Exercise 11.1.1 b)

Consider the problem \text{FALSE-SAT}:

Given a boolean expression \( E \) that is false when all its variables are made false, is there some other truth assignment that makes \( E \) false besides all-false?

Decide whether the problem is in \( \text{NP} \) or \( \text{coNP} \).

Describe its complement.

If the problem or its complement is \( \text{NP-complete} \), prove it.

Proof:

The problem is \( \text{NP-complete} \).

- In \( \text{NP} \): given a boolean expression \( E \), we need to check:
  1) that \( E \) is false when all variables are assigned false
  2) that there is some other truth assignment making \( E \) false

  (1) can be done in poly-time by a DTM
  (2) can be done in poly-time by a NDTM

  guess a truth assignment \( T \) different from all false, and answer yes if under \( T \), \( E \) evaluates to false

- \( \text{NP-hard} \): by a reduction from \( \text{SAT} \)

  Let \( E \) be a boolean expression with variables \( x_1, \ldots, x_n \);
  we construct an expression \( E' \) s.t. \( E \in \text{SAT} \) iff \( E' \in \text{FALSE-SAT} \)

  1) test if \( E \) is true when all variables are false (polynomial)
      if so, \( E \in \text{SAT} \), and we convert it to a fixed expression
      that is in \( \text{FALSE-SAT} \), e.g. \( \neg x \land y \).
2) Otherwise, let $E'$ be $\neg E \land (x_1 \lor x_2 \lor \ldots \lor x_n)$. 

Clearly, the reduction is poly-time.

We have that $E'$ is false when all of $x_1, \ldots, x_n$ are false.

Notice that in case (2), $E'$ is false when all variables are false.

Hence, if $E \in \text{SAT}$, then it is satisfied by a truth assignment $T$ different from all-false.

Thus, $\neg E$ is made false by $T$, and $E' \in \text{FALSE-SAT}$.

Conversely, if $E' \in \text{FALSE-SAT}$, then since $x_1, x_2, \ldots, x_n$ is false only for the all-false truth assignment, there must be some other truth-assignment $T$ that makes $\neg E$ false. Then $T$ makes $E$ true, and $E \in \text{SAT}$. 
Exercises on problems in \( P \), \( NP \), and \( NP \)-complete

**Exercise 4:**

Consider the following optimization version of SAT:

**MAXSAT:** Input: a propositional formula \( F \) in CNF, and an integer \( k \).

Output: yes, if there is a truth assignment that satisfies at least \( k \) clauses of \( F \).

no, otherwise.

What is the complexity of **MAXSAT**?

a) **MAXSAT \( \in \) \( NP \)**: immediate, by the following \( NP \) algorithm:

1) guess a truth assignment \( \alpha \) (non-deterministic polynomial)

2) count the \# of clauses satisfied by \( \alpha \), and answer yes iff it is \( \geq k \) (deterministic polynomial).

b) **MAXSAT \( \in \) \( NP \)-hard**.

This follows from the fact that **CSAT** is a special case of **MAXSAT**.

Formally, we can polynomially reduce **CSAT** to **MAXSAT**, i.e.

\[ SAT \leq_{poly} MAXSAT \]

Given an instance \( F \) of **CSAT**, we construct an instance \((F, k) \) of **MAXSAT**, where \( k \) is the \# of clauses of \( F \).

Obviously, \( k \) can be obtained in polytime from \( F \), and

\[ F \in \text{CSAT} \iff (F, k) \in \text{MAXSAT} \]