1) (15 pts) A tree is a undirected unweighted graph that is connected and does not have any cycles. Let \( L_1 = \{ G \mid G \text{ is a undirected unweighted graph that is a tree} \} \). Prove that \( L_1 \in P \).

Here is an algorithm that runs in polynomial time that decides membership in \( L_1 \):

1. Count the number of vertices in \( G \). Let this be \( n \).
2. Count the number of edges in \( G \). If this number isn’t \( n-1 \), reject.
3. Run a DFS from an arbitrarily chosen vertex of \( G \). If all vertices are marked as visited at the end of the DFS, accept, else reject.

A tree of \( n \) vertices always has \( n-1 \) edges and is connected. (This is an alternate definition of a tree.) In this algorithm, we simply check to see if \( G \) satisfies these requirements. All three steps run in \( O(V+E) \) time, where \( V \) equals the number of vertices in the graph and \( E \) equals the number of edges in the graph. This is polynomial in the size of the input, as desired.

**Grading:** Algorithm – 12 pts (4 pts for verifying connectivity, 8 pts for verifying no cycles)
Poly Time Justification – 3 pts

2) (15 pts) In the last class, we showed that 3-SAT \( \leq_p \) Vertex-Cover via a polynomial-time reduction. Show the output of executing that reduction on the Boolean formula in 3-CNF shown below:

\[(a \lor b \lor c) \land (\overline{a} \lor \overline{b} \lor c) \land (\overline{a} \lor \overline{b} \lor \overline{c}) \land (\overline{a} \lor \overline{b} \lor \overline{c})\]

Grading:
- triangles = 5 pts
- bars = 3 pts
- cross edges = 4 pts
- \( k = 3 \) pts
3) (20 pts) Briefly explain why in the proof of the Cook-Levin Theorem, a $2 \times 3$ window was used to verify the consistency of the tableau. In particular, explain why 1 row is not enough, why 3 rows is unnecessary, why 2 columns is not enough, and why 4 columns is unnecessary.

First, consider the number of rows. If the window only had 1 row, we could only verify if a particular configuration of the Turing Machine was a valid Turing Machine configuration in general. We could not verify anything about a transition, because the information for any transition necessarily lies on two different rows, identifying the configuration at the current time step and the subsequent time step. Three rows is unnecessary because no transition effects three consecutive transitions and whatever can be verified from a window on row $i$, $i+1$ and $i+2$ can also be verified by two windows, one on rows $i$ and $i+1$ and another on rows $i+1$ and $i+2$.

Next, let’s look at the number of columns. 2 columns is too few because a left transition potentially effects the placement of three consecutive items in a single configuration: the item it read/wrote, the item to the left of it, and the state to which it moves. If we had a window with only two columns and we looked at $[q, a]$, for example, and on the following transition we saw $[b, a]$, we have no way of verifying that new state of the TM followed the proper transition of being in $q$ and reading an $a$. Four columns is not necessary because for all left and right transitions, no more than three columns are affected. Any window with four columns will necessarily have one of its columns remaining unchanged. Knowing the state of this column is unnecessary to judge whether or not a transition was valid.

Grading: There are four items to justify. Each is worth 5 points.