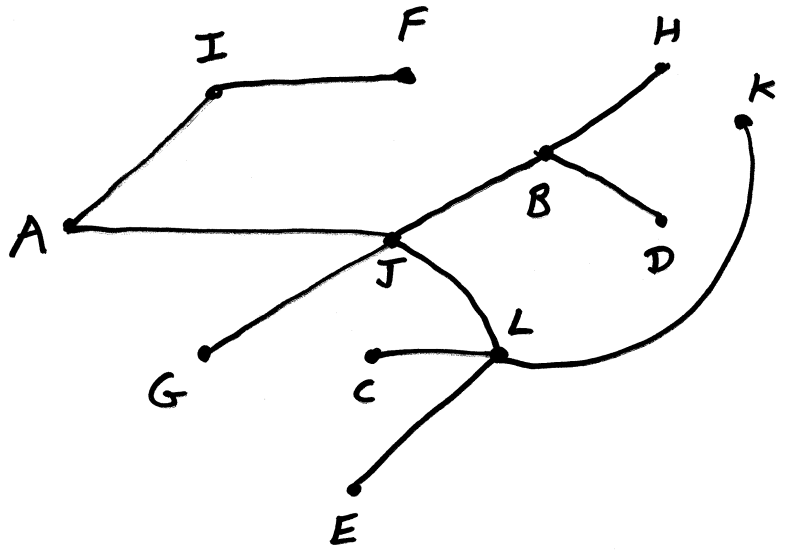
   Note that I will not show the list of elements whose distance values are fixed (permanent) at each step, but will merely show which new one becomes fixed in each step. Of course, once a value becomes fixed, it won't decrease further (why?), and we won't consider it again in later steps to determine the new vertex to fix. Recall that we also take the minimum (breaking ties arbitrarily) of the vertices that **are not fixed** at each step.

   The values of the predecessors are initially undefined, and I will only show them once the distance value becomes finite.

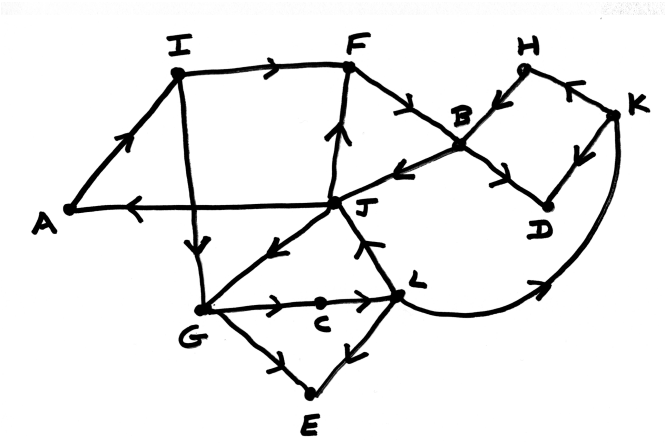
fixed	A	B	C	D	E	F	G	H	I	J	K	L
	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
A	0,$\emptyset$	∞	∞	∞	∞	∞	∞	∞	1, A	1, A	∞	∞
I	0,$\emptyset$	∞	∞	∞	∞	2, I	∞	∞	1, A	1, A	∞	∞
J	0,$\emptyset$	2, J	∞	∞	∞	2, I	2, J	∞	1, A	1, A	∞	2, J
B	0,$\emptyset$	2, J	∞	3, B	∞	2, I	2, J	3, B	1, A	1, A	∞	2, J
F	0,$\emptyset$	2, J	∞	3, B	∞	2, I	2, J	3, B	1, A	1, A	∞	2, J
L	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
G	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
D	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
C	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
E	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
H	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J
K	0,$\emptyset$	2, J	3, L	3, B	3, L	2, I	2, J	3, B	1, A	1, A	3, L	2, J

In this case, after vertex L becomes fixed, no other distance label decreases (so the table doesn't change past this point). Of course, for large graphs, it may take many steps until we're done, and we cannot necessarily see that we're finished until the very end (indeed, all but the last vertex in the table could join up to some new vertex in the graph).

The subgraph that consists of edges in the shortest paths (specified by the predecessor array) is shown below.



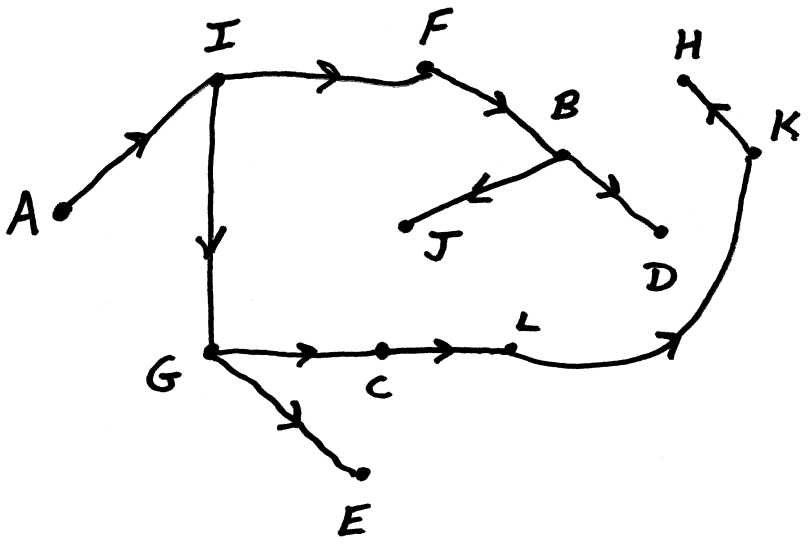
The next graph follows.



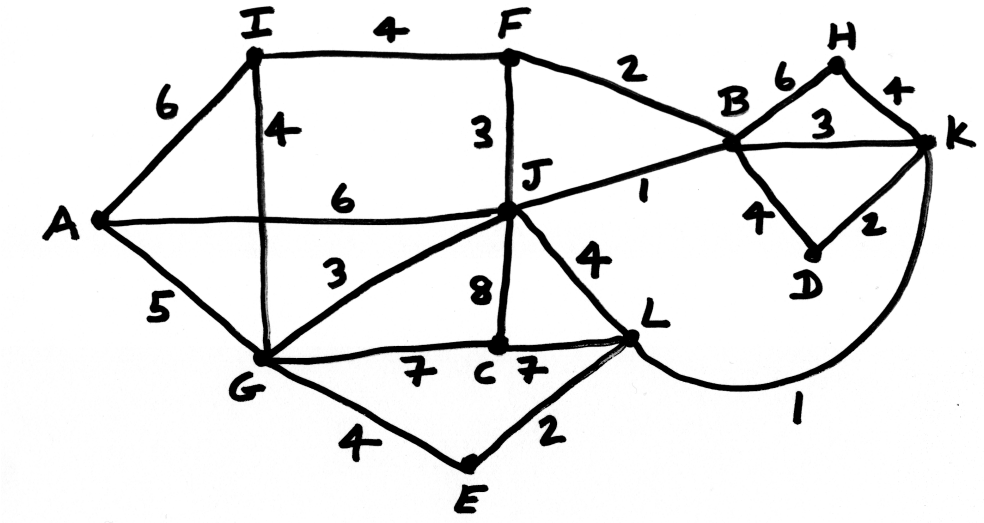
For this directed graph, we end up with this table from the execution of Dijsktra's algorithm. As above, the exact order when the vertex labels become fixed might differ slightly from mine (in the case of ties). You should end up with the same smallest distances, but the predecessor array might differ slightly (again, because of how you might have broken ties).

fixed	A	B	C	D	E	F	G	H	I	J	K	L
	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
A	0,$\emptyset$	∞	∞	∞	∞	∞	∞	∞	1, A	∞	∞	∞
I	0,$\emptyset$	∞	∞	∞	∞	2, I	2, I	∞	1, A	∞	∞	∞
G	0,$\emptyset$	∞	3, G	∞	3, G	2, I	2, I	∞	1, A	∞	∞	∞
F	0,$\emptyset$	3, F	3, G	∞	3, G	2, I	2, I	∞	1, A	∞	∞	∞
C	0,$\emptyset$	3, F	3, G	∞	3, G	2, I	2, I	∞	1, A	∞	∞	4, C
E	0,$\emptyset$	3, F	3, G	∞	3, G	2, I	2, I	∞	1, A	∞	∞	4, C
B	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	∞	1, A	4, B	∞	4, C
D	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	∞	1, A	4, B	∞	4, C
L	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	∞	1, A	4, B	5, L	4, C
J	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	∞	1, A	4, B	5, L	4, C
K	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	6, K	1, A	4, B	5, L	4, C
H	0,$\emptyset$	3, F	3, G	4, B	3, G	2, I	2, I	6, K	1, A	4, B	5, L	4, C

The subgraph consisting of edges in the shortest paths is shown below.



2. Consider the weighted (undirected) graph below. (Edges $\{G, I\}$ and $\{A, J\}$ cross but there is not a vertex at their intersection.)



Find shortest paths from vertex A to all other vertices in the graph. You should find *both* the shortest distances *and* the predecessor array which will allow us to reconstruct a path joining A to any vertex.

As above, *draw* the subgraph that contains only those edges used in the shortest paths.

In the table below, I keep track of the current minimum distances, as well as the predecessor on the path to that vertex. Note that the predecessor

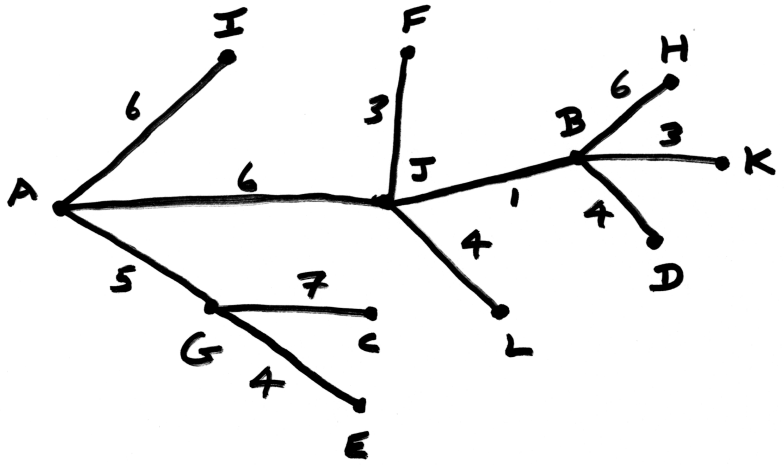
of A is listed as $\emptyset$ since the vertex A has no predecessor. For lack of room, I don't list all of the vertices that are currently "fixed" in the first column, just the *new* vertex that becomes fixed in that round.

Fixed	A	B	C	D	E	F	G	H	I	J	K	L
$\emptyset$	$0, \emptyset$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
A	$0, \emptyset$	∞	∞	∞	∞	∞	$5, A$	∞	$6, A$	$6, A$	∞	∞
G	$0, \emptyset$	∞	$12, G$	∞	$9, G$	∞	$5, A$	∞	$6, A$	$6, A$	∞	∞
I	$0, \emptyset$	∞	$12, G$	∞	$9, G$	$10, I$	$5, A$	∞	$6, A$	$6, A$	∞	∞
J	$0, \emptyset$	$7, J$	$12, G$	∞	$9, G$	$9, J$	$5, A$	∞	$6, A$	$6, A$	∞	$10, J$
B	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
E	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
F	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
K	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
L	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
D	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
C	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$
H	$0, \emptyset$	$7, J$	$12, G$	$11, B$	$9, G$	$9, J$	$5, A$	$13, B$	$6, A$	$6, A$	$10, B$	$10, J$

In this case, after the vertex B becomes fixed, it turns out that none of the distances decrease in the following steps. In general, it might take longer for other distances to decrease to their proper values. (And certainly if I had chosen other values for some of the edges, this could have easily been the case.)

The predecessors allow us to determine the path from vertex A to any other vertex by tracing backwards through the path. For example, to get to vertex K, we come through vertex B, which is reached via vertex J, and we get to J from A. Thus the path from A to K is A, B, J, K. In a similar manner we can determine any of the other shortest paths from vertex A to other vertices.

The subgraph that consists of all the "shortest path edges" is shown on the next page. (As an aside, note that the subgraph generated in this manner is necessarily a spanning tree of the graph, since each vertex (except the starting vertex) has a unique predecessor. This will not, in general be a minimum spanning tree of the original graph.)


I
F
H
6
3
6
B
3
K
A
6
J
1
4
4
D
5
7
C
G
L
4
E