Firstly, given a proof of Exercise 43, which will be used in the latter exercises.

43. Let G be a connected graph. Show that if T is a spanning tree of G constructed using depth-first search, then an edge of G not in T must be a back edge, that is, it must connect a vertex to one of its ancestors or one of its descendants in T.

Proof(mostly based on the sample answer): If an edge uv is not followed while we are processing vertex u during the depth-first search process, then it must be the case that the vertex v had already been visited. There are two cases. If vertex v was visited after we started processing u, then, because we are not finished processing u yet, v must appear in the subtree rooted at u (and hence, must be a descendant of u). On the other hand, if the processing of v had already begun before we started processing u, then it must be that we are still forming the subtree rooted at v, so u is a descendant of v, and hence, v is an ancestor of u.

3. If there is a simple path of length more than k, then there must be a simple path of length k, which contradicts the assumption. So the most length of a simple path in G must be k-1. So the most length of a simple path in a spanning tree of G must be k-1. Considering a spanning tree of G constructed using depth-first search. Then using the conclusion in Exercise 43, every edge in G not in the spanning tree must connect a vertex to one of its ancestors or one of its descendants in T. And it's obvious that every edge which satisfies that it's both in G and in T must connect a vertex to one of its children or the parent. And because G is a simple graph, we can construct an injection from every edge in G to a pair of vertices in T which satisfies u is a descendant of v or v is a descendant of u (assume the pair of vertices are u and v). We call this kind of pairs of vertices as "good" pairs of vertices. Hence the number of the edges in G is less than or equal to the number of these pairs of vertices in T. In fact, the number of these pairs of vertices in T is at most (k-1)(n-1), for we can give a counting way in which the total number is at most (k-1)(n-1). For every vertex but the root, note the vertex as v, considering all vertices on the only simple path from the root to v but v itself, all these vertices can form a "good" pair of vertices with v. And in this counting way, we won't count a "good" pair twice or more and won't overlook any pair. And because of the assumption that the most length of a simple path in a spanning tree of G must be k-1, the number of all vertices on the only simple path from the root to v but v itself for any v is under k-1. So the total number of edges in G is at most(k-1)(n-1).

4. If the edge from v to w in G is in T, then either v is w's ancestor or w is v's ancestor. If the edge from v to w in G isn't in T, then because of what we have proved in Exercise 43, it must connect a vertex to one of its ancestors or one of its descendants in T, which means either v is w's ancestor or v is w's descendant. So either v is w's ancestor or v is w's descendant. If v is w's ancestor, because u is v's ancestor, u is w's ancestor, i.e. w is u's descendant in T(ii). If w is v's ancestor, because u is v's ancestor and what we proved in 1., either (i) w is u's ancestor in T, (ii) w is u's descendent in T, or (iii) w = u.

5. Write the simple path from the root to v in T as $r=v_0, v_1, \ldots, v_i, \ldots, v_n=v$. If u is v's ancestor in T, then there exists i such that $v_i=u$. If v is visited, then the edge $v_{n-1}v_n$ must

be followed, for if not, there must be another edge from v_j to v is followed, and because every vertex in T can be a forward step's terminal at most 1 time, so the edge $v_{n-1}v_n$ will not in T. So v_{n-1} must be visited. Similarly, we can infer v_{n-2}, v_{n-3}, …, v_0 must be visited. So $u=v_i$ must be visited. So the first time visiting u happens before the first time visiting v in the DFS process.

6. Assume w is visited after v, so there exists a moment when v is visited but w is not. If there is a simple path in G from v to w which passes through neither u nor u's ancestors, then we can infer all vertices in this path won't be visited before u is visited, otherwise if v' in this path is visited before u, then v won't be visited through the forward step u to v, but will be visited through a path from v' to v through this path. Write the simple path from v to w in T as $v=v_0, v_1, …, v_i, …, v_n=w$. So in the process of v, v_1 must be visited, otherwise the process will never end for when it goes back to v_0 then there is an edge v_0v_1 is augmentable. Similarly, v_2, …, $v_n=w$ must be visited. But every vertex in T can be a forward step's terminal at most 1 time, so the edge uw will not in T, which contradicts the assumption. So any simple path in G from v to w either passes through u or passes through at least one u's ancestor in the tree generated by this DFS process.