EXERCISES for Section 4.2

In Exercises 4.2.1 through 4.2.7, draw the depth-first tree that results when Algorithm 4.2.1 is applied to the graph shown below, starting at the specified vertex. Include the dfnumbers and use: (a) lexicographic order as the default priority; (b) reverse lexicographic order as the default priority.

4.2.1[S] Vertex w.

4.2.2 Vertex u.

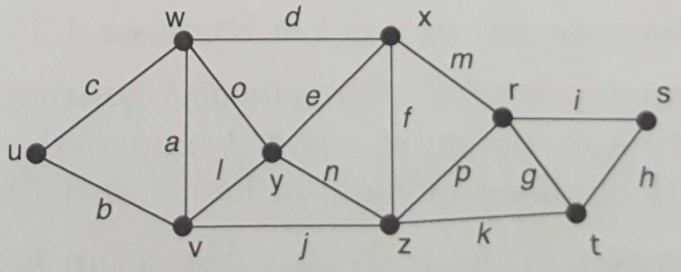
4.2.3 Vertex x.

4.2.4 Vertex y.

4.2.5[S] Vertex t.

4.2.6 Vertex z.

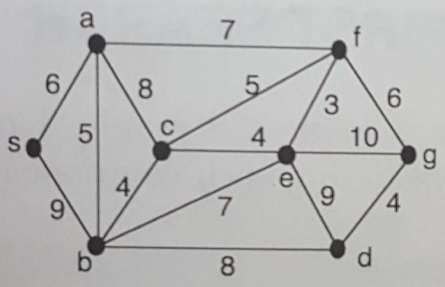
4.2.7 Vertex s.



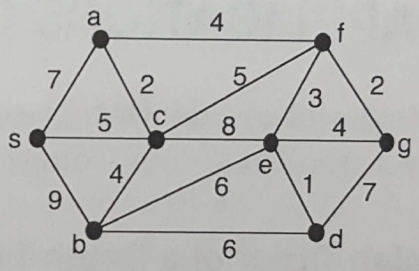
EXERCISES for Section 4.3

In Exercises 4.3.1 through 4.3.4, apply Prim's algorithm (Algorithm 4.3.1) to the given weighted graph, starting at vertex s and resolving ties as in Example 4.3.1. Draw the minimum spanning tree that results, and indicate its total weight. Also give the discovery number at each vertex.

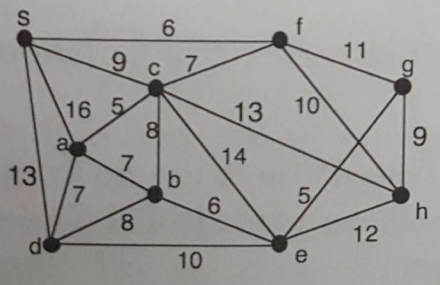
4.3.1[S]



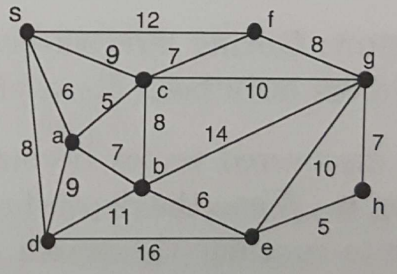
4.3.2



4.3.3



4.3.4



In Exercises 4.3.5 through 4.3.8, apply Dijkstra's algorithm (Algorithm 4.3.2) to the given weighted graph, starting at vertex s and resolving ties as in Example 4.3.3. Draw the shortest-path tree that results, and indicate for each vertex v, the discovery number and the distance from vertex s to v.

4.3.5[S] The graph of Exercise 4.3.1.

4.3.6 The graph of Exercise 4.3.2.

4.3.7 The graph of Exercise 4.3.3.

4.3.8 The graph of Exercise 4.3.4.

4.3.9 Suppose that the weighted graph shown below represents a communication network, where the weight p_{ij} on arc ij is the probability that the link from i to j does not fail. If the link failures are independent of one another, then the probability that a path does not fail is the product of the link probabilities for that path. Under this assumption, find the most reliable path from s to t. (Hint: Consider $-log_{10}p_{ij}$.)

EXERCISES for Section 4.4

In Exercises 4.4.1 through 4.4.7, (a) give the dfnumber and low values for each vertex that result from a depth-first search on the graph shown, starting at the specified vertex. Use the alphabetical order of the vertex names as the default priority; (b) verify the characterization given by Corollary 4.4.12 for the calculations of part (a).

4.4.1[S] Vertex *a*.

4.4.2 Vertex *b*.

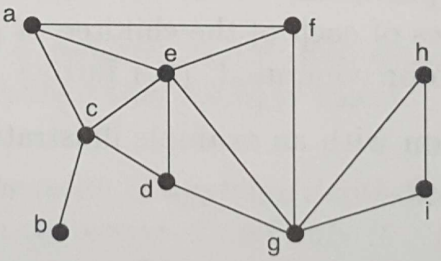
4.4.3 Vertex *c*.

4.4.4 Vertex *e*.

4.4.5[S] Vertex *g*.

4.4.6 Vertex *i*.

4.4.7 Vertex *f*.

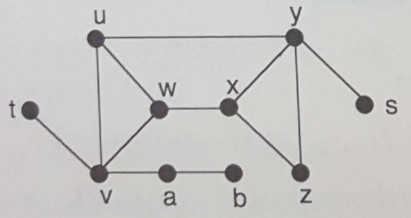


4.4.8 Prove that a postorder traversal of a depth-first tree reproduces the finish order of the depth-first search.

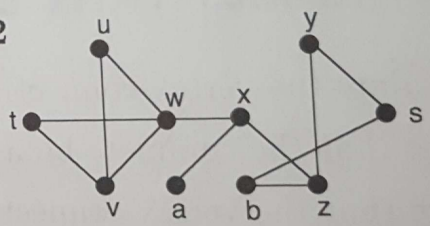
EXERCISES for Section 5.4

In Exercises 5.4.1 through 5.4.4, identify the blocks in the given graph and draw the block graph.

5.4.1[S]



5.4.2

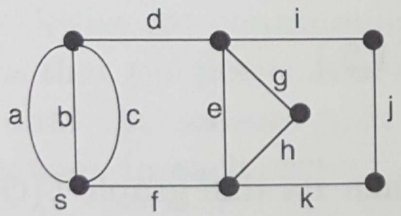


5.4.6 Find two non-isomorphic connected graphs with six vertices, six edges, and three blocks.

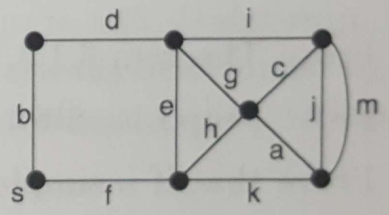
5.4.7 Find a graph whose block graph is the n-cycle graph C_n.

In Exercises 6.1.6 through 6.1.9, apply Algorithm 6.1.1 to construct an eulerian tour of the given graph. Begin the construction at vertex s.

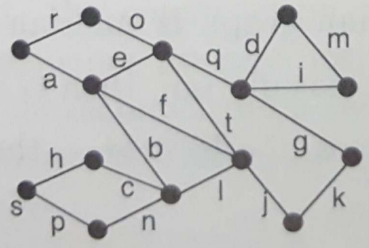
6.1.6



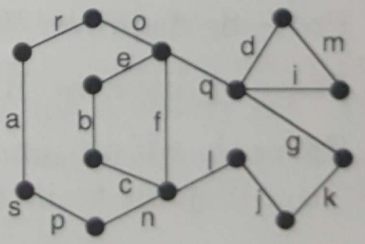
6.1.7[S]



6.1.8

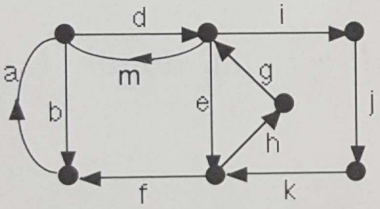


6.1.9

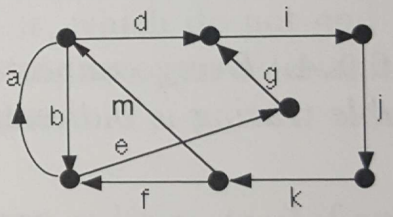


In Exercises 6.1.16 through 6.1.19, use an appropriate modification of Algorithm 6.1.1 to construct an eulerian tour or open eulerian trail in the given digraph.

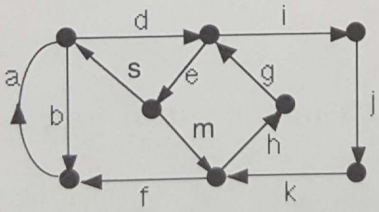
6.1.**16**



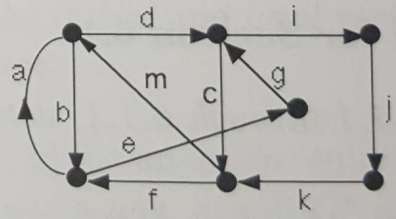
6.1.**17**



6.1.**18**$^{\mathbf{S}}$

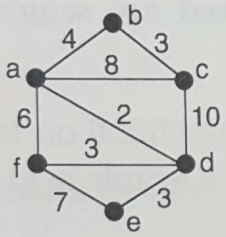


6.1.**19**

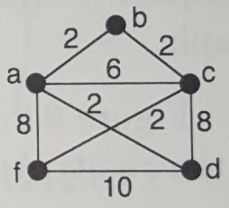


In Exercises 6.2.12 through 6.2.15, apply Algorithm 6.2.2 to find a minimum-weight postman tour for the given weighted graph. Determine whether the solution is unique.

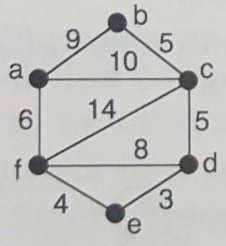
6.2.**12**



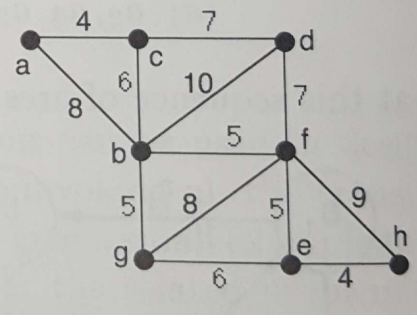
6.2.**13**



6.2.**14**^{S}



6.2.**15**



In Exercises 6.3.6 through 6.3.10, draw the specified graph or prove that it does not exist.

6.3.6[S] An 8-vertex simple graph with more than 8 edges that is both eulerian and hamiltonian.

6.3.7 An 8-vertex simple graph with more than 8 edges that is eulerian but not hamiltonian.

6.3.8 An 8-vertex simple graph with more than 8 edges that is hamiltonian but not eulerian.

6.3.9 An 8-vertex simple hamiltonian graph that does not satisfy the conditions of Ore's theorem.

6.3.10 A 6-vertex simple graph with 10 edges that is not hamiltonian.