<table>
<thead>
<tr>
<th>Problem Set 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Out:</strong> January 11 <strong>Due:</strong> January 18</td>
</tr>
</tbody>
</table>

Reminder: you are encouraged to work in groups of two or three; however you must turn in your own write-up and note with whom you worked. You may consult the course notes and the text (Sipser). The full honor code guidelines and collaboration policy can be found in the course syllabus.

Please attempt all problems. To facilitate grading, please turn in each problem on a separate sheet of paper and put your name on each sheet. Do not staple the separate sheets.

1. Let $f : \Sigma^* \rightarrow \Sigma^*$ be an arbitrary function. You are asked to (1) define a related language $L_f \subseteq \Gamma^*$, and (2) describe how a computer program could use a procedure that decides $L_f$ to compute $f$, and vice-versa. Here $\Gamma$ is an alphabet that may or may not be the same as $\Sigma$. Ideally, on input $x$, your program should invoke the procedure that decides $L_f$ at most polynomially many times in the length of $x$ and the length of $f(x)$, (but attaining this quantitative goal is not required to get full credit for this problem).

This justifies our use of languages (or decision problems) rather than the more general notion of function problems.

2. This problem is based on Problem 1.38 (Problem 1.31 in the First Edition), which defines a new kind of automaton: the all-paths-NFA. An all-paths-NFA is defined by a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ just like an NFA. The only difference is in the acceptance criterion. An all-paths-NFA accepts a string $x$ if every computation of $M$ on $x$ ends in an accept state (computations that “die” before reading to the end of the input are not counted). For comparison, recall that an NFA accepts a string $x$ if some computation of $M$ on $x$ ends in an accept state.

(a) Prove that $L$ is a regular language if and only if $L$ is recognized by an all-paths-NFA.

(b) Use part (a) to prove that the regular languages are closed under intersection. That is, prove that if $A$ and $B$ are regular languages, then

$$C = (A \cap B) = \{x : x \in A \text{ and } x \in B\}$$

is a regular language.

(c) Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA that recognizes language $L$. Let $M_{\text{flip}} = (Q, \Sigma, \delta, q_0, F')$ be the all-paths-NFA defined by taking $F' = Q - F$, and let $L_{\text{flip}}$ be the language recognized by $M_{\text{flip}}$. How are $L$ and $L_{\text{flip}}$ related? Briefly justify your answer.

3. A palindrome is a string that reads the same forwards as backwards (ignoring spaces); an example is “lonely tylenol”. Let $\Sigma$ be the 26 letter English alphabet. Prove that the language consisting of all palindromes is not regular.
4. A language over an alphabet $\Sigma$ with only one symbol is still meaningful. It is called a unary language. In this problem $\Sigma = \{0\}$.

(a) For a positive integer $n$, define the language $L_n \subseteq \Sigma^*$ to be the set of all strings whose length is not divisible by $n$. Prove that for all $n \geq 1$, $L_n$ is a regular language.

(b) Prove that the language $\text{primes} \subseteq \Sigma^*$, consisting of all strings whose length is a prime number, is not regular.

Spend a few minutes to convince yourself that this does not contradict Problem 2(b), in which you proved that the regular languages are closed under intersection.

5. For a positive integer $i$, let $N(i)$ denote its decimal representation (the usual string we write when writing the number $i$, with no leading zeros). Let $N'(i)$ denote the string $N(i)$ written in reverse order (least-significant digit first). Show that the language

$$L = \{ N(i) \# N'(i + 2) : i \geq 1 \}$$

over the alphabet $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \#\}$ is recognizable by a pushdown automata.