Problem 1 (25 points) It is known that 3-SAT is NP-complete. Show that 4-SAT is NP-complete. (Don’t forget to show that it is in NP.)

Ans: To show that 4-SAT is NP-complete, we prove that 4-SAT is in NP and NP-hard.
First, 4-SAT is in NP, we can write a nondeterministic polynomial-time algorithm which takes a 4-SAT instance and a proposed truth assignment as input. This algorithm evaluates the 4-SAT instance with the truth assignment. If the 4-SAT instance evaluates to true, the algorithm outputs yes; otherwise, the algorithm outputs no. This runs in polynomial time.
To prove that 4-SAT is NP-hard, we reduce 3-SAT to 4-SAT as follows. Let \( \phi \) denote an instance of 3-SAT. We convert \( \phi \) to a 4-SAT instance \( \phi' \) by turning each clause \((x \lor y \lor z)\) in \( \phi \) to \((x \lor y \lor z \lor h) \land (x \lor y \lor z \lor \neg h)\), where \( h \) is a new variable. Clearly this is polynomial-time doable.

\[\Rightarrow\] If a given clause \((x \lor y \lor z)\) is satisfied by a truth assignment, then \((x \lor y \lor z \lor h) \land (x \lor y \lor z \lor \neg h)\) is satisfied by the same truth assignment with \( h \) arbitrarily set. Thus if \( \phi \) is satisfiable, \( \phi' \) is satisfiable.

\[\Leftarrow\] Suppose \( \phi' \) is satisfied by a truth assignment \( T \). Then \((x \lor y \lor z \lor h) \land (x \lor y \lor z \lor \neg h)\) must be true under \( T \). As \( h \) and \( \neg h \) assume different truth values, \( x \lor y \lor z \) must be true under \( T \) as well. Thus \( \phi \) is satisfiable.

Problem 2 (25 points) Show that if there exists a language \( L \in \text{NP} \) not in \( P \), then no NP-complete language is in \( P \).

Ans: Suppose \( L \in \text{NP} \), \( L \notin P \). Now, if there is an \( L' \in P \) which is NP-complete, then \( L \in \text{NP} \) can be reduced to \( L' \), and hence \( L \in P \), a contradiction.
Problem 3 (25 points) Show that $L \neq P$ or $P \neq PSPACE$.

Ans: Suppose $L = P$ and $P = PSPACE$ instead. Then $L = PSPACE$. However, we know these two classes are different by the space hierarchy theorem, a contradiction. ■

Problem 4 (25 points) Show that $\{M : M \text{ halts on all inputs}\}$ is not recursive.

Ans: We reduce halting problem to this problem. Given $M; x$, we construct the following machine $M'$:

$$M'(y) : \text{if } y = x \text{ then } M(x) \text{ else halt.}$$

Obviously, $M'$ halts on all inputs if and only is $M$ halts on $x$. ■