4-2.

Not necessarily true. Counter example: The example graph: 

The minimum spanning tree:

The distance between b and c in the minimum spanning tree is 2+4=6; however, the shortest distance in the full graph is 5.

4-3. Not necessarily true. Counter example is the same as in 4-2.

4-4.

Bridge e can not be a back edge. Suppose e connects a and b. If e is a back edge from b to a, then there would exist another pathway connecting a and b, and thus contradiction to the fact that e is a bridge.

4-5.

a. All n-1 vertex are connected directly to the starting vertex v.
b. Vertices in the graph: 1, 2, 3, ..., n. Edges in the graph: (1, 2), (2, 3), ..., (n − 1, n).
   So that the n vertices are connected into a line. The starting point is 1.

c. The same graph as in b., but the starting point is n/2.

4-6. Given pre-order and in-order traversals of a binary tree, the tree can be reconstituted. Algorithm: The first element in the pre-order array is the root of the tree. In the in-order array, elements left to the root element belongs to the left subtree of the root, and elements right to the root element belongs to the right subtree of the root. Thus, we know the elements in both subtrees and hence get the pre-order traversals of both subtrees. So that we can recursively perform the above operation and reconstruct the tree.

Given pre-order and post-order traversals, the tree can not be constructed. Counter example: Consider the following two trees:

pre-order: a, b, c
post-order: c, b, a

The two trees are different, but have the same pre and post order traversals.

4-7. Perform a post order traversal of the tree to find the postfix expression. This takes $O(n)$ for $n$ nodes in the tree.

Scan the postfix expression from left to right and using a stack do the following:

If input is an operand then push it into the stack. If input is an operator, then pop the required number of operands from the stack, perform the operation and push the result on the stack. The final content of the stack is the result. One scan of the postfix expression accomplishes this. Hence it is an $O(n)$ algorithm.
The two steps may be merged to recursively evaluate the tree. The non-recursive stack
algorithm is discussed above to understand the exact evaluation process. When a leaf
is visited, it’s value is copied to it’s parent’s node. When an internal or operator node
is visited and the values for all its children are available, it is evaluated and the value
copied to its parent node. A tree has \( n \) nodes and \( n - 1 \) edges, hence the number of
evaluations is \( n \) and number of copies made is \( n - 1 \).

- **4-8.** The only difference with the previous problem is that here a node may have
  multiple parents. Each time a node is evaluated, copy the value produced in all its
  parent nodes. Since there are \( n \) nodes and \( m \) edges, there are \( m \) copies to be made and
  \( n \) evaluations performed. Thus the complexity is \( O(m + n) \).

- **4-9.**
  1. Divide the nodes into two groups
     - group 1 : contain nodes with degree \( \geq k \)
     - group 2 : contain nodes with degree \( < k \).
  2. Select one node from the group 2, and remove it and its connected edges
  3. After removing, if there is a node that has degree \( < k \) in group 1, moves it into
     group 2
  4. repeat above steps until all the nodes in group 2 are removed
  5. Maximum induced subgraph can be constructed from group 1.

- **4-10.** After doing DFS over the graph, the leaf nodes are not articulation point. If
  there is a back edge, the intermediate nodes of the cycle are not articulation points,
  where the intermediate nodes have no other branches.

- **4-11.**
  1. Do DFS
  2. remove leaf nodes
  3. After removing leaf node, if its parent node is also leaf nodes, remove its parent
     as well.

- **4.** Do a depth-first traversal of the tree. We will compute the largest diameter through
each node in the DFS tree. Note that the diameter-path may either go through the
node to its parent, in which case we add one to the length of this distance as we work
up the tree. Otherwise, it is the sum of the length of two paths which end at this node
through two different branches.

- **5.** Find a cycle in the graph, delete it (leaving vertices as even degree) and then splice
  this cycle in with another other cycle sharing a vertex (which must exist if the graph
  is connected)

- **6.**
1. Divide the nodes into two groups 
   - group 1 : contain nodes with degree = 2 
   - group 2 : contain nodes with degree != 2. 
2. Select one node from the group 1, and remove it by replacing edges (u,w) and 
   (v,w) by a new edge (u,w). If there is a pre-existing edge (u,w)', do not add the 
   new edge. 
3. Update 2 groups. 
4. repeat above steps 

• 7. Do this through dynamic-programming – suppose you knew the length of the shortest 
   path between each pair of vertices with k-1 edges; use that to determine it for k edges 
   by trying all edges in the graph.