This week, we’ve started to understand what makes some problems seemingly hard to compute. In this lab, we’ll consider the problem of verifying that an algorithm’s answer is correct. Recall that a decision problem is a problem that requires a yes or no answer. Alternatively, we can describe decision problem as a set $L \subseteq \{0, 1\}^*$; think of $L$ as the set of all yes inputs i.e., the set of inputs $x$ such that one should output yes on input $x$. Let $|x|$ denote the length of $x$, in bits.

**Polynomial-time Verifiers.** Call $V$ an efficient verifier for a decision problem $L$ if

1. $V$ is a polynomial-time algorithm that takes two inputs $x$ and $w$.
2. There is a polynomial function $p$ such that for all strings $x$, $x \in L$ if and only if there exists $w$ such that $|w| \leq p(|x|)$ and $V(x, w) = \text{yes}$.

The string $w$ is usually called the witness or certificate. Think of $w$ as some proof that $x \in L$. For $V$ to be a polynomial-time verifier, $w$ must have size some polynomial of the input $x$. For example, if $x$ represents a graph with $n$ vertices and $m$ edges, the length of $w$ could be $n^2$ or $m^3$ or $(n+m)^{100}$ but not $2^n$.

Consider this lab a success if you complete problem 2 and make progress on problems 3,4.

1. **Verifier Debugging.** Consider the **Three-Coloring** problem: Given $G = (V, E)$ return yes iff the vertices in $G$ can be colored using at most three colors such that each edge $(u, v) \in E$ is bichromatic.

Consider the following verifier for Three-Coloring. The witness we request is a valid three coloring of the undirected graph $G = (V, E)$, which is specified as a list of two-digit binary strings $w = w_1w_2\ldots w_k$ where we interpret

$$w_i = \begin{cases} 
00, \text{ vertex } i \text{ is colored BLUE} \\
01, \text{ vertex } i \text{ is colored GREEN} \\
10, \text{ vertex } i \text{ is colored RED}
\end{cases}$$

**threeColoringVerifier**($G = (V, E), w$)
1. for each $w_i$ in $w$
2. if $w_i = 11$
3. return NO
4. for $j$ from $i + 1$ to $|w|$
5. if $w_i = w_j$ and $(i, j) \in E$
6. return NO
7. return YES

This verifier is not quite right.

Give an example witness $w$ and graph $G$ which is not three-colorable, such that

**threeColoringVerifier**($G, w) = \text{yes}$

1
2. Rewrite \texttt{threeColoringVerifier} so that it is a valid verifier for \textsc{Three-Coloring}. Prove that it is correct.

3. Show $\text{Sat} \leq_P \text{3-Sat}$.

4. You will eventually show that \textsc{Three-Coloring} is \textsc{NP-Complete}. Before getting there, it will be helpful to create some interesting three-colorable graphs. In all of the following exercises, you are to create a three-colorable graph (say the colors are red, blue, green) with certain special properties. The graphs you create should include three vertices marked $a, b, c$ but can (and often will) include other vertices. Except for the properties specified, these vertices should be \textit{unconstrained}. For example, unless the problem states that e.g. $a$ cannot be red, it must be possible to color the graph in such a way that $a$ is red. (You may fix colors for other vertices, just not $a, b, c$, and not in a way that constrains the colors of $a, b, c$.)

(a) Create a graph such that $a, b, c$ all have different colors.
(b) Create a graph such that $a, b, c$ all have the same color.
(c) Create a graph such that $a, b, c$ do \textit{NOT} all have the same color.
(d) Create a graph such that none of $a, b, c$ can be green.
(e) Create a graph such that none of $a, b, c$ are green, and they cannot \textit{all} be blue.