1. (10 points) Show that a language is decidable iff some enumerator enumerates the language in lexicographic order. (Recall that the lexicographic order, puts shorts strings before longer strings and uses dictionary order for strings of the same length. For example, the lexicographic ordering of all strings over \{0, 1\} is \{ε, 0, 1, 00, 01, 10, 11, 000, \ldots\}.)

2. (10 points) Show that a language \(C\) is Turing-recognizable iff there exists a decidable language \(D\) such that \(C = \{x | \exists y (\langle x, y \rangle \in D)\}\)

3. (10 points) Define the language \(S = \{ \langle M \rangle | M\) is a DFA that accepts \(w^R\) whenever it accepts \(w\}\).

Prove that \(S\) is decidable.

4. (10 points) Prove that deterministic pushdown automata (DPDA) that have 2 stacks can simulate Turing machines. (The transition function of the 2-DPDA depends on the input symbol as well as the top symbols of the two stacks; the 2-DPDA can push/pop from both stacks and like a regular PDA, it has one-way read-only access to the input but the input is followed by a blank symbol as an end marker. Moreover, at each step precisely one transition of the DPDA is defined but, unlike a DFA, a 2-DPDA may make a transition from a state without reading any input symbol.)

5. (10 points) A (deterministic) queue automaton is like a (deterministic) pushdown automaton except that the stack is replaced by a queue. A queue is a tape allowing symbols to be written on the left end and read only at the right end. Each write operation (a push) adds a symbol to the left end of the queue and each read operation reads and removes the symbol at the right end of the queue. The input is given on a separate one-way read-only input tape with a blank end marker. Prove that deterministic queue automata are equivalent to Turing Machines.
6. (10 points) An unrestricted grammar (or a rewriting system) is a 4-tuple $G = (V, \Sigma, R, S)$ where

- $V$ is an alphabet;
- $\Sigma \subset V$ is the set of terminal symbols, and $V - \Sigma$ is called the set of nonterminal symbols;
- $S \in V - \Sigma$ is the start symbol; and
- $R$, the set of rules, is a finite subset of $(V^*(V - \Sigma)V^*) \times V^*$.

(Thus the “only” difference from context-free grammars is that the left-hand sides of rules need not consist of single nonterminals.) Let us write $\alpha \rightarrow \beta$ if $(\alpha, \beta) \in R$; and let’s define $u \Rightarrow_G v$ iff, for some $w_1, w_2 \in V^*$ and some rule $\alpha \rightarrow \beta \in R$, $u = w_1 \alpha w_2$ and $v = w_1 \beta w_2$. Let $\Rightarrow^*_G$ denote the reflexive, transitive closure of $\Rightarrow_G$. We say that a string $w \in \Sigma^*$ is generated by $G$ if and only if $S \Rightarrow^*_G w$. Finally, $L(G) \subseteq \Sigma^*$, the language generated by $G$, is defined to be the set of all strings in $\Sigma^*$ generated by $G$.

Your exercise is now to prove that a language is generated by an unrestricted grammar if and only if it is Turing-recognizable.

(Hint: Using nondeterminism and/or multiple tapes might aid in constructing Turing machines to simulate a grammar. For the other direction, to simulate a Turing Machine $M$ by a grammar, try to construct a grammar whose rules simulate backward moves of $M$, and whose derivations will consequently simulate backward computations of $M$.)