Question 1: Shorties

(a) Given an instance of 3COL, G = (V,E). We construct an instance $G' = (V', E')$ as follow:
Let $V' = V \cup v, E' = E \cup \{uv | u \in V\}$. We could then run our black box for 4COL and return the same result it gives.
This construction adds 1 node and $|V|$ edges to the original graph. So it can be done in $O(|V|)$ time.

(b) The second part of step 2 in the construction is wrong. Because you don't know if and when M does not halt on y. So you could only decide what to do when M accepts/rejects y, but not when M loops on y.

(c) No. Because $P \subset NP$. Therefore, A could be a problem in P, which indeed can be solved by a poly-time algorithm.
Instead, what you want to do is first show that A is in NP-Complete and then find a polytime for A.

(d) $a \cup b \cup (a\{a, b\}^* a) \cup (b\{a, b\}^* b)$

Question 2: Mystery Grammar

Context free grammar G with production rule $S \rightarrow aS|Sb|bSa|\epsilon$.
The language generated by this grammar is $A = \{a, b\}^*$

First, we will show that $A \subset L(G)$,
We prove by induction that if $x \in A$, then $S \xrightarrow{*} x$ base case: $S \rightarrow \epsilon$, $S \rightarrow aS \rightarrow a$,$S \rightarrow Sb \rightarrow b$,$S \rightarrow bSa \rightarrow ba$,$S \rightarrow aS \rightarrow aSb \rightarrow ab$
Induction hypothesis: For some $k \geq 2$, if $x \in A$ and $|x| \leq k$ then $y \in L(G)$
Induction steps: Consider $y \in A$ with $|y| = k + 1$, we must have y = ax or y= xb or y = bxa for some x, where x is itself in A and $|x| \leq k$, and thus our inductive hypothesis implies that $S \xrightarrow{*} x$
If y = ax then we can use the derivation $S \xrightarrow{1} aS \xrightarrow{*} ax = y$. If y = xb then we can use the derivation $S \xrightarrow{1} Sb \xrightarrow{*} xb = y$. If y = bxa then we can use the derivation $S \xrightarrow{1} bSa \xrightarrow{*} bxa = y$
Thus we have prove by induction that $A \subset L(G)$

Then we show that $L(G) \subset A$
We prove that for all $i \geq 1$ if $S \xrightarrow{i} x$ then $x \in A$ by induction:
Base case: $S \xrightarrow{1} \epsilon$ and $\epsilon \in A$
Induction hypothesis: For all $i \leq k$, we have if $S \xrightarrow{i} x$, then $x \in A$
Induction steps: We now consider y s.t. $S \xrightarrow{k+1} y$
There will be 3 cases:
$y \xrightarrow{1} aS \xrightarrow{k} y$. Then $y = ax$ for some $x \in \Sigma^*$. By induction hypothesis,$x \in A$, therefore, $y = ax \in A$
$y \xrightarrow{1} Sb \xrightarrow{k} y$. Then $y = xb$ for some $x \in \Sigma^*$. By induction hypothesis,$x \in A$, therefore, $y = xb \in A$
$y \xrightarrow{1} bSa \xrightarrow{k} y$. Then $y = bxa$ for some $x \in \Sigma^*$. By induction hypothesis,$x \in A$, therefore, $y = bxa \in A$
Thus, we have shown that $L(G) \subset A$.

In conclusion, we have shown that $A \subset L(G)$ and $L(G) \subset A$. Therefore, A = L(G).

Question 3: Balanced Concat

$A \heartsuit B = \{xy | x \in A, y \in B, |x| = |y|\}$

(a) $A = \{a\}^*$, $B = \{b\}^*$

(b) A and B are regular sets accepted by DFAs $M_A = (Q_A, \Sigma, \delta_A, s_A, F_A)$ and $M_B = (Q_B, \Sigma, \delta_B, s_B, F_B)$
We now construct a NPDA $M = (Q, \Sigma, \delta, \Gamma, s, F)$ that accepts $A \heartsuit B$

$Q = Q_A \times Q_B \times \{0, 1, 2\}$
$\Gamma = \{C, \$\}$
$s = (s_A, s_B)$

$\delta((q_1, q_2, 0), \epsilon, \epsilon) = ((q_1, q_2, 0), \$)$
$\delta((q_1, q_2, 0), x, \epsilon) = ((\delta_A(q_1), q_2, 0), C)$
$\delta((q_1, q_2, 0), \epsilon, \epsilon) = ((q_1, q_2, 1), \epsilon)$
$\delta((q_1, q_2, 1), x, C) = ((q_1, \delta_B(q_2), 1), \epsilon)$
$\delta((q_1, q_2, 1), \epsilon, \$) = ((q_1, q_2, 2), \epsilon)$

$(q_1, q_2, 2) \in F$ if $q_1 \in F_A, q_2 \in F_B$

The state of this machine will be in the form (q1,q2,n) where q1 is a state of M_A and q2 is a state of M_B. n is 0 when we are treating the input character as a character from A and n is 1 when we are treating the input character as a character from B. n is 2 when we the condition $|x| = |y|$ is satisfied

The machine will write a $ on the tape in the beginning.
It will then assume the input character is a character of A and add a C on the tape
It have an ϵ transition from state $(q_1, q_2, 0)$ to $(q_1, q_2, 1)$. This utilise the fact that it is nondeterministic to try all the possible division of the input string. It then treat the input character as a character of B and pop a C from the tape. When it see a $ on the tape, this means that we have satisfied the condition that $|x| = |y|$. If there is no more input it will then check if we accept the state.

Question 4: Self-Acceptance

$C = \{\langle M \rangle |$ M is a Turing machine that accepts $\langle M \rangle\}$

(a) Rice's Theorem does not apply here. Because this is a property of the machine not the language.

(b) First, show that C is recognisable.
We could just simulate M on $\langle M \rangle$. if this accepts then accepts. We have built a machine that recognise C.

Now, to prove it is not decidable by showing $A_{TM} \leq_M C$
$f =$ "on input $\langle M, x \rangle$:

1. compute/construct a TM M_1 such that $\langle M, x \rangle \in \mathsf{A}_{TM} \iff \langle M_1 \rangle \in \mathsf{C}$:

   $M_1 =$":on input $\langle M_1 \rangle$
   1. Ignore $\langle M_1 \rangle$.
   2. Run M on x, accept y if and only if M accepts x. "
2. Return $\langle M_1 \rangle$. "

We claim that M halts on x if and only if $\langle M_1, M_2 \rangle$ is in F.

$$\begin{aligned}\langle M, x \rangle \in A_{TM} &\Rightarrow M \text{ halts on } x \\ &\Rightarrow M_1 \text{ accepts } \langle M_1 \rangle \\ &\Rightarrow f(\langle M \rangle) \in C\end{aligned}$$

$$\begin{aligned}\langle M, x \rangle \notin A_{TM} &\Rightarrow M \text{ does not halt on } x \\ &\Rightarrow M_1 \text{ does not accept} M_1 \\ &\Rightarrow f(\langle M \rangle) \notin C\end{aligned}$$

Thus, we have shown that $A_{TM} \leq_M C$. Since A_{TM} is not Turing decidable, we know that C is also not decidable

Question 5: Box Packing

First, We will show that Box Packing is NP.
Our certificate will be satisfying arrangement A for boxes. Given A, our verifier will check the there is no overlap, and there should be no remaining empty space. Doing this is in time proportional to the size of ϕ (O(WH) time). Conversely if A is a certificate accepted by our verifier , then we know that A is an assignment for ϕ.
Thus we have shown that Box Packing is NP.

We now show that SUBSET SUM $<_p$ BOX PACKING.
For an instance of SUBSET SUM, set X = (x1,x2,...,xn) and target value is M. Construct a BOX PACKING problem as follow. For every $x \in X$, construct a box of width x and height 1. Then construct a van with width M and height 1.
We now run our black box for Independent Set on BOX PACKING problem and return the same result it gives.
Runtime: we construct as many boxes as the number of element in X and we also construct a van, so it takes O(n) time.
To finish the proof, we simply need to show that there is an arrangement for BOX PACKING if and only of there is a solution for SUBSET SUM.

Claim. If there is a subset of X sums to M then there is a satisfying assignment for BOX PACKING.
Suppose SUBSET SUM problem has a solution $\{y1, y2, ..., yk\}$, then we know y1+..+yk = M. The corresponding boxes will have size y1,...,yk and the van have size M. Since all boxes and van have height 1, we can just put the boxes in a line an avoid overlapping.Thus the BOX with width y1,y1,...,yk will be a solution for BOX PACKING.
Claim. If there is a satisfying assignment for BOX PACKING then there is a subset of X sums to M .
Suppose BOX PACKING problem has a satisfying arrangement with boxes of width y1,...,yk. Since height of boxes are 1, they will have area y1,y2,...,yk. Since there's no overlap and no remaining empty space, we have y1+...+yk=M. So $y1, ..., yk$ is a solution for SUBSET SUM problem

In conclusion, BOX PACKING is NP complete