Name: Key **CS301 F9**

1. (__ /3 pts) Prove that the problem of determining whether a CFG generates any string in a given regular language is decidable. (See problem 4.14.) HW

   Let R be a regular language, and G be a CFG. Note that $R \cap L(G)$ is context-free by problem 2.18. The following TM effectively decides $R \cap L(G)$. On input $\langle G \rangle$:

   1.) Construct CFG H such that $L(H) = R \cap L(G)$

   2.) Test if $L(H) = \emptyset$ using decider for E_{CFG}, T.

   3.) If T accepts, reject; if T rejects, accept.

2. (__ /3 pts) Consider, from class, the proof that A_{TM} is recognizable, but not decidable. If D was defined to output the same value of H's output, would the proof still work? Why or why not?

   Assume D was defined in such a way...

   CASE: $D(\langle D \rangle)$ accepts — Implies $H(\langle D, \langle D \rangle \rangle)$ accepts, which means that $D(\langle D \rangle)$ accepts by def'n of H.

   NO CONTRADICTION!

   The case $D(\langle D \rangle)$ rejects works similarly.

3. (__ /3 pts) Consider, from homework, the proof of Theorem 4.22: *A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable.* Why is it important that M simulates M_1 and M_2 in *parallel*?

   M_1 and M_2 are not necessarily deciders for A and $\bar{A}$.

   It is possible, given input $w \in A \cup \bar{A}$, that one of M_1 or M_2 would not halt.

4. (__ /1 pts) Explain in "normal" programming terms what Corollary 4.23 means: A_{TM} *is not co-Turing-recognizable.*

   There is no algorithm that can look at any other algorithm M (together with its input w) and say definitively if M will not accept w.