Proof Technique Bag O' Tricks

1. Structural Arguments
   (a) Arguments that consider the way in which a graph or subgraph must be configured in terms of the "structure" of vertices and edges
   (b) Consider v of degree x that is configured in some way
   (c) Consider some v and $G' = G - v$
2. Extremal Arguments
   (a) Extremal Principle: within a well-ordered set, there is some maximum/minimum value within that set
   (b) Consider maximum path P
   (c) Consider v of maximum degree in G
3. Parity Arguments
   (a) We can often use parity on the countable properties of graphs
   (b) even + even = even; odd + odd = even; even + odd = odd
4. Weak Induction
   (a) $P(1), \ldots, P(k), P(k+1)$
   (b) Demonstrate our basis $P(1)$ – and/or $P(0)$ and/or $P(2)$, etc.
   (c) Assume what we're trying to prove for our $P(k)$ case via inductive hypothesis
   (d) Construct our $P(k+1)$ case
   (e) Show that what we're trying to prove still holds on $P(k+1)$
5. Strong Induction
   (a) $P(1), \ldots, P(k), \ldots, P(n)$
   (b) Demonstrate our basis
   (c) Consider our $P(n)$ case, where original assumptions hold
   (d) Construct our $P(k)$ case by removing some part of $P(n)$ – $P(k)$ construction must still fit our original assumptions of $P(n)$
   (e) Assume what we're trying to prove for our $P(k)$ case via inductive hypothesis
   (f) Show that what we're trying to prove still holds on $P(n)$
6. Construction Methods for Strong Induction
   (a) There are many ways we can get from $P(n)$ to $P(k)$ an a strong inductive proof

   (b) *Edge Deletion*: $P(k) = P(n) - e : e \in E(P(n))$
   (c) *Vertex Deletion*: $P(k) = P(n) - v : v \in V(P(n))$
   (d) *Edge Contraction*: $P(k) = P(n) \cdot e : e = (u, v) \in E(P(n))$
   (e) *Subgraph Deletion*: $P(k) = P(n) - S : S \subseteq P(n)$

7. Necessity and Sufficiency
   (a) To prove an equivalence, prove necessity and sufficiency
   (b) To show: A is equivalent to B
   (c) First show: A implies B
   (d) Then show: B implies A

8. Contrapositive
   (a) "A implies B" is equivalent to saying "not B implies not A"
   (b) "A is equivalent to B" is equivalent to saying "not A is equivalent to not B"

9. Proof by Algorithm
   (a) Construct an algorithm to demonstrate a property holds
   (b) Here's an algorithm that shows any graph with property A can be be processed in a way that definitively shows it has property B

10. Proof by Counter-Example
   (a) Demonstrating some property doesn't hold via an explicit construction
   (b) Here's a counter-example that shows how A does not imply B

11. Consider the Cases
   (a) For many of the above techniques, we may also need to consider multiple possibilities as part of our proof
   (b) E.g., consider connected graph G, vertex $v \in V(G)$, and $G - v$
   (c) Case 1: $G - v$ is still connected
   (d) Case 2: $G - v$ has exactly two components
   (e) Case 3: $G - v$ has three or more components