Question 1: Correctness of Alpha-Beta Pruning (25 points)

Let s be the state of the game, and assume that the game tree has a finite number of vertices. Let v be the value produced by the minimax algorithm:

$$v = \texttt{Minimax}(s)$$

Let v' be the result of running Alpha-Beta Pruning on s with some initial values of α and β (where $-\infty \leq \alpha \leq \beta \leq +\infty$):

$$v' = \texttt{Alpha-Beta-Pruning}(s, \alpha, \beta)$$

Prove that the following statements are true:

- If $\alpha \leq v \leq \beta$ then $v' = v$
- If $v \leq \alpha$ then $v' \leq \alpha$
- If $v \geq \beta$ then $v' \geq \beta$

Let's prove the statements using induction:

1. If v = v', where v is the value produced by the minimax algorithm and v' is the result of running Alpha-Beta Pruning with some initial values of α and β:
    a. Base Case: If s is a terminal state, both minimax and Alpha-Beta Pruning produce the same value, which is v = v'.
    b. Inductive Step: Assume the claim holds for all children of s. In the minimax algorithm, v represents the best possible outcome for the current player. Similarly, in Alpha-Beta Pruning, v' represents the best possible outcome within the bounds of α and β. Since the bounds α and β are tightened as we traverse the tree, Alpha-Beta Pruning will not exceed the minimax value. Therefore, v = v'.

2. If v < v':
    a. This implies that the minimax value is less than the value produced by Alpha-Beta Pruning. Since Alpha-Beta Pruning only prunes branches where the value is worse than β, the value v' is a conservative estimate of the true minimax value.

3. If v > v':
    a. This implies that the minimax value is greater than the value produced by Alpha-Beta Pruning. Similarly, since Alpha-Beta Pruning only keeps values between α and β, the value v' is a conservative estimate of the true minimax value.

Therefore, the hypothesis holds true for all cases: if the true minimax value is within the range defined by the initial values of α and β, then Alpha-Beta Pruning returns the correct value. If the true minimax value lies outside this range, Alpha-Beta Pruning may return a different value, but

it is bounded by the same constraints as the true minimax value. Therefore, Alpha-Beta Pruning will be correct with initial values of (-1, +1) for (α, β).

Question 2: CSP Reduction (25 points)

Prove that any n-ary constraint can be converted into a set of binary constraints. Therefore, show that all CSPs can be converted into binary CSPs (and therefore we only need to worry about designing algorithms to process binary CSPs).

To prove that any n-ary constraint can be converted into a set of binary constraints, let's consider a constraint involving n variables: (x_1, x_2, ..., x_n)

1. Create Synthetic Variables:
    a. Introduce a synthetic variable, Y, with a domain that is the Cartesian product of the domains of (X_1, X_2, ..., X_n). The domain of Y would consist of tuples (x_1, x_2, ..., x_n) where x_i belongs to the domain of X_i for i = 1 to n.

2. Replace Constraint w/ Binary Constraints:
    a. Replace the original n-ary constraint with a set of binary constraints.
    b. Each binary constraint will relate Y with one of the original variables, X_i, for i = 1 to n.
    c. For each tuple d = (x_1, x_2, ..., x_n) in the domain of Y, create a binary constraint between Y and X_i stating that X_i = x_i.

These binary constraints will effectively enforce the original n-ary constraint.
By introducing synthetic variables and transforming the n-ary constraint into a set of binary constraints, we've shown that any n-ary constraint can indeed be converted into binary constraints. Therefore, all Constraint Satisfaction Problems (CSPs) can be converted into Binary CSPs by representing each n-ary constraint in terms of binary constraints.