Problem 1

Let Fruits be the language of string representations of fruits. This is a finite set consisting of words such as “apple”, “banana”, “cherry”, and “durian”. Let Primes be the language of binary representations of prime numbers (which is an infinite set).

Consider the following languages, and prove whether they are decidable, undecidable but Turing-recognizable, or Turing-unrecognizable.

1. SuperFruit = \{⟨M⟩ : M is a TM and L(M) ⊂ Fruits\} (6 points)

2. Optimus’ = \{⟨M⟩ : G is a TM and L(M) = Primes\} (6 points)

Note: ⊂ indicates a strict subset.

Problem 2

The Bega and O’Nion families have always competed with each other. One arena for competition is Myami’s agricultural networking community, LeekedIn. The Begas and O’Nions each want to know whether their farming network has the larger clique. Define the language:
Problem 3

Define the **kernel** of a directed graph $G = (V,E)$ to be some $K \subseteq V$ such that:

1. $\forall k_1, k_2 \in K$, $(k_1, k_2) \notin E$, and
2. $\forall v \in V - K$, $\exists k \in K$ such that $(k, v) \in E$.

Let $L$ be the language of all directed graphs $G$ which have a kernel. Show that $L$ is NP-complete by a reduction from 3SAT. (12 points)

**Hint:** You will need a node for every variable and its complement along with three nodes for every clause. The kernel, if it exists, should contain at least the satisfying assignment of the Boolean formula. It may be helpful to review the proof of NP-completeness for 3-colorability.

Problem 4

We define the language **SetSplit** as follows.

$$\text{SetSplit} = \{ \langle S, F \rangle : S \text{ is a set, and } F \text{ is a set of subsets of } S \text{ and there is a partition of } S \text{ into two subsets } S_1 \text{ and } S_2 \text{ such that no subset in } F \text{ is entirely contained in either } S_1 \text{ or } S_2. \}$$

For example, if $S = \{1, 2, 3, 4\}$ and $F = \{\{1, 2\}, \{3, 4\}\}$, then $F \in \text{SetSplit}$ since we can choose $S_1 = \{1, 3\}$ and $S_2 = \{2, 4\}$. 
Recall the variation of 3-SAT called 3-\textsc{Not-All-Equal-SAT} (3-\textsc{NAE-SAT}).

\begin{equation*}
3\text{-}\textsc{NAE-SAT} = \{ \langle C, n \rangle : C \text{ is a collection of triples } C_1, C_2, \ldots, C_m \text{ of literals over } n \text{ boolean variables } x_1, x_2, \ldots, x_n \text{ such that there exists } a_1, a_2, \ldots, a_n \text{ such that every triple } C_i \text{ has a true and false literal} \}
\end{equation*}

Given \( \langle C, n \rangle \), each triple consists of three literals \((x_i, x_j, x_k)\). If each variable can be assigned such that for all triples, not all literals have the same truth value, then \( \langle C, n \rangle \) is an instance of 3-\textsc{NAE-SAT}.

Using a reduction from 3-\textsc{NAE-SAT}, prove that \textsc{SetSplit} is \textsc{NP}-Complete.

\textbf{(14 points)}

\textbf{Problem 5}

The year is 1997, and IBM’s Deep Blue is all the rage! It just beat Garry Kasparov, the world chess champion. How did it do it? One trick is that it has a clever heuristic that can tell whether there is a move to checkmate one’s opponent.

Suppose that Deep Blue is given a Boolean formula \( \phi \) over two sets of variables, \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_m \). (The formula \( \phi \) somehow captures the current chess board.) Deep Blue gets to decide how to set the truth values of \( x_i \)’s while Garry Kasparov gets to assign the values to \( y_1, \ldots, y_m \). Deep Blue can win if there exists a setting to \( x_1, \ldots, x_n \) (i.e. a move) such that no matter how Kasparov sets \( y_1, \ldots, y_m \), \( \phi \) evaluates to True. Formally, Deep Blue’s success hinges on its ability to decide the following language.

\begin{equation*}
\text{CheckMate} = \{ \langle \phi(x_1, \ldots, x_n, y_1, \ldots, y_m) \rangle : \text{there exists } a_1, \ldots, a_n \text{ such that for all } b_1, \ldots, b_m, \phi(a_1, \ldots, a_n, b_1, \ldots, b_m) = T \}
\end{equation*}

1. Recall the language \textsc{ChromaticNumber}.

\begin{equation*}
\text{ChromaticNumber} = \{ \langle G, k \rangle : G \text{ is a } k\text{-colorable graph that cannot be colored with } k - 1 \text{ colors} \}.
\end{equation*}

Show that \textsc{ChromaticNumber} \( \leq_p \text{CheckMate} \). \hfill (8 points)

\textbf{Hint:} The Cook-Levin theorem gives us a procedure that computes a Boolean formula \( \psi \) from the values \( \langle G, k \rangle \) such that \( \psi \) is satisfiable.
if and only if $G$ is $k$-colorable. You can use this procedure (without worrying about how it works) in your reduction.

2. Explain why part (1) implies that CheckMate is both NP- and co-NP-hard.

(4 points)

3. Suppose that researchers at IBM have developed a top secret algorithm $A$ that decides SAT in polynomial time. Give an algorithm that decides CheckMate in polynomial time using $A$ as a subroutine.

(8 points)

**Hint:** It will actually help if, as an intermediate step, you first come up with a non-deterministic polynomial-time algorithm.