Two algorithms to find a spanning tree

Algorithm 1 (Cutting-down algorithm).

Input. An arbitrary connected graph G.

Output. A spanning tree T of G.

Algorithm.

(0) Delete any multiple edges and loops from G; let H be the resulting subgraph of G. $\{$Then H is connected and includes all vertices of G.$\}$

(1) While H still includes any cycle as a subgraph, repeat:

(1.1) Delete *one* edge of some cycle of H; call the resulting graph H again. $\{$Then H is connected and includes all vertices of G.$\}$

(2) Let $T = H$. $\{$Then T is a spanning tree of G.$\}$

Algorithm 2 (Depth-first search algorithm).

Input. An arbitrary connected graph G with exactly n vertices.

Output. A spanning tree T_n of G.

Algorithm.

(0) Delete any multiple edges and loops from G; call the resulting graph G again. $\{$Then G is connected and includes all vertices of the original G.$\}$

(1) Set $k = 1$; choose a vertex v_1 of G; and let

$$V_1 = \{v_1\}, \quad E_1 = \emptyset, \quad T_1 = (V_1, E_1).$$

$\{$Then T_1 is a subgraph of G with 1 vertex that's a tree.$\}$

(2) While $k < n$, do: $\{$You already have

$$V_k = \{v_1, v_2, \ldots, v_k\}, \quad E_k = \{e_1, e_2, \ldots, e_{k-1}\}, \quad T_k = (V_k, E_k)$$

with T_k a tree with k vertices that's a subgraph of G.$\}$

(2.1) Choose that vertex $u \in V_k$ having the *largest* subscript such that there is some edge e with ends u and some x that is not already pat of any cycle consisting of edges from E_k. $\{$Then $x \notin V_k$ and $e \notin E_k$.$\}$

(2.2) Let

$$v_{k+1} = x, \qquad e_k = e,$$

and then let

$$V_{k+1} = \{v_1, v_2, \ldots, v_k, v_{k+1}\}, E_{k+1} = \{e_1, e_2, \ldots, e_{k-1}, e_k\}, T_{k+1} = (V_{k+1}, E_{k+1}).$$

$\{$Then T_{k+1} is a tree with $k+1$ vertices that's a subgraph of G.$\}$

(2.3) Increment k by 1.