Problem 7.34

Show that 3-coloring is NP-complete.

- To show this, we must show two things
  1. $3\text{COLOR} \in \text{NP}$. This is easily seen by viewing the coloring of the graph as the certificate. Given a list of colorings for nodes, it is easy to verify in polynomial time that no two adjacent nodes have the same color.
  2. For any other language $L \in \text{NP}$, there is a polynomial reduction that maps strings in the language to 3-colorable graphs. This is shown by a reduction from SAT. Since we already know that SAT is NP-complete, we only need to show that there is a polynomial time algorithm that maps satisfiable formulas to 3-colorable graphs, and non-satisfiable formulas to non-3-colorable graphs. Then we can compose that algorithm with the reduction to SAT to make a polynomial reduction of $L$ to $3\text{COLOR}$. See attached construction.

Problem 8.11

Show that the language of properly nested parenthesis is in L.

- Consider the following algorithm:
  B = “On input $w$, where $w$ is a sequence of parenthesis.
  1. Starting at the first character of $w$, move right across $w$
  2. Every time you encounter a $(, add 1 to the work-tape and move right.
  3. Whenever you encounter a $)$ and the work tape is blank, reject. Otherwise subtract 1 from the work-tape and move right.
  4. When you’ve reached the end, accept if the work tape is blank, reject if it’s not.”
• Notice that the only space this algorithm uses is for the counter on the work tape. If this counter is in binary, then the most space it will ever use is $O(\ln k)$ where $k$ is the number of $s$. Since the number of $s$ is of course less than or equal to $n$, the size of the tape, this places the language in $L$. 