Problem 1 (To Infinity And Beyond)

(*Note*: You only need to do **one** of problems 1 or 2 for the weekly.)

Let $EQ_{\mathsf{DFA}} = \{ \langle D_1, D_2 \rangle \mid D_1 \text{ and } D_2 \text{ are equivalent DFAs} \}$. Recall that two DFAs are equivalent if their languages are identical. Theorem 4.5 of Sipser shows that EQ_{DFA} is decidable by way of language closure properties. In this problem, we'll consider a more direct approach.

(a) Consider the following proof that EQ_{DFA} is decidable by way of constructing a deciding Turing machine that recognizes the language.

Proof. Define M, a deciding Turing machine that decides EQ_{DFA} as follows:

$M =$ "On input $\langle D_1, D_2 \rangle$—two DFAs:

(a) For every possible string $w \in \Sigma^*$, if D_1 and D_2 have differing acceptance behavior on w, *i.e.*, one rejects and the other accepts, then *reject*.

(b) Otherwise, D_1 and D_2 have the same acceptance behavior on all strings: *accept*". □

This construction has a fatal flaw! In a few sentences, describe what the flaw is.

(b) We can patch up this flaw by noting that we don't have to test all strings! Determine a bound on the size of strings we need to test to determine if two DFAs are equal. In a few sentences, argue why this bound is correct.

Problem 2 (Can You Do It? Yes You Can!)

(*Note*: You only need to do **one** of problems 1 or 2 for the weekly.)

(a) (*Optional*). Call a language L *prefix-free* if for all $w \in L$, there does not exist a $w' \in L$ such that $w \neq w'$ and w' is a prefix of w. For example ab is a prefix of $abcde$ and thus $abcde$ would not be in a prefix-free language if ab was also in the language. Give a deciding procedure to determine if the language of a DFA D is prefix free. Prove that it is correct by arguing both correctness and termination of your algorithm.

(b) Apply the construction you developed to the problem of determining whether a pushdown automata (PDA) is prefix-free. Identify:

- What you must change about the construction in order to adapt it to PDAs.
- What problems do you ultimately run into with your construction that you cannot reconcile?

(c) The previous part does not necessarily mean that the prefix-free property is not decidable for PDAs in general. Individually, spend approximately 10 minutes brainstorming an alternative approach to determining whether a PDA is prefix-free. Come back and determine which of your algorithms has the most promise. Describe the algorithm at a high-level and give a positive example of its execution, *i.e.*, a PDA whose language is prefix-free and how the algorithm accepts this PDA.

(d) It turns out that the problem of determining whether a PDA is prefix free is actually undecidable! Since we can't prove this fact yet, give an example of a PDA that your algorithm should accept or reject, but the algorithm either fails to give the correct answer or goes into an infinite loop.

Problem 3 (Countability)

(*Note*: This problem is **not required** for the weekly turn-in; we discussed it in class!)

(a) Prove that the language $L = \Sigma^*$ where $\Sigma = \{0, 1\}$ is countably infinite. (*Hint*: simply declaring that the mapping from Σ^* to $\mathbb{N}$ is the binary interpretation of $w \in \Sigma^*$ is insufficient since 0 and 000 are distinct strings but both represent the value zero. The resulting mapping would, therefore, not be a bijection!)

(b) Generalize your construction to any Σ of finite size.

(c) Use this fact to argue that the set of possible Java programs is countably infinite.

Problem 4 (Hey, Wait a Second)

Consider the following proof that $\mathbb{N}$ is uncountable.

Proof. Assume that $\mathbb{N}$ is countable. Then there is a bijection f that covers every natural numbers in $\mathbb{N}$. Construct the natural number n where the ith digit of n is the ith digit of the ith natural number in the bijection (*i.e.*, $f(i)$) plus one mod 10 (so that it is a decimal digit). That is, if k is the ith digit of the ith natural number, then the ith digit of n is given by $k + 1 \mod 10$. n is a valid natural number and by construction, n differs from every natural number in the bijection by one digit. Therefore, n cannot be in the bijection and therefore our assumption that such a bijection exists is incorrect. Thus, $\mathbb{N}$ is uncountable. □

However, we already know that $\mathbb{N}$ is countable (the bijection is the identity function). What is wrong with this proof? (*Hint*: think about the assumptions latent in Cantor's diagonalization argument that the reals are uncountable. What specifically is different in this case?)

Problem 5 (It Hurts My Head)

Prove that the following languages are undecidable by reduction from a known undecidable language.

(a) $L_1 = \{\langle M \rangle \mid M$ is a TM that accepts w^R whenever it accepts $w\}$. (Recall that w^R is the reversal of w.)

(b) $L_2 = \{\langle M, w \rangle \mid M$ is a TM that, on input w, writes a '$' on the tape$\}$.