An Advice on Problem Solving

If possible, try **not to** use arguments of the following type in proofs:

- The Best case is ....
- The worst case is ....
- The slowest running time for my algorithm is ....

These arguments need **rigorous** justification, and they are usually the main reason that your proofs can become wrong, or unjustified.
Minimum Spanning Tree (MST)

Given a connected graph $G = (V, E)$ with real-valued edge weights $c_e$, an MST is a subset of the edges $T \subseteq E$ such that $T$ is a spanning tree whose sum of edge weights is minimized.

$G = (V, E)$

$c(T) = \sum_{e \in T} c_e = 50$
Cuts

In a graph $G = (V, E)$ a cut is a bipartition of $V$ into sets $S, V - S$ for some $S \subseteq V$. We show it by $(S, V - S)$.

An edge $e = \{u, v\}$ is in the cut $(S, V - S)$ if exactly one of $u, v$ is in $S$.

**OBS**: If $G$ is connected $\Rightarrow$ there is at least one edge in every cut.
Cycles and Cuts

Claim. A cycle crosses a cut (from S to V-S) an even number of times.

Pf. (by picture)
Properties of the OPT

Simplifying assumption: All edge costs $c_e$ are distinct.

Cut property: Let $S$ be any subset of nodes (called a cut), and let $e$ be the min cost edge with exactly one endpoint in $S$. Then every MST contains $e$.

Cycle property. Let $C$ be any cycle, and let $f$ be the max cost edge belonging to $C$. Then no MST contains $f$.

red edge is in the MST

Green edge is not in the MST
Cut Property: Proof

Simplifying assumption: All edge costs $c_e$ are distinct.

Cut property. Let $S$ be any subset of nodes, and let $e$ be the min cost edge with exactly one endpoint in $S$. Then the $T^*$ contains $e$.

Pf. By contradiction

Suppose $e = \{u,v\}$ does not belong to $T^*$.

Adding $e$ to $T^*$ creates a cycle $C$ in $T^*$. $C$ crosses $S$ even number of times $\Rightarrow$ there exists another edge, say $f$, that leaves $S$.

$$T = T^* \cup \{e\} - \{f\}$$ is also a spanning tree.

Since $c_e < c_f$, $c(T) < c(T^*)$.

This is a contradiction.
Cycle Property: Proof

Simplifying assumption: All edge costs $c_e$ are distinct.

Cycle property: Let $C$ be any cycle in $G$, and let $f$ be the max cost edge belonging to $C$. Then the MST $T^*$ does not contain $f$.

Pf. (By contradiction)
Suppose $f$ belongs to $T^*$.
Deleting $f$ from $T^*$ cuts $T^*$ into two connected components. There exists another edge, say $e$, that is in the cycle and connects the components.

$$T = T^* \cup \{e\} - \{f\}$$ is also a spanning tree.
Since $c_e < c_f$, $c(T) < c(T^*)$.
This is a contradiction.
Kruskal's Algorithm [1956]

Kruskal(G, c) {
    Sort edges weights so that \( c_1 \leq c_2 \leq \ldots \leq c_m \).
    \( T \leftarrow \emptyset \)

    foreach \((u \in V)\) make a set containing singleton \{u\}

    for i = 1 to m
        Let \((u,v) = e_i\)
        if (u and v are in different sets) {
            \( T \leftarrow T \cup \{e_i\} \)
            merge the sets containing u and v
        }
    return T
}
Kruskal’s Algorithm: Pf of Correctness

Consider edges in ascending order of weight.

Case 1: If adding e to T creates a cycle, discard e according to cycle property.

Case 2: Otherwise, insert e = (u, v) into T according to cut property where S = set of nodes in u's connected component.
Implementation: Kruskal’s Algorithm

Implementation. Use the union-find data structure.

- Build set $T$ of edges in the MST.
- Maintain a set for each connected component.
- $O(m \log n)$ for sorting and $O(m \log n)$ for union-find

```plaintext
Kruskal(G, c) {
    Sort edges weights so that $c_1 \leq c_2 \leq \ldots \leq c_m$.
    $T \leftarrow \emptyset$

    foreach $(u \in V)$ make a set containing singleton \{
    $u$\}

    for $i = 1$ to $m$
        Let $(u, v) = e_i$
        if (u and v are in different sets) \{\[
            T \leftarrow T \cup \{e_i\}
            merge the sets containing $u$ and $v$
        \} \text{ } O(m \log n)

    return \{ \[
        T \leftarrow \emptyset \text{ } O(1)
    \}
}
```