The minimum spanning tree problem is defined on undirected, weighted graphs as follows: a spanning tree of that graph is a subgraph that is a tree and connects all the vertices together. The minimum spanning tree is a spanning tree of minimum weight. For example, given the following graph:

Its minimum spanning tree is:

(a) Show that the minimum spanning tree has an optimal substructure by showing that given an MST for a graph $G = (V, E)$, any sub-tree of the MST is a minimum spanning tree with respect to the subset of vertices and their edges in $G$.

(b) A cut in a graph $G = (V, E)$ is a division of the vertices into two disjoint non-empty subsets, $S$ and $T$. An edge that crosses the cut is an edge that has one vertex in $S$ and one vertex in $T$. An example here shows a cut in the graph above, where $S = \{v_1, v_3, v_4\}$ and $T = \{v_2, v_5\}$. Edges crossing the cut are shown in bold. The greedy choice property says that for any cut in the graph, the lightest edge crossing the cut is a part of a minimum spanning tree. Prove it is true for any cut (not only this example!). You can show by a simple exchange argument.
(c) Show that the lightest edge in the graph is always a part of an MST (there may be more than one MST if the weights are not unique).

2. Given a tree \( T = (V, E) \) (not necessarily binary) an independent set is subset of nodes in the tree such that no two nodes are adjacent (in a tree it means, no parent and child can be in the same independent set). The problem of finding the maximum independent set in a tree involves finding an independent set of maximum size (obviously).

(a) Show that the problem has an optimal substructure by showing that given a maximum independent set on a tree, every subset of it represents a maximum independent set with respect to its subtree.

(b) Every tree has at least one leaf. Show that any leaf node \( v \) in a tree must be a part of a maximum size independent set. **Hint:** Assume we have a maximum independent set \( S \) on the tree. It either contains \( v \) of not, take it from there.

(c) Give a linear time algorithm to obtain a maximum size independent set in a tree.

3. Prove that a binary tree that is not full (i.e., that at least one of its nodes has only one child) cannot correspond to an optimal prefix code.

4. Suppose that a data file contains a sequence of 8-bit characters such that all 256 characters are about equally common: the maximum character frequency is less than twice the minimum character frequency. Prove that Huffman coding in this case is no more efficient than using an ordinary 8-bit fixed-length code.

5. The proof of Lemma 1.3 in the Lecture 12 handout is a proof by induction, but is presented a bit informally. What is the inductive hypothesis?

6. Show the \( d \) and \( \pi \) values that result from running breadth-first search on the undirected graph in the figure, using vertex \( u \) as the source.

7. Give an example of a directed graph \( G = (V, E) \), a source vertex \( s \in V \), and a set of tree edges \( E_\pi \in E \) such that for each vertex \( v \in V \), the unique simple path in the graph \( (V, E_\pi) \) from \( s \) to \( v \) is a shortest path in \( G \), yet the set of edges \( E_\pi \) cannot be produced by running BFS on \( G \), no matter how the vertices are ordered in each adjacency list.
8. Show by induction that the number of degree-2 nodes in any nonempty binary tree is 1 fewer than the number of leaves. Conclude that the number of internal nodes in a full binary tree is 1 fewer than the number of leaves. Be sure to carefully state the inductive hypothesis. It will help you in constructing the proof.