Question 1

Required: Prove that if $P = NP$, then $EXP = NEXP$

Proof. Assume $P = NP$.

Trivially, we know that $EXP \subseteq NEXP$ since any deterministic exp-time TM can be viewed as a nondeterministic exp-time TM (without any non-trivial branching from the transition function). So any langauge decided by a deterministic exp-time TM can be decided by a nondeterministic exp-time TM.

Hence, we just need to show $NEXP \subseteq EXP$. Let $A \in NEXP$. Hence, there is a nondeterministic exp-time TM M that decides A in $\mathcal{O}(2^{n^k})$ where n is the input length.

We want to show $A \in EXP$. As the hint suggests, let $A' = \{x1^{2^{|x|^k}} : x \in A\}$ be the corresponding padded language.

First we will show that $A' \in NP$. Consider the following nondeterministic polytime TM N_1 that decides A'.

$N_1 =$ "on input w:

1. Test that $w = x1^{2^{|x|^k}}$ for some string x. If not, then reject immediately.

2. Run M on input x nondeterministically. If M accepts on any branch, then accept. Otherwise, reject."

Clearly N_1 is a decider for A' since M is a decider for A. Clearly Step 1 takes polynomial time with respect to the input length. Notice that on an input $w = x1^{2^{|x|^k}}$, on Step 2, M runs on x which takes time $\mathcal{O}(2^{|x|^k})$. Hence, the overall runtime of N_1 is polynomial with respect to the original input length $|x1^{2^{|x|^k}}|$. Hence, N_1 is a poly-time nondeterministic TM that decides A'.

Hence, $A' \in NP$. Since we assumed $P = NP$, we have that $A' \in P$.

Since $A' \in P$, there exists some poly-time deterministic TM N_2 that decides A'.

Now we will define an exp-time deterministic TM H that decides A.

$H =$ "on input x:

1. Let $w = x1^{2^{|x|^k}}$. i.e. w is a padding of x.

2. Run N_2 on w. If N_2 accepts, then accept. If N_2 rejects, then reject."

Clearly H is a decider for A since N_2 is a decider for A'. Step 1 takes exponential time with respect to the input length $|x|$ since Step 1 is an exponential padding of x. On Step 2, N_2 will run on $w = x1^{2^{|x|^k}}$. Since N_2 is polytime, we have that N_2 will run in polynomial time with respect to $|x1^{2^{|x|^k}}|$. But $|x1^{2^{|x|^k}}|$ is exponential with respect to the original input length $|x|$. Hence, H will run in exponential time in terms of $|x|$. Hence, H is an exp-time deterministic TM that decides A. Hence, $A \in EXP$.

Therefore, $NEXP \subseteq EXP$.

Therefore, we have shown $EXP = NEXP$, as required. $\square$

Question 2

Required: Show that $MINFORMULA \in PSPACE$.

Consider the following TM M that decides $MINFORMULA$ in polynomial space.

$M =$ "on input w:

1. Check that $w = \langle \phi \rangle$ for some formula ϕ. If not, reject immediately.

2. Otherwise, run the following loop.

For each string s such that $|s| < |\langle \phi \rangle|$, execute the following substeps i) and ii):

i) Check that $s = \langle \psi \rangle$ for some formula ψ such that ϕ and ψ have the same set of variables. If not, move on to the next iteration of the for loop (if there is a new iteration).

ii) Evaluate ϕ and ψ on every possible truth assignment. If ϕ and ψ evaluate to the same truth value on every truth assignment, then reject.

3. If we have not entered the reject state after Step 2, then accept."

Clearly M decides $MINFORMULA$ since if $w \in MINFORMULA$, then $w = \langle \phi \rangle$ for some formula ϕ. And since ϕ is minimal, there is no formula ψ with the same variables and is of shorter length that is equivalent to ϕ. Hence, we cannot possibly ever reject on Step 2 ii). Hence, we will accept on Step 3.

And if $w \notin MINFORMULA$, then either w is not an encoding of a formula and we will reject immediately on Step 1; or $w = \langle \phi \rangle$ such that there is some formula ψ with the same variables where $|\langle \psi \rangle| < |\langle \phi \rangle|$ and ψ is equivalent to ϕ. Our TM M will eventually find $\langle \psi \rangle$, and we will reject on an iteration of Step 2 ii).

Hence, M is a decider for $MINFORMULA$.

To show that M runs in polynomial space, notice that in Step 2, we are only considering strings s such that $|s| < |\langle \phi \rangle| = |w|$. So we have $\mathcal{O}(|w|)$ space here. And we can reuse this space for each string s that is considered. Similarly, for any $s = \langle \psi \rangle$, any truth assignment for ϕ and ψ will take $\mathcal{O}(m)$ space where m is the number of variables in both ϕ and ψ. We can reuse this space for each assignment that we are considering. And since the number of variables m is at most $|w| = |\langle \phi \rangle|$, we have $\mathcal{O}(|w|)$ space. So M runs in $\mathcal{O}(|w|)$ space.

Therefore, $MINFORMULA \in PSPACE$, as required.

Question 3

Required: Prove that if $P = NP$, then $MINFORMULA \in P$.

Proof. Assume $P = NP$.

First we will show that the following language L is such that $L \in NP$.

$L = \{\langle \phi, \psi \rangle : \phi$ and ψ have the same variables and ϕ is not equivalent to $\psi\}$

Consider the following verifier V_1 for L which uses a truth assignment that makes one of ϕ and ψ true, and the other false as a certificate.

$V_1 =$ "on input $\langle \langle \phi, \psi \rangle, c \rangle$:

1. Test that ϕ and ψ have the same variables.

2. Test that c is an encoding of a truth assignment to the variables in ϕ and ψ.

3. Test that one of ϕ and ψ is true under the assignment encoded by c, and the other is false.

4. If all tests pass, then accept. Otherwise reject."

Clearly V_1 is a verifier for L since if $w = \langle \phi, \psi \rangle \in L$, then ϕ and ψ have the same variables and are not equivalent, and hence there is a truth assignment that makes one true and the other false. This truth assignment can be encoded in a certificate c. And if $w \notin L$, then either w is not of the correct form, or $w = \langle \phi, \psi \rangle$ and ϕ and ψ are equivalent. Hence, there can be no valid certificate c.

To show that V_1 is a poly-time verifier, notice that any truth assignment just has to assign truth values to the variables in ϕ and ψ. The number of variables is less than $|\langle \phi, \psi \rangle|$. Hence, $|c| < |\langle \phi, \psi \rangle|$. And each test clearly takes polynomial steps with respect to the input length. Hence, V_1 is a poly-time verifier.

Hence, $L \in NP$. Since we assumed $P = NP$, we have that $L \in P$. Hence, there is a TM M that decides L in polynomial time.

Now consider the following verifier V_2 for $\overline{MINFORMULA}$.

$V_2 =$ "on input $\langle \phi, c \rangle$:

1. Test that $c = \langle \psi \rangle$ for some formula ψ such that $|\langle \psi \rangle| < |\langle \phi \rangle|$ and ϕ and ψ share the same set of variables. If not reject, immediately.

2. If Test 1 passes, run M on input $\langle \phi, \psi \rangle$. If M rejects, then accept. If M accepts, then reject. i.e. Do the opposite of what M does."

Clearly V_2 is a verifier for $\overline{MINFORMULA}$ since if $w \in \overline{MINFORMULA}$, then either w is not an encoding of a formula, or $w = \langle \phi \rangle$ such that ϕ is not minimal. Hence, there is some formula ψ with the same set of variables as ϕ such that $|\langle \psi \rangle| < |\langle \phi \rangle|$ and ϕ is equivalent to ψ. Hence, $c = \langle \psi \rangle$ is a certificate. And if $w \notin \overline{MINFORMULA}$, then $w = \langle \phi \rangle$ for some minimal formula ϕ. Hence, we cannot have a valid certificate c.

To show that V_2 is a poly-time verifier, first notice that an accepting certificate c is such that $c = |\langle \psi \rangle| < |\langle \phi \rangle|$. Clearly Step 1 takes polynomial time with respect to the length of the input. In Step 2, M is polynomial with respect to its input length $|\langle \phi, \psi \rangle|$. But since $|\langle \psi \rangle| < |\langle \phi \rangle|$, we know that $|\langle \phi, \psi \rangle| \leq k|\langle \phi \rangle|$ for some constant k. This constant k would depend on the specifics of our coding scheme. Regardless, we have that M runs in polynomial time with respect to $|\langle \phi \rangle|$. Hence, our verifier V_2 runs in polynomial time with respect to $|\langle \phi \rangle|$.

Hence, V_2 is a poly-time verifier for $\overline{MINFORMULA}$. Hence, $\overline{MINFORMULA} \in NP$.

Since $\overline{MINFORMULA} \in NP$ and since we assumed $P = NP$, we have that $\overline{MINFORMULA} \in P$.

Since $P = coP$ and $\overline{MINFORMULA} \in P$, we have that $\overline{MINFORMULA} \in coP$.

Since $\overline{MINFORMULA} \in coP$, we have that $MINFORMULA \in P$, as required.

$\square$

Question 4

Required: Show $BIPARTITE \in NL$.

We are given the hint that a graph is not bipartite if and only if it has a cycle with an odd number of vertices.

We also know that $NL = coNL$. Hence, it is sufficient to just show that $BIPARTITE \in coNL$. i.e. We must show that $\overline{BIPARTITE} \in NL$.

Consider the following nondeterministic log space TM M that decides $\overline{BIPARTITE}$.

$M =$ "on input $\langle G \rangle$:

1. Keep a counter c starting at $c = 0$.

2. Nondeterministically select some initial vertex s from G.

3. Nondeterministically select some vertex s' connected to s. Increment the counter so that $c \leftarrow c + 1$.

4. Repeat the following so long as $c < |V|$ where V is the set of vertices of G and so long as $s' \neq s$:

    i) Nondeterministically select some vertex s'' connected to s'. Let $s' \leftarrow s''$.
    ii) Increment the counter so that $c \leftarrow c + 1$.
    iii) If $s' = s$, then terminate the loop and move on to Step 5.

5. If $s = s'$ and c is odd, then accept. Otherwise, reject."

And we implicitly assume that if M is given a string w that is not an encoding of some graph G, then M rejects w.

Clearly M decides $\overline{BIPARTITE}$ given the hint that a graph is not bipartite if and only if it has a cycle with an odd number of vertices.

To see that M runs in log space, notice that the counter c is just a binary number less than or equal to the number of vertices in G. Hence, c can be stored in log space with respect to the input length. And the inital vertex s can be stored in log space with respect to the input length. The space used to store s' is log space with respect to the input length, and this space is reused each time we update s'. Hence, M runs in log space with respect to the input length.

Therefore, $\overline{BIPARTITE} \in NL$. Hence, $BIPARTITE \in coNL$. Since $NL = coNL$, we conclude that $BIPARTITE \in NL$.