1. For any integer k, the problem k-COLOR asks whether the vertices of a given graph G can be colored using at most k colors so that neighboring vertices does not have the same color.

   (a) Prove that k-COLOR is NP-hard, for every integer $k \geq 3$.

   (b) Now fix an integer $k \geq 3$. Suppose you are given a magic black box that can determine **in polynomial time** whether an arbitrary graph is k-colorable; the box returns TRUE if the given graph is k-colorable and FALSE otherwise. The input to the magic black box is a graph. Just a graph. Vertices and edges. Nothing else.

   Describe and analyze a **polynomial-time** algorithm that either computes a proper k-coloring of a given graph G or correctly reports that no such coloring exists, using this magic black box as a subroutine.

2. A boolean formula is in *conjunctive normal form* (or *CNF*) if it consists of a *conjuction* (AND) or several *terms*, each of which is the disjunction (OR) of one or more literals. For example, the formula

$$(\overline{x} \vee y \vee \overline{z}) \wedge (y \vee z) \wedge (x \vee \overline{y} \vee \overline{z})$$

   is in conjunctive normal form. The problem ***CNF-SAT*** asks whether a boolean formula in conjunctive normal form is satisfiable. 3SAT is the special case of CNF-SAT where every clause in the input formula must have exactly three literals; it follows immediately that CNF-SAT is NP-hard.

   Symmetrically, a boolean formula is in *disjunctive normal form* (or *DNF*) if it consists of a *disjunction* (OR) or several *terms*, each of which is the conjunction (AND) of one or more literals. For example, the formula

$$(\overline{x} \wedge y \wedge \overline{z}) \vee (y \wedge z) \vee (x \wedge \overline{y} \wedge \overline{z})$$

   is in disjunctive normal form. The problem DNF-SAT asks whether a boolean formula in disjunctive normal form is satisfiable.

   (a) Describe a polynomial-time algorithm to solve DNF-SAT.

   (b) Describe a reduction from CNF-SAT to DNF-SAT.

   (c) Why do parts (a) and (b) not imply that P=NP?

3. The 42-PARTITION problem asks whether a given set S of n positive integers can be partitioned into subsets A and B (meaning $A \cup B = S$ and $A \cap B = \varnothing$) such that

$$\sum_{a \in A} a = 42 \sum_{b \in B} b$$

   For example, we can 42-partition the set $\{1, 2, 34, 40, 52\}$ into $A = \{34, 40, 52\}$ and $B = \{1, 2\}$, since $\sum A = 126 = 42 \cdot 3$ and $\sum B = 3$. But the set $\{4, 8, 15, 16, 23, 42\}$ cannot be 42-partitioned.

   (a) Prove that 42-PARTITION is NP-hard.

   (b) Let M denote the largest integer in the input set S. Describe an algorithm to solve 42-PARTITION in time polynomial in n and M. For example, your algorithm should return TRUE when $S = \{1, 2, 34, 40, 52\}$ and FALSE when $S = \{4, 8, 15, 16, 23, 42\}$.

   (c) Why do parts (a) and (b) not imply that P=NP?