Assignment 6

COMP 4500
Due: April 5, 2024 by 11:59 p.m.
Upload your Word, PDF, or LaTeX file(s) to Blackboard

Exercises from the Textbook (60 points):

Page 415:

2	(10 points)
4	(10 points)

Page 505:

1	(10 points)
2	(10 points)
4	(20 points)

Additional Exercises (40 points):

1. The version of SAT introduced by the book uses what's called conjunctive normal form (CNF). In CNF, a Boolean expression is made up of **clauses**, each of which is ANDed together. A clause is made up of **terms**, each of which is ORed together. A term is either a Boolean variable x_i or its negation $\overline{x_i}$. In other words, CNF is an AND of ORs.
   There is another version of SAT that uses disjunctive normal form (DNF). In DNF, clauses are all ORed together. Furthermore, a clause is made up of terms that are ANDed together. In other words, DNF is an OR of ANDs.
   Here's an example of a SAT problem presented in DNF:
   $$(x_1 \wedge \overline{x_3}) \vee (\overline{x_1} \wedge x_2 \wedge \overline{x_3}) \vee (x_1 \wedge \overline{x_2} \wedge x_3)$$

   The following statements are facts:
   i. An instance of SAT in DNF can be solved in polynomial time, since all that's needed is to try to set all of the terms in a clause to true, which will fail only if a term and its negation is present in the same clause.
   ii. It is possible to convert any Boolean formula in CNF into a logically equivalent one in DNF.

Given statements (i) and (ii) above, it appears that SAT (in CNF) could be solved in polynomial time, meaning that **P** = **NP**! If that's true, write up an explanation, submit it to the Clay Mathematics Institute, and collect your $1 million in two years. Otherwise, please explain why these two facts do not prove that SAT (in CNF) could be solved in polynomial time, leaving us no closer to resolving the **P** = **NP** debate.

2. In the graph-coloring problem, the goal is to determine the minimum number of colors needed to color the vertices of an undirected graph such that no two adjacent vertices have the same color.
   a. Describe an algorithm to color a graph with two colors, if such a thing is possible on the given graph. Your algorithm should be polynomial in the number of vertices and edges.
   b. Consider a special case of the graph-coloring problem, the seven-coloring problem:

      > Given an undirected graph $\boldsymbol{G} = (\boldsymbol{V},\boldsymbol{E})$, is it possible to use only seven colors to color each vertex $\boldsymbol{v} \in \boldsymbol{V}$ such that no two adjacent vertices have the same color?

      Next, consider the three-coloring problem:

      > Given an undirected graph $\boldsymbol{G} = (\boldsymbol{V},\boldsymbol{E})$, is it possible to use only three colors to color each vertex $\boldsymbol{v} \in \boldsymbol{V}$ such that no two adjacent vertices have the same color?

      Show that, if the three-coloring problem is **NP-complete**, then the seven-coloring is also **NP-complete**.