1. Give regular expressions describing the following languages:

(a) The set of strings of \( \{a, b\}^* \) starting with \( a \) and ending with \( a \).
\[ a(a + b)^*a + a. \]

(b) The set of strings of \( \{a, b\}^* \) containing at most two consecutive \( a \)’s.
\[ (\epsilon + a + aa)(b(\epsilon + a + aa))^*. \]

(c) The set of strings of \( \{a, b\}^* \) containing exactly two occurrences of \( ab \)’s.
\[ b^*a^*abb^*a^*abb^*a^*. \]

2. Given the alphabet \( \Sigma = \{0, 1, \ldots, 9\} \),

(a) create an automaton that accepts numbers in the range \( 0 – 999999 \).
See automaton of Figure 1.

(b) create a transducer that maps numbers (in the range \( 0 – 999999 \)) represented as strings of digits to their English read form, e.g.,
\[ 1 \rightarrow \text{one} \]
\[ 11 \rightarrow \text{eleven} \]
\[ 111 \rightarrow \text{one hundred eleven} \]
\[ 1111 \rightarrow \text{one thousand one hundred eleven} \]
\[ 11111 \rightarrow \text{eleven thousand one hundred eleven} \]
The transducer $T$ can be constructed using rational operations. Start with a digit transducer $D$ mapping single-digit numbers to their English expressions. Similarly, construct a transducer $T_1$ mapping numbers $11 - 19$ to their English expressions and $T_2$ mapping $10, 20, \ldots, 90$ to their English form, etc.

(c) Randomly generate several numbers both as strings of digits and in their read form. Use \texttt{fsmrandgen}.

(d) Create a weighted automaton over the real semiring associating to each sequence of digits its integer value (hint: here is some information about the automaton: it can be constructed with just two states and the initial state has a self-loop with weight one for each digit). You should prove that the automaton is correct. See automaton $A$ of figure 2.

\begin{center}
\begin{tikzpicture}
  \node[state, initial, accepting] (q0) at (0,0) {0};
  \node[state, accepting, right=2cm] (q1) at (2,0) {1/1};
  \node[state, right=2cm] (q2) at (4,0) {9/1};
  \node[state, above=2cm] (q3) at (2,2) {1/1};
  \node[state, above=2cm] (q4) at (4,2) {10/1};
  \node[state, right=2cm] (q5) at (6,0) {9/9};
  \node[state, below=2cm] (q6) at (2,-2) {0};
  \node[state, below=2cm] (q7) at (4,-2) {9/10};

  \path[->, below] (q0) edge node {1/1} (q1);
  \path[->, above] (q1) edge node {1/1} (q2);
  \path[->, above] (q2) edge node {1/1} (q0);
  \path[->, below] (q3) edge node {1/1} (q1);
  \path[->, above] (q4) edge node {1/1} (q2);
  \path[->, above] (q2) edge node {1/1} (q4);
  \path[->, below] (q5) edge node {1/1} (q3);
  \path[->, above] (q6) edge node {1/1} (q4);
  \path[->, above] (q4) edge node {1/1} (q6);
  \path[->, below] (q7) edge node {1/1} (q5);
  \path[->, above] (q5) edge node {1/1} (q7);
  \path[->, below] (q6) edge node {1/1} (q7);
  \path[->, above] (q7) edge node {1/1} (q6);

\end{tikzpicture}
\end{center}

Figure 2: Automaton accepting numbers in the range $0 - 999999$.

(e) Use the weighted automaton of the previous question and the transducer previously constructed to create a weighted automaton associating to an English sequence of the type "one hundred eleven" the number in the range $0 - 999999$ that it represents. Check the correctness of the weighted automaton by applying it to "eleven thousand two hundred fifteen".

proj$_3(A \circ T)$

3. Given the alphabet $\Sigma = \{a, b, \ldots, z, \langle space \rangle\},$

(a) create a transducer that implements the \textit{rot13} cipher $- a \rightarrow n, b \rightarrow o, \ldots, m \rightarrow z, n \rightarrow a, o \rightarrow b, \ldots, z \rightarrow m$, 

(b) encode the message "\textit{my secret message}" (assume $\langle space \rangle \rightarrow \langle space \rangle),
(c) decode the encoded message from above.

4. Construct a finite-state transducer that maps any string to the set of its substrings ($y \in \Sigma^*$ is a substring of $x \in \Sigma^*$ when there exists $u, v \in \Sigma^*$ such that $x = uyv$). Create a similar weighted transducer over the real semiring and explain how it could be used to count the number of occurrences of a sequence $x$ in a text $t$.

See transducer of Figure 3. Simply augment it with weights all equal to one, including the final weights, for the last question.

![Figure 3: Transducer mapping sequences to the set of their substrings.](image-url)