Exercise 1.

The PESSIMISTIC SAT problem is the equivalent Horn-satisfiability problem in literature.
To prove the HORN SAT problem is in P, we define a TM M that decides the problem in $O(p(|w|))$ for some polynomial p. M follows an algorithm based on unit propagation:

- Find a unit clause (clause that consists of a single literal x) in the formula S. For S to be satisfiable, we must have the assignment $\pi(x) = T$. We apply two rules of unit propagation:

  1. remove all clauses that contain the literal x since these clauses are automatically satisfiable because $x = T$ and removing satisfiable clauses from S does not change its semantics.
  2. remove the negation $\neg x$ from the clause it is in since $\pi(\neg x) = F$ and $c' \vee F = c'$ for any subclause c', i.e. removing $\neg x$ does not change the meaning of the clause.

  The second rule might produce new unit clauses, which are propagated in the same manner. The stopping condition for the propagation is either of the three cases:

  1. The resulted formula S' is empty. This means all clauses are satisfied by the assignment obtained through unit propagation. Hence S is satisfiable.
  2. The resulted formula S' contains no unit clause, i.e. each clause in S' consists of two or more literals. Since a horn clause contains at most one positive literal, we can assign all variables in S' to F i.e. $\pi(var) = F$ for $\forall var \in S'$. With this truth assignment, there will be at least one negative literal that is made T in each clause, hence all clauses in S' are satisfiable. S', therefore S, is satisfiable.
  3. The resulted formula S' contains an empty clause. This is the situation when both a literal and its negation appears as unit clauses in an intermediate step of propagation. Rule 2 removes the negation such that the clause becomes empty. S', therefore S, is unsatisfiable since there is no truth assignment to make the empty clause true.

Given a coding w of a formula S that consists of only Horn clauses, to implement the unit propagation on w, M scans through w each time to identify a unit clause. M uses state to remember the literal x, removes it from the original tape by replacing the cells with new tape symbols e.g. †'s, scans through w again to find any clause containing x or $\neg x$ and remove them (replace with †'s) according to rules of unit propagation. M repeats this unit propagation process.
M accepts w if there is no unit clause left on the tape, or all clauses are removed (replaced wtih †'s). M rejects if some, but not all, clause is empty. By construction, M decides the language since it accepts iff S is satisfiable. The decision process requires $O(|w|^2)$ steps since the number of clauses in S is bounded by $|w|$, and in the worst case we remove all of them by $|w|$ rounds of scanning, and each scan requires $O(2|w|)$ steps (back and forth). Hence, M decides the problem in polytime and the problem is in P.

Exercise 2.

The Halting problem is in NP-hard but not in NP.

- To prove the Halting problem is in NP-hard, we define a polytime reduction from SAT to Halting. Since SAT is NP-hard, every $A \in$ NP is P reducible to SAT, and by concatenation with our defined P reduction from SAT to Halting, every $A \in$ NP is therefore P reducible to Halting. Hence, Halting is NP-hard.
  P-Reduction. For an instance ϕ of SAT, we map it to an instance $\langle M \rangle 111w$ of Halting, where
  - M is the TM that takes ψ as input, iterates over all truth assignments to the variables of ψ and accepts if there is an assignment that makes ψ true, or enters a loop (never halts) if there is no satisfying assignment.
  - $w = \phi$

  We show that ϕ is satisfiable iff M accepts ϕ.
  - $\Rightarrow$. If ϕ is satisfiable, then $\exists \pi$ such that $\pi \models \phi$. M will eventually find the satisfying π for ϕ and accepts.
  - $\Leftarrow$. Show its contraposition. If ϕ is not satisfiable, then no variable assignments can make ϕ true, hence M never halts on ϕ.

  Now we show that the reduction can be done in polytime in terms of size of ϕ, i.e. there is a TM T that given the input ϕ, outputs $\langle M \rangle 111\phi$ on tape in poly steps in terms of $|\phi|$. First, we note that the transition function of M does not depend on the input ϕ. Given an input ψ, M first reads the number of variables in ψ and then tries truth assignments repeatedly until there is a satisfying assignment. Hence the size of encoding $\langle M \rangle$ is constant in terms of $|\phi|$. This means that T can write $\langle M \rangle$ on tape in constant time. T then appends ϕ to the back of $\langle M \rangle 111$ in linear time in terms of $|\phi|$. Hence T can convert ϕ into $\langle M \rangle 111\phi$ in poly-time, and SAT is poly-time reducible to Halting. Hence Halting is NP-hard.

- Halting is not in NP since all problems in NP are decidable by some (non-deterministic poly-time) TMs, yet there is no TM that can decide the Halting problem.
  Proof. By contradiction, suppose Halting is decidable. Then there exists a TM T that given $\langle M \rangle 111w$ accepts iff M halts on w. We can construct a TM T_2 that runs as follows:
  - T_2 takes as input $\langle \mathcal{M} \rangle$ and appends it with $111\langle \mathcal{M} \rangle$
  - T_2 moves to the leftmost of input string and simulate T on $\langle \mathcal{M} \rangle 111 \langle \mathcal{M} \rangle$.
    * If T accepts $\langle \mathcal{M} \rangle 111 \langle \mathcal{M} \rangle$, T_2 loops forever on $\langle \mathcal{M} \rangle$.
    * Else, T_2 halts on $\langle \mathcal{M} \rangle$.

  Now run T_2 on $\langle T_2 \rangle$. There are two situations:
  - T_2 halts on $\langle T_2 \rangle \Rightarrow \langle T_2 \rangle 111 \langle T_2 \rangle \in$ Halting $\Rightarrow T$ accepts $\langle T_2 \rangle 111 \langle T_2 \rangle \Rightarrow T_2$ loops forever on $\langle T_2 \rangle$. Contradiction.

– T_2 does not halt on $\langle T_2 \rangle \Rightarrow \langle T_2 \rangle 111 \langle T_2 \rangle \notin \texttt{Halting} \Rightarrow T$ rejects $\langle T_2 \rangle 111 \langle T_2 \rangle \Rightarrow T_2$ halts on $\langle T_2 \rangle$. Contradiction.

Since either way we have a contradiction, T cannot exist and `Halting` is not decidable. Therefore, it is not in `NP`.

Exercise 3.

`JAILBREAK` is in essense the `CLIQUE` problem. To prove `JAILBREAK` is `NP-complete`, we show that it is both `NP` and `NP-hard`.

- We prove `JAILBREAK` is in `NP` by showing that the problem is poly-time verifiable. We construct a TM T that verifies the certificate of `JAILBREAK` in poly-time. The certificate is a solution to the problem i.e. a set of vertices of size $\geq k$. Note that in order to have a clique of size $\geq k$, the number of edges in G should be at least k. Hence, given the pair of problem description $\langle G(V, E), k \rangle$, and the certificate c, as an input to the verifier T:

  – T first checks if the number of edges in G is at least k. This requires only polynomially many steps in terms of size of the graph $|\langle G(V, E), k \rangle|$, because T only needs to traverse the encoding of the graph once to count the number of edges in G and if the number of edges is smaller than k, T rejects the certificate directly, because there cannot be a $\geq k$ clique for the graph. Hence, even in the worst case that the encoding of $|V|$ and k is in binary, and the graph does not contain any edges, T can still verify(reject) the certificate in polytime in terms of the size of the input graph.

  – If the number of edges is $\geq k$, then it must be the case that the input length $|\langle G(V, E), k \rangle| \geq k$ since we need to list each pair of edges in the encoding. T can then verify whether the certificate c is indeed a complete subgraph of G, i.e. all pairs of vertices in c are connected. To check if a pair is connected, T scans through the encoding of G to see if there is a corresponding edge, and each scan takes $O(|\langle G(V, E), k \rangle|)$ steps. There are $O(k^2)$ pairs to check. In total, the number of steps is $O(|\langle G(V, E), k \rangle|^3)$ since $k \leq |\langle G(V, E), k \rangle|$. Hence the TM T will verify the language in poly-time.

- We prove `JAILBREAK` is in `NP-hard` by showing `JAILBREAK` is poly-time reducible from a known `NP-hard` problem, in this case, the `3SAT` problem. Given an instance of `3SAT`, $\phi = c_1 \wedge c_2 \wedge ... \wedge c_k$, we construct an instance of `JAILBREAK` $G_\phi(V, E), k$ where

  – $V = \{(i, \pi) : 1 \leq i \leq k,\ \pi : var(\phi) \mapsto \{T, F\} \text{ and } \pi \models c_i\}$ where $var(\psi)$ is the set of all variables in ψ

  – $E = \{((i, \pi), (i', \pi')) : \pi \text{ and } \pi' \text{ agree on } var(c_i) \cap var(c_{i'})\}$

  We show that ϕ is satisfiable iff $G_\phi(V, E)$ has a clique of size $\geq k$.

  – $\Rightarrow$. If ϕ is satisfiable, then $\exists \pi$ such that $\pi \models c_i$ for $\forall i = 1..k$. A clique of size k is given by the set of vertices $S = \{(i, \pi_{var(c_i)}) : 1 \leq i \leq k\}$ where $\pi_{var(c_i)}$ is the variable assignment π restricted

to variables in c_i.

For any pair of vertices $(i, \pi_{var(c_i)})$ and $(i', \pi_{var(c_{i'})})$ of S, they are connected by an edge since $\pi_{var(c_i)}$ and $\pi_{var(c_{i'})}$ agree on $var(c_i) \cap var(c_{i'})$ as π is the same. Size of S is k since we have k vertices. Hence, $G_\phi(V, E)$ has a clique of size $\geq k$.

- $\Leftarrow$. If $G_\phi(V, E)$ has a clique of size $\geq k$, then it must be the case that the size of the clique is exactly k and the k-clique is given by $\{(i, \pi_i) : i = 1..k\}$. This is because by definition of V, any two vertices with the same i cannot be connected because the truth assignments π and π' will not agree on variables in the same clause. Hence, any two connected vertices in the graph must have different i's. There are in total k different values for i, so the clique must be of size k. Also by definition of V, $\pi_i \models c_i$, so all c_i's are satisfiable. Hence, we define $\pi_\phi(x) = \pi_i(x)$ if $x \in var(c_i)$. π_ϕ is well-defined since all π_i's agree on the shared variables by definition of E. Hence π_ϕ is a satisfying assignment of ϕ, which means ϕ is satisfiable.

Now we show that the reduction can be done in polytime in terms of size of ϕ. In the worst case, a k-clause ϕ has $m \leq 3k$ variables and a size of $O(3\log(m) * k) = O(3\log(3k) * k) = O(k^2)$. In the mapping, we construct at most 8 vertices for each c_i, as we have $2^3 = 8$ assignments for a clause of at most 3 variables. Also the assignments should be satisfying, so in reality the number will be less than 8. Hence, the number of vertices in V is upper bounded by $8k$. Writing each vertex on the tape requires $O(\log(k))$ steps to encode $i \leq k$ in binary, and $O(3)$ steps to encode π for 3 literals. This means that writing the entire V on tape takes $O(8k * \log(k)) = O(k^2)$ steps, which is polynomial (in fact linear) in terms of $|\phi| = O(k^2)$. Size of E is in $O(|V|^2)$, so writing E on tape takes $O(k^4)$ steps, which is polynomial in $|\phi|$. Hence, overall the mapping that takes ϕ as input and outputs $G_\phi(V, E), k$ on the tape can be done in poly time in terms of $|\phi|$. Hence, we have a poly-time reduction from `3SAT` to `JAILBREAK`, which means `JAILBREAK` is `NP-hard`.