Let $G = (V, E)$ be a connected graph with $|V| = n > 1$, possibly with a weight function $w$.

A **spanning tree** of $G$ is a subgraph that is a tree and that includes every vertex of $G$.

A **minimum spanning tree** of a weighted graph is a spanning tree of least weight.

Two spanning trees $T_1$ and $T_2$ are **different** if they use different edges of $G$.

Suppose the vertices of $G$ are labeled $v_1, v_2, \ldots, v_n$. The **adjacency matrix** of $G$ is the $n \times n$ matrix $A = (a_{i,j})$ with entries $a_{i,j} = 1$ if $v_i v_j$ is an edge and 0 otherwise.

**Theorem 1 (Kirchhoff)** Let $A$ be the adjacency matrix of $G$ and $M$ the matrix obtained from $A$ by changing all 1’s to −1’s and $a_{i,i} = 0$ to $a_{i,i} = \deg(v_i)$. Then the number of spanning trees of $G$ is the value of any cofactor of $M$.

**Kruskal’s Minimum Spanning Tree (MST) Algorithm**

**Step 1:** Find a minimum weight edge $e_1 \in E$. Set $k = 1$ and $S_k = \{e_1\}$.

**Step 2:** While $k < n$,

- **Step 2a:** Look for an edge $e \in E$ such that $\{e\} \cup S_k$ does not contain a circuit.
- **Step 2b:** If no such $e$ exists, output $S_k = \{e_1, \ldots, e_k\}$ and stop.
- **Step 2c:** Else, let $S_{k+1} = \{e_{k+1}\} \cup S_k$ where the edge $e_{k+1}$ has least weight among the possible $e$ for this iteration.

**Prim’s Minimum Spanning Tree (MST) Algorithm**

**Step 1:** Choose any vertex $v$ and let $e_1$ be an edge of least weight incident with $v$. Set $k = 1$ and $S_k = \{e_1\}$.

**Step 2:** While $k < n$,

- **Step 2a:** Look for a vertex $w$ that is not in the subgraph $T$ whose edges are $S_k$.
- **Step 2b:** If no such $w$ exists, output $S_k = \{e_1, \ldots, e_k\}$ and stop.
- **Step 2c:** Else, let $S_{k+1} = \{e_{k+1}\} \cup S_k$ where the edge $e_{k+1}$ has least weight among all edges of the form $ux$, where $u$ is a vertex in $T$ and $x$ is not.