1.

The language $SUBSET_{CFG}^{DFA}$ is decidable. If $L(G) \subseteq L(M)$ then $w \in L(G)$ implies $w \in L(M)$ and therefore $w \notin \overline{L(M)}$. Therefore, the languages $L(G)$ and $\overline{L(M)}$ must be disjoint. We know that given M we can construct a DFA $\overline{M}$ with $L(\overline{M}) = \overline{L(M)}$ by interchanging the final and non-final states of M. We also can effectively construct a PDA P with L(P) = L(G). Given these machines we can construct a PDA for $L(P) \cap L(\overline{M})$ by giving the machine states that keep track of both the current state of P and the current state of $\overline{M}$. Finally, given this PDA we can construct a CFG for its language. We also know that we can decide emptiness for the language of a CFG. By applying the emptiness procedure for the CFG for for $L(G) \cup L(\overline{M})$ we can decide whether $L(G) \subseteq L(M)$.

The language $SUBSET_{DFA}^{CFG}$ is not decidable. If it were decidable, given $\langle M, w\rangle$ we could decide whether $\langle M, w\rangle \in A_{TM}$ by constructing a CFG, H, for the complement of the accepting computation histories of M (with alternating configurations reversed) and checking to see whether $\Sigma^* \subseteq L(H)$. If not, we know that $L(H) \neq \Sigma^*$ implying that M accepts w. This is the same proof technique we used to show that ALL_{CFG} is not decidable. In fact, an alternate approach to this problem is to show that ALL_{CFG} can be reduced to $SUBSET_{DFA}^{CFG}$.

2.

If $A \leq_m A_{TM}$ then there is some computable function f such that $f(w) \in A_{TM} \iff w \in A$. We know that there exists a machine $M_{A_{TM}}$ that recognizes A_{TM}. Therefore, we could construct a machine M_A that recognized A by having that machine first apply f to its input and then run $A \leq_m A_{TM}$ on the result f produces.

If A is recognizable, then there must exist some TM M_A that recognizes A. The computable function $f(w) = \langle M_A, w\rangle$ then shows that $A \leq_m A_{TM}$.

3.

(a) The key thing to recognize here is that "decidable" means decidable by a Turing machine.

One of the direct consequences of the assumption the problem makes about DMs is that there is a Turing machine UD that decides the language $\{\langle D, w\rangle \mid$ D is a DM and $w \in L(D)\}$. We can build a Turing machine that decides $REJECT_{DM}$ using UD. In particular, we can build M_{REJECT} that operates as follows:

- On input w, if w is not an encoding of a DM, reject
- Otherwise, given $w = \langle D\rangle$, make a copy of w on the machine's tape including extra delimiting symbols as needed to form $\langle D, \langle D\rangle\rangle$.

- Run UD on $\langle D, \langle D\rangle\rangle$ and accept if it rejects or reject if it accepts (we can do this since UD is a decider).

Given this construction, it is clear that $L(M_{REJECT}) = REJECT_{DM}$ showing that this language is decidable.

(b) Suppose that the machine D_{REJ} is a DM that decides $REJECT_{DM}$. That is, assume that $L(D_{REJ}) = REJECT_{DM}$.

Consider whether $\langle D_{REJ}\rangle \in L(D_{REJ})$. If $\langle D_{REJ}\rangle \in L(D_{REJ})$ then D is not a DM that rejects its own description, so $\langle D_{REJ}\rangle \notin REJECT_{DM} = L(D_{REJ})$. That is, $\langle D_{REJ}\rangle \in L(D_{REJ})$ implies $\langle D_{REJ}\rangle \notin L(D_{REJ})$ which is a contradiction. Similarly, if we assume $\langle D_{REJ}\rangle \notin L(D_{REJ})$ then D_{REJ} is a DM that rejects its own description so $\langle D_{REJ}\rangle \in REJECT_{DM} = L(D_{REJ})$, a similar contradiction. From these contradictions, we can conclude that our assumption that $L(D_{REJ}) = REJECT_{DM}$ must be false and that there is no DM that decides $REJECT_{DM}$.

4.

Consider the computable function

$$\begin{aligned} f(w) = &\quad w \quad \text{if } w \text{ is not a valid TM description} \\ = &\quad M' \quad \text{if } w = \langle M\rangle \end{aligned}$$

where M' is a machine identical to M except its reject state has been replaced by a state that loops infinitely.

This function shows that $ALL_{TM} \leq_m TOTAL_{TM}$ because if M accepts all input, it is clearly total and it never enters its reject state so that M' will behave identically to M and therefor $\langle M'\rangle \in TOTAL_{TM}$. On the other hand, if M rejects some input, M' will loop on that input so $\langle M'\rangle \notin TOTAL_{TM}$.

At the same time, f shows that $\overline{ALL_{TM}} \leq_m \overline{TOTAL_{TM}}$.

We know that neither ALL_{TM} nor $\overline{ALL_{TM}}$ is recognizable. Therefore, we can conclude that neither $TOTAL_{TM}$ nor $\overline{TOTAL_{TM}}$ is recognizable.