Name  Smarty Pantz
Subject  Theory of Computation
Instructor  Straubing
Section  Class
Date  October 23, '07  Book No.
1. \( L_1 \) consists of the strings over \( \{a, b\} \) whose first letter is different from the last letter.

- In: \( ab, baa \)
- Out: \( aba, baabb \)

- Here is the smallest possible NFA:

\[
\begin{align*}
1 & \xrightarrow{a} 2 \\
& \xrightarrow{b} 3 \\
3 & \xrightarrow{a} 4 \\
& \xrightarrow{b} 3 \\
2 & \xrightarrow{a} 4
\end{align*}
\]
Here is the result of applying the subset construction.

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>{2}</td>
<td>2</td>
<td>2,4</td>
</tr>
<tr>
<td>{3}</td>
<td>3,4</td>
<td>3</td>
</tr>
<tr>
<td>{4}</td>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>{2,4}</td>
<td>2</td>
<td>2,4</td>
</tr>
<tr>
<td>{3,4}</td>
<td>3,4</td>
<td>3</td>
</tr>
</tbody>
</table>

In the resulting DFA, we don't need to write down the unreachable states {4} and ∅.

```
0 → a 2 → b 2,4 b → a 0
    ↓     ↓     ↓
    b 3 → a 2,4
    ↓     ↓     ↓
    a 3,4 → b
```
(2) Here is a DFA that recognizes $L_2$

$$\begin{align*}
\emptyset & \xrightarrow{b,c} a & a & \xrightarrow{c} a, b, c \\
\emptyset & \xrightarrow{b} & \emptyset & \xrightarrow{c} \emptyset
\end{align*}$$

To compute the regular expression, you really don't need the rightmost state, so we work with

$$\begin{align*}
\emptyset & \xrightarrow{b,c} a & a & \xrightarrow{c} a \\
\emptyset & \xrightarrow{a} & \emptyset & \xrightarrow{c} \emptyset
\end{align*}$$

Eliminate state 2:

$$\begin{align*}
\emptyset & \xrightarrow{a} \emptyset & \emptyset & \xrightarrow{b} \emptyset \\
\emptyset & \xrightarrow{a} \emptyset & \emptyset & \xrightarrow{c} \emptyset
\end{align*}$$

Result:

$$\begin{align*}
\emptyset & \xrightarrow{a} \emptyset & \emptyset & \xrightarrow{a} \emptyset
\end{align*}$$

\[(bucu aa^*b)a^*\]
3. Because

\[ K(L^*) = \left[ K(L) \right]^* \]

\[ K(L_1 \cup L_2) = K(L_1) \cup K(L_2) \]

\[ K(L_1 L_2) = K(L_1) K(L_2) \]

for all languages \( L, L_1, L_2 \), we get a regular expression for \( K(L) \) simply by taking a regular expression for \( L \) and replacing each \( b \) by \( ab \). Thus if \( L \) is regular, so is \( K(L) \).

Example. \( L \) generated by \((a \cup b)a)^* b \)

\( K(L) \) generated by \((a \cup b)a)^* ab \).
@ True. The set described is
\[ L' = L \cap \{a^nb^n b \} \]

Since the family of regular languages is closed under intersection, if \( L \) is regular, then so is \( L' \).

@ False. As a very simple example,

let \( L_1 = \{ a^n b^n : n > 1 \} \)

\( L_2 = \{ b^n a^n : n > 1 \} \)

Both are non-regular, but

\( L_1 \cap L_2 = \emptyset \) is regular.
(0) True.

If $L$ is non-regular, then

$$L = \overline{\overline{L}}$$

is regular, because the family of regular languages is closed under complementation.

(0) False: the problem is that simply reversing the arrows introduces nondeterminism. If

$$L = (a \cup b)^* (a \cup b) \cup (a \cup b)^*$$

$$L = (a \cup b) (a \cup b) a (a \cup b)^*$$

then

$L$ is recognized by the DFA
\[ \rightarrow 0 \xrightarrow{a,b} 0 \xrightarrow{a,b} 0 \xrightarrow{a} 0 \xrightarrow{b} 0 \xrightarrow{2a,b} \]

with 5 states, but the reversal requires a DFA with 8 states.

(proved in a homework problem)