Lectures and Reading. The major concept which is shortchanged by the text and other sources is that of a reduction. Often this concept is defined separately for recursive and then polynomial time, but we will take a broad view. The basic idea is that for many classes $C$ of languages (i.e., yes/no decision problems) there is a corresponding class $F$ of functions. For instance:

- If $C$ is the class $\text{REC}$ of decidable languages, then $F$ is the class of total computable functions.

- If $C$ is $P$, then $F$ is the class of polynomial-time computable functions.

- If $C$ is the class of regular languages, then $F$ is some class of functions computed by finite-state transducers (FSTs). (Two historical particular forms of FSTs are “Moore machines” and “Mealy machines,” but the best definition of FST generalizes both by allowing any fixed string to be emitted by any transition, allowing one final output string when the machine halts, and allowing the machine to reject and disavow any previous input it gave.)

Then an $F$-reduction from a language $A$ to a language $B$ is a function $f \in F$ such that for all $x \in \Sigma^*$:

$$x \in A \iff f(x) \in B.$$  

We then write $A \leq^F_m B$ and call the relation $\leq^F_m$ by the name $F$ many-one reducibility. For the relation to be meaningful we need it to be reflexive, transitive, and to give this theorem in regard to the corresponding class $C$:

- If $B \in C$ and $A \leq^F_m B$ then $A \in C$.

When $C$ is $\text{REC}$, we omit $F$ altogether in the notation. When $C$ is $P$ we just use a superscript $p$ in writing $A \leq^p_m B$, and similarly reg for regular reductions. Anyway, we get the theorem we want in all these cases. The impact is in the contrapositive, namely:

- If $A$ is undecidable and $A \leq_m B$ then $B$ is undecidable.

- If $A \notin P$ and $A \leq^p_m B$ then $B \notin P$.

- If $A$ is not regular and $A \leq^{\text{reg}}_m B$ then $B$ is not regular.

Tighter reductions like $\leq^{\text{reg}}_m$ or $\leq^p_m$ can be substituted for coarser ones like $\leq_m$ in these statements, but not vice-versa. That’s exactly why one wants to scale down $F$ accordingly with the level of complexity in $C$. The later impact will come when we add other classes such as $\text{NP}$ to the picture—then we will get statements like:

- If $A$ is $\text{NP}$-hard and $A \leq^p_m B$ then $B$ is $\text{NP}$-hard; if $B$ is also in $\text{NP}$ then it is $\text{NP}$-complete (and $A$ too).
Only the top-level reductions for (un)decidability will be featured before Prelim I, which is set for Wed. Oct. 19 in class period. This is the last assignment before it.

(1) Show that the concatenation of any two decidable languages is decidable, and that the concatenation of any two c.e. languages is c.e. Find, however, an example of two undecidable languages $A$ and $B$ such that $A \cdot B$ is decidable. Indeed, you can arrange that $A \cdot B$ is regular and even make $B = A$. (Vague hint: Take any language $L \subset \{0,1\}^*$ whatever and "dilate it" in such a way that you can fill in lots of extra strings into the "gaps" you made but the resulting language $A$ is still undecidable. 9+6+12 = 27 pts.)

(2) Prove—preferably by reduction—that the following decision problem is undecidable:

**Instance:** A Turing machine $M$.

**Question:** Is every string accepted by $M$ a palindrome?

State the language $L$ of this problem, then state the complementary language $\tilde{L}$ by a prose definition, and finally sketch why $\tilde{L}$—though likewise undecidable—is computably enumerable. (18+3+3+9 = 33 pts., for 60 total on the set)