1. Show that a language $L$ is decidable iff there is some enumerator $E$ that prints the strings of $L$ in lexicographic order.

**Solution:** Assume $L$ is infinite, otherwise if $L$ is finite, $L$ is decidable and the equivalence is straightforward. Suppose $L$ is decidable and let $M$ be a Turing machine that decides $M$. Then one can construct an enumerator for $E$ by iterating through strings $s_i$ in lexicographic order, running $M$ on each string, and printing $s_i$ iff $M$ accepts $s_i$. Every string in the language $L$ will be listed at some point since $M$ terminates and by construction $E$ prints the strings in lexicographic order.

Suppose that there is enumerator $E$ that prints $L$ in lexicographic order. Then one can test whether or not any string $x$ is in the language $L$ or not in finite time by running $E$, accepting $x$ if $x$ is printed by $E$, or rejecting $x$ once a string greater than $x$ in lexicographic order is printed.

2. Let $A$ and $B$ be decidable languages. Show that the union $A \cup B$, the intersection $A \cap B$, the concatenation $AB = \{uv \mid u \in A, v \in B\}$, and the complement $\overline{A}$ are also decidable.

Which of the above properties remain true when decidable is replaced by semi-decidable?

**Solution:** Let $M_1$ be a decider for $A$ and $M_2$ be a decider for $B$. Let $M$ be a Turing machine that when given an input $x$ firstly runs $M_1$ on $x$ and then $M_2$ on $x$. If $M$ accepts iff both $M_1, M_2$ accepts, then $L(M) = A \cap B$. Otherwise if $M$ accepts iff either $M_1, M_2$ accept, then $L(M) = A \cup B$. A Turing machine for $AB$ can be designed by firstly guessing a decomposition of the input string $x$ into $x_1x_2$ and then checking if $x_1 \in A$ and $x_2 \in B$. Finally, by switching the accepting and rejecting states of $M_1$, we obtain a Turing machine $M'_1$ whose language is the complement $\overline{A}$.

When $A$ and $B$ are semi-decidable, the union, concatenation, and intersection are also semi-decidable. However, the construction above may not work for the union since it is possible for $x \in L_2$ but $M_1$ could loop on $x$. A more complicated construction that runs $M_1$ and $M_2$ in parallel is a Turing machine that recognizes the union $A \cup B$. Furthermore, if $A$ is semi-decidable, the complement may not be semi-decidable. An example of a semi-decidable language whose complement is not semi-decidable is the language $A_{TM}$.

3. Show that $A$ is semi-decidable if and only if there is a mapping reduction $A \leq_m A_{TM}$. Recall that $A_{TM}$ is the language

$$A_{TM} = \{ \langle M, w \rangle : M \text{ is a Turing machine that accepts } w \}.$$ 

This exercise, combined with the fact that $A_{TM}$ is semidecidable, shows that $A_{TM}$ is complete for the class of semi-decidable problems.
Solution: Suppose $A$ is semi-decidable. Then there is a Turing machine $M_A$ whose language $L(M_A) = A$. Then one can construct a reduction between $A$ and $A_{TM}$ by mapping strings $w \in \Sigma^*$ to the pair $\langle M_A, w \rangle$. By definition, $w \in A$ if and only if $M_A$ accepts $w$.

Conversely, suppose a reduction $A \leq_m A_{TM}$ exists. Let $f$ be the function that computes the reduction. Then $A$ is semi-decidable since the Turing machine that when given $x \in \Sigma^*$, runs the universal Turing machine $U$ on $f(x)$ and then accepts if $U$ accepts is a Turing machine whose language is $A$. So $A$ is semi-decidable.

4. Show that if $A$ is semi-decidable and there is a mapping reduction $A \leq_m \overline{A}$, then $A$ is decidable.

Solution: Since there is a reduction $A \leq_m \overline{A}$ then there is also a reduction $\overline{A} \leq_m A$ by taking complements of both sides. Hence $\overline{A}$ is semi-decidable since $A$ is semi-decidable. If $A$ and $\overline{A}$ are both semi-decidable, then $A$ is decidable.