There are three problems labeled A–C to be handed in. They all follow a theme: Sometimes we can decide a language, if we have some insight that lets us put a bound how much work needs to be done.

A. Problem 2.26, which is about Chomsky normal form. We used this result today in class, to bound the amount of work needed to derive a string from a CFG.

In class today, we showed that $EMPTY_{DFA}$ is decidable using breadth-first search. We also agreed (I hope) that we couldn't decide $EMPTY_{DFA}$ by simulating $M$ on all inputs $w$. But there is a version of that proof that works. Here's a sketch. The decider $D$, on input $\langle M \rangle$, does this:

1. Deduce a certain number $N$ by inspecting $M$.

2. Simulate $M$ on all inputs $w$ of length up to $N$.

3. If $M$ accepts any of those $w$s, then reject. Otherwise, accept.

B. In the decider above, what is the number $N$, and why does the decider then decide $EMPTY_{DFA}$ correctly? (Hint: Pumping lemma.)

C. Problem 5.15, which is about a Turing machine's never moving its head left. The language consists of strings $\langle M, w \rangle$, where $M$ and $w$ obey the relationship described in the problem.