This is a "core dump" of potential questions for Midterm 1. This should give you a good idea of the *types* of questions that we will ask on the exam—in particular, there *will* be a series of True/False questions—but the actual exam questions may or may not appear in this handout. This list intentionally includes a few questions that are too long or difficult for exam conditions.

Questions from Spring 2014 exams are labeled ⟨⟨*S14*⟩⟩. Questions from homework are labeled ⟨⟨*HW*⟩⟩. Questions from labs are labeled ⟨⟨*Lab*⟩⟩.

---

1. **Induction on strings.** Give complete, formal inductive proofs for the following claims. Your proofs must reply on the formal recursive definitions of the relevant string functions, not on intuition. Recall that the concatenation $\bullet$ and length $|\cdot|$ functions are formally defined as follows:

$$w \bullet y := \begin{cases} y & \text{if } w = \varepsilon \\ a \cdot (x \bullet y) & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

$$|w| := \begin{cases} 0 & \text{if } w = \varepsilon \\ 1 + |x| & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

- For any string w and any non-negative integer n, let $\boldsymbol{w^n}$ denote the string obtained by concatenating n copies of w; more formally, define

$$w^n := \begin{cases} \varepsilon & \text{if } n = 0 \\ w \bullet w^{n-1} & \text{otherwise} \end{cases}$$

For example, $(\texttt{BLAH})^5 = \texttt{BLAHBLAHBLAHBLAHBLAH}$ and $\varepsilon^{374} = \varepsilon$.

(a) Prove that $w^m \bullet w^n = w^{m+n}$ for every string w and all non-negative integers n and m.

(b) Prove that $(w^m)^n = w^{mn}$ for every string w and all non-negative integers n and m.

(c) Prove that $|w^n| = n|w|$ for every string w and every integer $n \geq 0$.

- The ***reversal*** $\boldsymbol{w^R}$ of a string w is defined recursively as follows:

$$w^R := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ x^R \bullet a & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

(a) Prove that $(w \bullet x)^R = x^R \bullet w^R$ for all strings w and x. ⟨⟨*lab*⟩⟩

(b) Prove that $(w^R)^R = w$ for every string w. ⟨⟨*lab*⟩⟩

(c) Prove that $|w| = |w^R|$ for every string w.

(d) Prove that $(w^n)^R = (w^R)^n$ for every string w and every integer $n \geq 0$.

- Consider the following pair of mutually recursive functions:

$$\mathit{evens}(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \mathit{odds}(x) & \text{if } w = ax \end{cases} \qquad \mathit{odds}(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a \cdot \mathit{evens}(x) & \text{if } w = ax \end{cases}$$

For example, $\mathit{evens}(\texttt{0001101}) = \texttt{010}$ and $\mathit{odds}(\texttt{0001101}) = \texttt{0011}$.

(a) Prove the following identity for all strings w and x:

$$evens(w \bullet x) = \begin{cases} evens(w) \bullet evens(x) & \text{if } |w| \text{ is even,} \\ evens(w) \bullet odds(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

(b) Prove the following identity for all strings w:

$$evens(w^R) = \begin{cases} (evens(w))^R & \text{if } |w| \text{ is odd,} \\ (odds(w))^R & \text{if } |w| \text{ is even.} \end{cases}$$

(c) Prove that $|w| = |evens(w)| + |odds(w)|$ for every string w.

- Consider the following recursive function:

$$scramble(w) := \begin{cases} w & \text{if } |w| \le 1 \\ ba \bullet scramble(x) & \text{if } w = abx \text{ for some } a, b \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

For example, $scramble(\texttt{0001101}) = \texttt{0010011}$.

(a) Prove that $|scramble(w)| = |w|$ for every string w.

(b) Prove that $scramble(scramble(w)) = w$ for every string w.

2. **Regular expressions.** For each of the following languages over the alphabet $\{0, 1\}$, give an equivalent regular expression.

   - Every string of length at most 3. *[Hint: Don't try to be clever.]*
   - Every string except 010. *[Hint: Don't try to be clever.]*
   - All strings in which every run of consecutive 0s has even length and every run of consecutive 1s has odd length.
   - All strings *not* containing the substring 010.
   - All strings containing at least two 1s and at least one 0.
   - All strings containing *either* at least two 1s *or* at least one 0.
   - All strings such that *in every prefix*, the number of 0s and the number of 1s differ by at most 1.
   - The set of all strings in $\{0, 1\}^*$ whose length is divisible by 3.
   - ⟨⟨*S14*⟩⟩ The set of all strings in 0^*1^* whose length is divisible by 3.
   - The set of all strings in $\{0, 1\}^*$ in which the number of 1s is divisible by 3.

3. **Direct DFA construction.** Draw or formally describe a DFA that recognizes each of the following languages. If you draw the DFA you may omit transitions to "dump" states.

   - Every string of length at most 3.
   - Every string except 010.
   - The language {LONG, LUG, LEGO, LEG, LUG, LOG, LINGO}.
   - The language $\mathtt{MOO}^* + \mathtt{MEOO}^*\mathtt{W}$
   - All strings in which every run of consecutive 0s has even length and every run of consecutive 1s has odd length.
   - All strings *not* containing the substring 010.
   - The set of all strings in $\{0, 1\}^*$ whose length is divisible by 3.
   - ⟨⟨*S14*⟩⟩ The set of all strings in 0^*1^* whose length is divisible by 3.
   - The set of all strings in $\{0, 1\}^*$ in which the number of 1s is divisible by 3.
   - All strings w such that the binary value of w^R is divisible by 5.
   - All strings such that *in every prefix*, the number of 0s and the number of 1s differ by at most 2.

4. **Fooling sets.** ***Prove*** that each of the following languages is *not* regular.

   - The set of all strings in $\{0, 1\}^*$ with more 0s than 1s. ⟨⟨*S14*⟩⟩
   - The set of all strings in $\{0, 1\}^*$ with fewer 0s than 1s.
   - The set of all strings in $\{0, 1\}^*$ with exactly twice as many 0s as 1s.
   - The set of all strings in $\{0, 1\}^*$ with at least twice as many 0s as 1s.
   - $\{0^{n^3} \mid n \geq 0\}$
   - $\{0^{2^n} \mid n \geq 0\}$ ⟨⟨*Lab*⟩⟩
   - $\{0^{F_n} \mid n \geq 0\}$, where F_n is the nth Fibonacci number, defined recursively as follows:

$$F_n := \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F_{n-1} + F_{n-2} & \text{otherwise} \end{cases}$$

   - $\{x\#y \mid x, y \in \{0, 1\}^* \text{ and } \#(0, x) = \#(1, y)\}$
   - $\{xy \mid x \in \{0, 1\}^* \text{ and } y = \mathit{flip}(x)\}$, where $\mathit{flip}(w)$ is the string obtained from w by flipping every bit. For example, $\mathit{flip}(0001101) = 1110010$.
   - The language of properly balanced strings of parentheses, described by the context-free grammar $S \to \varepsilon \mid SS \mid (S)$.
   - $\{(01)^n(10)^n \mid n \geq 0\}$
   - $\{(01)^m(10)^n \mid n \geq m \geq 0\}$
   - $\{w\#x\#y \mid w, x, y \in \Sigma^* \text{ and } w, x, y \text{ are not all equal}\}$

5. **Regular or not?** For each of the following languages, either prove that the language is regular (by describing a DFA, NFA, or regular expression), or prove that the language is not regular (using a fooling set argument).

   - The set of all strings in $\{0, 1\}^*$ in which the substrings 01 and 10 appear the same number of times. (For example, the substrings 01 and 01 each appear three times in the string 1100001101101.)
   - The set of all strings in $\{0, 1\}^*$ in which the substrings 00 and 11 appear the same number of times. (For example, the substrings 00 and 11 each appear three times in the string 1100001101101.)
   - The set of all strings in $\{0, 1, (,), *, +, \emptyset, \varepsilon\}^*$ that are regular expressions over the alphabet $\{0, 1\}$.
   - The set of all strings in $\{0, 1\}^*$ such that in every prefix, the number of 0s is greater than the number of 1s.
   - The set of all strings in $\{0, 1\}^*$ such that in every *non-empty* prefix, the number of 0s is greater than the number of 1s.
   - The language generated by the following context-free grammar:

   $$S \to 0A1 \mid \varepsilon$$
   $$A \to 1S0 \mid \varepsilon$$

   - The language generated by the following context-free grammar:

   $$S \to 0A1$$
   $$A \to 1S0 \mid \varepsilon$$

   - The language generated by the following context-free grammar:

   $$S \to 0S1 \mid 1S0 \mid \varepsilon$$

   - $\{w\#x \mid w, x \in \{0, 1\}^* \text{ and no substring of } w \text{ is also a substring of } x\}$
   - $\{w\#x \mid w, x \in \{0, 1\}^* \text{ and no \textit{non-empty} substring of } w \text{ is also a substring of } x\}$
   - $\{w\#x \mid w, x \in \{0, 1\}^* \text{ and \textit{every} non-empty substring of } w \text{ is also a substring of } x\}$
   - $\{w\#x \mid w, x \in \{0, 1\}^* \text{ and } w \text{ is a substring of } x\}$
   - $\{w\#x \mid w, x \in \{0, 1\}^* \text{ and } w \text{ is a proper substring of } x\}$
   - $\{xy \mid \#(0, x) = \#(1, y) \text{ and } \#(1, x) = \#(0, y)\}$
   - $\{xy \mid \#(0, x) = \#(1, y) \text{ or } \#(1, x) = \#(0, y)\}$
   - $\{x\#y \mid x, y \in \{0, 1\}^* \quad \text{and} \quad (\#(0, x) = \#(1, y) \text{ or } \#(1, x) = \#(0, y))\}$

6. **Product/subset constructions.** For each of the following languages $L \subseteq \{0,1\}^*$, formally describe a DFA $M = (Q, \{0,1\}, s, A, \delta)$ that recognizes L. ***Do not attempt to draw the DFA.*** Instead, give a complete, precise, and self-contained description of each of the components Q, s, a, and δ. (Don't just describe several smaller DFAs and then say "product construction!")

   - ⟨⟨*S14*⟩⟩ All strings that satisfy *all* of the following conditions:
     - the number of 0s is even
     - the number of 1s is divisible by 3
     - the total length is divisible by 5
   - All strings that satisfy *at least one* of the following conditions: ...
   - All strings that satisfy *exactly one* of the following conditions: ...
   - All strings that satisfy *exactly two* of the following conditions: ...
   - All strings that satisfy *an odd number of* of the following conditions: ...

   - Other possible conditions:
     - The number of 0s in w is odd.
     - The number of 1s in w is not divisible by 5.
     - The length $|w|$ is divisible by 7.
     - The binary value of w is divisible by 7.
     - The binary value of w^R is not divisible by 7.
     - w contains the substring 00
     - w does not contain the substring 11
     - ww does not contain the substring 101

7. **NFA construction.** Let L be an arbitrary regular language $\Sigma = \{\mathtt{0}, \mathtt{1}\}$. Prove that each of the following languages over $\{\mathtt{0}, \mathtt{1}\}$ is regular. "Describe" does not necessarily mean "draw".

   - All strings that satisfy *at least one* of the following conditions:
     - the number of 0s is even
     - the number of 1s is divisible by 3
     - the total length is divisible by 5.
   - All strings such that *in every prefix*, the number of 0s and the number of 1s differ by at most 2.
   - All strings such that *in every substring*, the number of 0s and the number of 1s differ by at most 2.
   - $\text{PREFIXES}(L) := \{x \mid xy \in L \text{ for some string } x \in \Sigma^*\}$
   - $\text{SUFFIXES}(L) := \{y \mid xy \in L \text{ for some string } y \in \Sigma^*\}$
   - $\text{ONEINFRONT}(L) := \{\mathtt{1}x \mid x \in L\}$
   - $\text{MISSINGFIRST}(L) := \{w \in \Sigma^* \mid aw \in L \text{ for some symbol } a \in \Sigma\}$
   - $\text{EVENS}(L) := \{evens(w) \mid w \in L\}$, where the functions *evens* and *odds* are recursively defined as follows:

     $$evens(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ odds(x) & \text{if } w = ax \end{cases} \qquad odds(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a \cdot evens(x) & \text{if } w = ax \end{cases}$$

     For example, $evens(\mathtt{0001101}) = \mathtt{010}$ and $odds(\mathtt{0001101}) = \mathtt{0011}$.
   - $\text{EVENS}^{-1}(L) := \{w \mid evens(w) \in L\}$, where the functions *evens* and *odds* are recursively defined as above.
   - $\text{FLIP}(L) := \big\{flip(w) \bigm| w \in L\big\}$, where the function *flip* is defined recursively as follows:

     $$flip(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \mathtt{1} \cdot flip(x) & \text{if } w = \mathtt{0}x \text{ for some string } x \\ \mathtt{0} \cdot flip(x) & \text{if } w = \mathtt{1}x \text{ for some string } x \end{cases}$$

     For example, $flip(\mathtt{0001101}) = \mathtt{1110010}$.
   - $\text{SHUFFLE}(L) := \big\{shuffle(w, x) \bigm| w, x \in L\big\}$, where the function *shuffle* is defined recursively as follows:

     $$shuffle(w, x) := \begin{cases} x & \text{if } w = \varepsilon \\ a \cdot shuffle(x, y) & \text{if } w = ay \text{ for some } a \in \Sigma \text{ and some } y \in \Sigma^* \end{cases}$$

     For example, $shuffle(\mathtt{0001101}, \mathtt{11111}) = \mathtt{0}^{\mathtt{1}}\mathtt{0}^{\mathtt{1}}\mathtt{0}^{\mathtt{1}}\mathtt{1}^{\mathtt{1}}\mathtt{1}^{\mathtt{1}}\mathtt{01}$.
   - $\text{SCRAMBLE}(L) := \big\{scramble(w) \bigm| w \in L\big\}$, where the function *scramble* is defined recursively as follows:

     $$scramble(w) := \begin{cases} w & \text{if } |w| \le 1 \\ ba \cdot scramble(x) & \text{if } w = abx \text{ for some } a, b \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

     For example, $scramble(\mathtt{00\,01\,10\,1}) = \mathtt{00\,10\,01\,1}$.

- $\textsc{Stutter}(L) := \left\{ stutter(w) \;\middle|\; w \in L \right\}$, where the function *stutter* is defined recursively as follows:

$$stutter(w) := \begin{cases} e & \text{if } w = \varepsilon \\ aa \cdot stutter(x) & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

  For example, $stutter(\texttt{00101}) = \texttt{0000110011}$.

- $\textsc{Stutter}^{-1}(L) := \left\{ w \;\middle|\; stutter(w) \in L \right\}$, where the function *stutter* is defined recursively as above.

- $\textsc{Stuttered}(L) := \left\{ w \in L \;\middle|\; w = stutter(x) \text{ for some } x \in \Sigma^* \right\}$, where the function *stutter* is defined recursively as above.

8. **True or False (sanity check).** For each statement below, check "True" if the statement is ***always*** true and "False" otherwise. Each correct answer is worth 1 point; each incorrect answer is worth -½ point; checking "I don't know" is worth ¼ point; and flipping a coin is (on average) worth ¼ point.

   Read each statement *very* carefully. Some of these are deliberately subtle.

Definitions

- For all languages L, if L is regular then L can be represented by a regular expression.
- For all languages L, if L is not regular then L cannot be represented by a regular expression.
- For all languages L, if L can be represented by a regular expression then L is regular.
- For all languages L, if L cannot be represented by a regular expression then L is not regular.
- For all languages L, if there is a DFA that accepts every string in L, then L is regular.
- For all languages L, if there is a DFA that accepts every string not in L, then L is not regular.
- For all languages L, if there is a DFA that rejects every string not in L, then L is regular.
- ⟨⟨*S14*⟩⟩ For all languages L, if for every string $w \in L$ there is a DFA that accepts w, then L is regular.
- For all languages L, if for every string $w \notin L$ there is a DFA that rejects w, then L is regular.
- For all languages L, if some DFA recognizes L, then some NFA also accepts L.
- For all languages L, if some NFA recognizes L, then some DFA also accepts L.
- For all languages $L \subseteq \Sigma^*$, if L cannot be described by a regular expression, then some DFA accepts $\Sigma^* \setminus L$.

Closure Properties

- For all regular languages L and L', the language $L \cap L'$ is regular.
- For all regular languages L and L', the language $L \cup L'$ is regular.
- For all regular languages L, the language L^* is regular.
- For all regular languages A, B, and C, the language $(A \cup B) \setminus C$ is regular.

- For all languages $L \subseteq \Sigma^*$, if L is regular, then $\Sigma^* \setminus L$ is regular.
- For all languages $L \subseteq \Sigma^*$, if L is regular, then $\Sigma^* \setminus L$ is not regular.
- For all languages $L \subseteq \Sigma^*$, if L is not regular, then $\Sigma^* \setminus L$ is regular.
- For all languages $L \subseteq \Sigma^*$, if L is not regular, then $\Sigma^* \setminus L$ is not regular.

- ⟨⟨*S14*⟩⟩ For all languages L and L', the language $L \cap L'$ is regular.
- For all languages L and L', the language $L \cup L'$ is regular.
- For all languages L, the language L^* is regular.
- For all languages L, if L^* is regular, then L is regular.
- For all languages A, B, and C, the language $(A \cup B) \setminus C$ is regular.

- For all languages L, if L is finite, then L is regular.
- For all languages L and L', if L and L' are finite, then $L \cup L'$ is regular.

- For all languages L and L', if L and L' are finite, then $L \cap L'$ is regular.
- For all languages L, if L contains a finite number of strings, then L is regular.
- For all languages $L \subseteq \Sigma^*$, if L contains infinitely many strings in Σ^*, then L is not regular.
- ⟨⟨*S14*⟩⟩ For all languages $L \subseteq \Sigma^*$, if L contains all but a finite number of strings of Σ^*, then L is regular.
- For all languages $L \subseteq \{0, 1\}^*$, if L contains a finite number of strings in 0^*, then L is regular.
- For all languages $L \subseteq \{0, 1\}^*$, if L contains a all but a finite number of strings in 0^*, then L is regular.

- If L and L' are not regular, then $L \cap L'$ is not regular.
- If L and L' are not regular, then $L \cup L'$ is not regular.
- ⟨⟨*S14*⟩⟩ If L is regular and $L \cup L'$ is regular, then L' is regular.
- ⟨⟨*S14*⟩⟩ If L is regular and $L \cup L'$ is not regular, then L' is not regular.
- If L is not regular and $L \cup L'$ is regular, then L' is regular.
- If L is regular and $L \cap L'$ is regular, then L' is regular.
- If L is regular and $L \cap L'$ is not regular, then L' is not regular.
- ⟨⟨*S14*⟩⟩ If L is regular and L' is finite, then $L \cup L'$ is regular.
- If L is regular and L' is finite, then $L \cap L'$ is regular.
- If $L \subseteq L'$ and L is regular, then L' is regular.
- If $L \subseteq L'$ and L' is regular, then L is regular.
- If $L \subseteq L'$ and L is not regular, then L' is not regular.
- If $L \subseteq L'$ and L' is not regular, then L is not regular.

Equivalence Classes

- For all languages L, if L is regular, then $\equiv_L$ has finitely many equivalence classes.
- ⟨⟨*S14*⟩⟩ For all languages L, if L is not regular, then $\equiv_L$ has infinitely many equivalence classes.
- For all languages L, if $\equiv_L$ has finitely many equivalence classes, then L is regular.
- For all languages L, if $\equiv_L$ has infinitely many equivalence classes, then L is not regular.
- For all regular languages L, each equivalence class of $\equiv_L$ is a regular language.
- For all languages L, each equivalence class of $\equiv_L$ is a regular language.

Fooling Sets

- For all languages L, if L has an infinite fooling set, then L is not regular.
- For all languages L, if L has an finite fooling set, then L is regular.
- For all languages L, if L does not have an infinite fooling set, then L is regular.
- For all languages L, if L is not regular, then L has an infinite fooling set.
- For all languages L, if L is regular, then L has no infinite fooling set.
- For all languages L, if L is not regular, then L has no finite fooling set.

Specific Languages (gut check)

- ⟨⟨S14⟩⟩ $\{0^i1^j2^k \mid i+j-k=374\}$ is regular.
- $\{0^i1^j2^k \mid i+j-k \leq 374\}$ is regular.
- $\{0^i1^j2^k \mid i+j+k=374\}$ is regular.
- $\{0^i1^j2^k \mid i+j+k>374\}$ is regular.
- ⟨⟨S14⟩⟩ $\{0^i1^j \mid i<374<j\}$ is regular.
- $\left\{0^i1^j \;\middle|\; |i-j|<374\right\}$ is regular.
- ⟨⟨S14⟩⟩ $\{0^i1^j \mid (i-j)$ is divisible by $374\}$ is regular.
- $\{0^i1^j \mid (i+j)$ is divisible by $374\}$ is regular.

- $\left\{0^{n^2} \;\middle|\; n \geq 0\right\}$ is regular.
- $\left\{0^{37n+4} \;\middle|\; n \geq 0\right\}$ is regular.
- $\left\{0^n10^n \;\middle|\; n \geq 0\right\}$ is regular.
- $\left\{0^m10^n \;\middle|\; m \geq 0 \text{ and } n \geq 0\right\}$ is regular.

- $\{w \in \{0,1\}^* \mid |w|$ is divisible by $374\}$ is regular.
- $\{w \in \{0,1\}^* \mid w$ represents a integer divisible by 374 in binary$\}$ is regular.
- $\{w \in \{0,1\}^* \mid w$ represents a integer divisible by 374 in base 473$\}$ is regular.

- $\left\{w \in \{0,1\}^* \;\middle|\; |\#(0,w)-\#(1,w)|<374\right\}$ is regular.
- $\left\{w \in \{0,1\}^* \;\middle|\; |\#(0,x)-\#(1,x)|<374 \text{ for every prefix } x \text{ of } w\right\}$ is regular.
- $\left\{w \in \{0,1\}^* \;\middle|\; |\#(0,x)-\#(1,x)|<374 \text{ for every substring } x \text{ of } w\right\}$ is regular.

- $\left\{w0^{\#(0,w)} \;\middle|\; w \in \{0,1\}^*\right\}$ is regular.
- $\left\{w0^{\#(0,w) \bmod 374} \;\middle|\; w \in \{0,1\}^*\right\}$ is regular.

Automata Transformations

- Let M be a DFA over the alphabet Σ. Let M' be identical to M, except that accepting states in M are non-accepting in M' and vice versa. Each string in Σ^* is accepted by exactly one of M and M'.
- Let M be an NFA over the alphabet Σ. Let M' be identical to M, except that accepting states in M are non-accepting in M' and vice versa. Each string in Σ^* is accepted by exactly one of M and M'.
- If a language L is recognized by a DFA with n states, then the complementary language $\Sigma^* \setminus L$ is recognized by a DFA with at most $n+1$ states.
- If a language L is recognized by an NFA with n states, then the complementary language $\Sigma^* \setminus L$ is recognized by a NFA with at most $n+1$ states.
- If a language L is recognized by a DFA with n states, then L^* is recognized by a DFA with at most $n+1$ states.
- If a language L is recognized by an NFA with n states, then L^* is also recognized by a NFA with at most $n+1$ states.

Language Transformations

- For every regular language L, the language $L^R = \{w^R \mid w \in L\}$ is also regular. ⟨⟨*HW*⟩⟩
- For every language L, if the language $L^R = \{w^R \mid w \in L\}$ is regular, then L is also regular.
- For every regular language L, the language $\{\texttt{0}^{|w|} \mid w \in L\}$ is also regular.
- For every language L, if the language $\{\texttt{0}^{|w|} \mid w \in L\}$ is regular, then L is also regular.