Aughdon Breslin and Isabella Cruz
Problem Set 9
“I pledge my honor that I have abided by the Stevens Honor System”

Problem 1
Show that the following language is decidable by giving a high-level description of a TM that decides the language.

$$L_{inf} = \{ \langle M \rangle: \text{M is a PDA and L(M) is an infinite language}\}$$

A:
Let TM N decide L_{inf}.

Idea: if L(M) is infinite, N(<M>) accepts, else N(<M>) rejects.
Strategy: Need to decide if M accepts an iterable input w.

$L_{PDA} = \{\langle M, w \rangle$: M is a PDA that accepts input w$\}$

If there are p states in M, then we can find out if the language is infinite by checking if any w, where $w \in L(M)$, follows $|w| >= p$. If $|w| >= p$, by the Pigeonhole Principle, while reading input w, some state in M will have to be visited more than once, creating a loop.

This loop means that there will exist valid $w \in L(M)$ where the loop can be taken arbitrarily many times, creating an infinite language L(M). If there are no such inputs, then there will be a finite number of accepted inputs, and L(M) will be a finite language.

Let $L_{loop} = \{ w: w \in L(M) \text{ and } |w| >=p \}$

If L_{loop} contains any elements at all, that is L_{loop} does not equal Φ, then N(<M>) accepts and the language is infinite, else N(<M>) rejects and the language is finite.

Problem 2

Let G be a context-free grammar that generates strings over the alphabet $\Sigma = \{a, b\}$. Show that the problem of determining if G generates a string in $a*$ is decidable. In other words, show that the following language is decidable:

F = {<G>: G is a CFG over {a, b} and a* ∩ L(G) ≠ Φ}

A:

First, we will define a TM R to decide the language E where:

E = {<G>: G is a CFG and L(G) = Φ}

TM R to decide E:

On input <G>:

1. Mark every terminal symbol in G
2. Repeat until no new variable is marked:
   If A → a is a rule such that every variable in A is marked, then mark A
3. If the Start variable is not marked, ACCEPT; otherwise REJECT.

(TM from Lecture 25 slides)

Using TM R, we will define a TM Q that will decide the language F.

First, let's define a language A where A = a* ∩ L(G). Since A is a CFL, there is a CFG B that outlines its ruleset.

TM Q to decide F:

On input <B>:

1. Run R(<B>)
   If R(<B>) rejects, ACCEPT;
   otherwise REJECT

Explanation: TM R accepts a language if it is equal to Φ and rejects if it doesn't. To decide the language F, you must accept when a* ∩ L(G) ≠ Φ and reject otherwise. By running the CFG (B) of the CFL (A = a* ∩ L(G)) through TM R, we are able to learn if B is empty or not. Thus, if R rejects that means the language A is not empty meaning we would want TM Q to accept. If R accepts, that means A *is* empty, meaning we would want TM Q to reject.

Thus, since a decider can be created for the language F = {<G>: G is a CFG over {a, b} and a* ∩ L(G) ≠ Φ}, it is decidable.

Problem 3
Let A be a TM-recognizable language of strings that encode TMs that are deciders. Prove that there is a decidable language which is not decided by any TM in A. (Hint: start with an enumerator for A.)

Strat: Create a language that is defined by the following:
There must exist a word in L(D) so that one of the words $\notin M_1$, one of the words $\notin M_2$, and so on for each of the strings in A that encode decidable-TMs. (Diagonalization)

The decider: accepts if some word is not in some M, rejects otherwise, the language can be 'built' following this format.

A:
Since A is TM-recognizable, there must exist an enumerator E for A.
E will enumerate all decidable-TMs in the form of their binary encodings.

Let D be a TM that decides a language that isn't decided by any TM in A.
Rules of D:

On input w:

1. If w is not a combination of 0s and 1s, reject
2. Since w is a combination of 0s and 1s, let w be the nth possible combination in {0,1}*
3. Enumerate E up until the $\langle M_n \rangle$ output.
4. Run M_n on w, and accept if M_n rejects; reject if M_n accepts.
   (This would mean this input is a part of M_n's language and thus does not differ. We need a language where at least one input $\notin$ each of the other TM's languages to ensure distinctness.)

D is a decider since it will always accept if $w \in L(D)$, and reject if $w \notin L(D)$. The enumerator always stops after the nth output, where n is a finite number, and so stops in a finite amount of time. M_n is defined to be a decider, and so by definition will not run forever. Thus D will never run forever.

Therefore, the language of D, L(D), is a decidable language that differs from every TM in A by at least one element in its language. Therefore, there is a decidable language which is not decided by any TM in A.

Problem 4
Consider the problem of determining whether a TM M on input w ever attempts to move its head left when its head is on the leftmost tape cell.

a) Formulate this problem as a decision problem for a language, and

E_{LEFT} = { <M,w> The tapehead of TM M will attempt to move left on the leftmost tape cell of input w }

b) Show that the language is undecidable.

<u>Theorem:</u> E_{LEFT} is undecidable

<u>Proof:</u> We will show that $A_{TM} \leq E_{LEFT}$

Assume that E_{LEFT} is decidable and let R be a TM to decide it
We will use R to construct a TM S that decides A_{TM}:

S: On input <M,w>:
    1. Construct a TM C
    2.Run R(<C,w>)
        If R accepts, ACCEPT
        If R rejects, REJECT

C: On input x
    If x != w, REJECT
    Copy x into a second input tape w# such the leftmost cell is $ followed by x
    Run M on w# starting at the second leftmost cell (beginning of w)
        If the tape head ever points at the symbol $, move one cell to the right
        If M enters an accept state, move left until the tape head points to the $ symbol and ACCEPT
        Else REJECT

This, however, is a contradiction. A_{TM} is provably undecidable, so no decider can exist for it. This implies that our assumption that E_{LEFT} is decidable was false. Thus, E_{LEFT} must be undecidable.