

The Pumping Lemma says that *any “sufficiently long” string in a regular language L can be broken into three parts such that if we “pump” the middle part (repeat it zero or more times) then the result would still belong to L .*

Pumping Lemma: Let L be a regular language. Then there exists a constant p such that for every string w in L , with $|w| \geq p$, we can break w into three parts $w = xyz$ such that

- (1) $y \neq \varepsilon$ (i.e. $|y| > 0$ or $|y| \neq 0$)
- (2) $|xy| \leq p$ (xy cannot occupy more than the first p symbols of w)
- (3) For all $k \geq 0$, the string xy^kz is also in L (i.e. $xy^*z \in L$)

The Pumping Lemma when used to prove that a language L is **not regular** can be viewed as a “game” between a **Prover** and a **Falsifier** as follows:

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** writes $w = xyz$ such that $|xy| \leq p$ and $y \neq \varepsilon$.

② **Falsifier** challenges **Prover** and picks a string $w \in L$ of length at least p symbols.

Often, we pick w to be “at the edge” of membership, i.e. as close as possible to failing to be a yes-instance.

④ **Falsifier** wins by finding a value for k such that xy^kz is **not** in L . If it cannot then it fails and **Prover** wins.

The language L is not regular if **Falsifier** can always win this game systematically.

The following are almost complete proofs using the Pumping Lemma (PL). Complete them by filling in the hidden details.

- (1) Show that the language $L = \{a^n b^n \mid n \geq 0\}$ is not regular.

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** tries to decompose w into three parts $w = \square$ but sees that the condition $|xy| \leq \square$ forces x and y to only contain the symbol \square . Furthermore, y cannot just be the empty string because of the condition \square . Seeing this, the only option available is to have $xy = a^m$ for some $m \geq 1$, and then we get $z = a^{p-m} b^p$.

② **Falsifier** challenges **Prover** and picks $w = a^p \square \in L$ ($|w| = \square \geq p$).

④ **Falsifier** now sees that $xy^0z, xy^2z, xy^3z, \dots$ all do not belong to L because they either have less or more \square 's than there are \square 's. So, any such string will be enough for **Falsifier** to win the game.

(2) $L = \{ww \mid w \in \{0,1\}^*\}$.

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** The PL now guarantees that w can be split into three substrings $w = xyz$ satisfying $|xy| \leq p$ and $y \neq \varepsilon$.

② **Falsifier** challenges **Prover** and chooses $w = (0^p1)(0^p1) \in L$.

This has length

$$|w| = (p+1) + (\boxed{}) = \boxed{} \geq p.$$

④ **Falsifier** Since

$$w = (0^p1)(\boxed{}) = xyz$$

with $|xy| \leq p$ then we must have that y only contains the symbol $\boxed{}$.

We can then pump y and produce $xy^2z = xyz \notin L$ because the first half $\boxed{}$ the second half.

So L is not regular.

(3) $L = \{a^i b^j c^k \mid 0 \leq i < j < k\}$

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** writes

$$w = (xy)z = (a^p)b^{p+1}c^{p+2}$$

where xy is a string of $\boxed{}$'s only

② **Falsifier** challenges **Prover** and chooses

$$w = a^{\boxed{}}b^{\boxed{}+1}c^{\boxed{}+2}.$$

Here $|w| = p + \boxed{} \boxed{} p$

④ **Falsifier** forms

$$xy^2z = a^{p+\boxed{}}b^{p+1}c^{p+2} \notin L$$

because $|y| \geq 1$.

(4) $L = \{a^i b^j \mid i > j\}$

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** writes

$$w = (xy)z = (a^{\boxed{}})ab^{\boxed{}}$$

i.e. xy is a string of $\boxed{}$'s only

② **Falsifier** challenges **Prover** and chooses

$$w = a^{\boxed{}+1}b^{\boxed{}}$$

Here $|w| = \boxed{} = 2p + 1 - \boxed{}p$

④ **Falsifier** forms

$$xy^0z = xz = a^{p+1-\boxed{}}b^p \notin L$$

because $|y| \geq 1$. (so $p + 1 - \boxed{} \leq p$).

(5) $L = \{a^i b^j c^k \mid i > j > k \geq 0\}$

① **Prover** claims L is regular and fixes the value of the pumping length p .

③ **Prover** writes

$$w = a^{\boxed{}}a^2b^{p+1}c^0 = xyz,$$

where xy can have a maximum of $\boxed{}$ symbols, so xy must be a string of $\boxed{}$'s only

② **Falsifier** challenges **Prover** and chooses

$$w = a^{\boxed{}}b^{p+1}c^0.$$

Here $|w| = \boxed{} + (p + 1) + 0 - \boxed{}\boxed{}.$

④ **Falsifier** forms

$$xy^{\boxed{}}z = xz = a^{\boxed{}-|y|}b^{p+1}c^0 \notin L$$

because $|y| \geq 1$.

- (6) **(Minimum pumping length)** The purpose of the following problem is for you to pay close attention to the exact formulation of the Pumping Lemma (PL).

The PL says that every RL has a pumping length p , such that every string in the language can be pumped if it has length p or more.

Note that if p is a pumping length for a language L then so is any other length $\geq p$. We define the *minimum pumping length* for L to be the smallest such p .

For example, if $L = ab^*$ then the minimum pumping length is 2. This is because the string $w = a$ is in L and has length 1, yet w cannot be pumped; but any string in L of length 2 or more contains a b and hence can be pumped by dividing it so that $x = a$, $y = b$ and z is the rest of the string.

For each of the following languages, give the minimum pumping length and justify your answer.

- 1) aab^*
- 2) a^*b^*
- 3) $aab + a^*b^*$
- 4) $a^*b^+a^+b^* + ba^*$

The notation a^+ is equivalent to aa^* , i.e. 1 or more a 's (as opposed to a^* which means zero or more a 's).

- 5) $(01)^*$
- 6) ε
- 7) $b^*ab^*ab^*$
- 8) $10(11^*0)^*0$
- 9) 1011
- 10) Σ^*

- (7) **(Pumping lemma applied to RLs)** When we try to apply the Pumping Lemma to a Regular Language the **Prover** wins, and the **Falsifier** loses.

Show why **Falsifier** loses when L is one of the following RLs:

- 1) $\{00, 11\}$
- 2) $(aa + bb)^*$
- 3) 01^*0^*1
- 4) \emptyset

Go through the JFLAP tutorial on: <http://www.jflap.org/tutorial/pumpinglemma/regular/> and then try all the “games.”

JFLAP plays the role of **Falsifier** and you play the role of **Prover**.

Note that *some of the languages below are actually regular* – in this case, you will need to devise a strategy for **Prover** to always win no matter what **Falsifier** chooses as a challenge string.

JFLAP’s notation:

- m is used instead of p (the pumping length).
- i is used instead of k in xy^kz .
- $n_a(w)$: the number of occurrence of the symbol a in the string w .
e.g. $n_a(aba) = 2$ and $n_b(aba) = 1$.
- w^R : the reverse string of w , e.g. $abb^R = bba$.

Assume $\Sigma = \{a, b\}$ unless otherwise specified.

The list of languages is as follows:

1. $\{a^n b^n \mid n \geq 0\}$ Hint: $a^p b^p$
2. $\{w \in \Sigma^* \mid n_a(w) < n_b(w)\}$ Hint: $a^p b^{p+1}$
i.e. language of strings which have less a’s than there are b’s.
3. $\{ww^R \mid w \in \Sigma^*\}$ Hint: $a^p b^{2p} a^p$
4. $\{(ab)^n a^m \mid n > m \geq 0\}$ Hint: $(ab)^{p+1} a^p$
5. $\{a^n b^m c^{n+m} \mid n \geq 0, m \geq 0\}$
6. $\{a^n b^\ell a^k \mid n > 5, \ell > 3, \ell \geq k\}$ Hint: Regular
7. $\{a^n \mid n \text{ is even}\}$ Hint: Regular
8. $\{a^n b^m \mid n \text{ is odd or } m \text{ is even}\}$ Hint: Regular
9. $\{bba(ba)^n a^{n-1} \mid n \geq 1\}$
10. $\{b^5 w \mid w \in \Sigma^* \text{ and } 2n_a(w) = 3n_b(w)\}$
11. $\{b^5 w \mid w \in \Sigma^* \text{ and } n_a(w) + n_b(w) \equiv 0 \pmod{3}\}$
12. $\{b^m (ab)^n (ba)^n \mid m \geq 4, n \geq 1\}$
13. $\{(ab)^{2n} \mid n \geq 1\}$ Hint: Regular

Warning: The games played by JFLAP are for a specific challenge string. This is only meant to give you a feel for how the general game proceeds. When we write our proofs we are not allowed to choose a fixed value for p .

- (1) Let $\Sigma = \{0, 1, +, =\}$, and ADD be the language given by

$$\{u=v+w \mid u, v, w \text{ are binary integers, and } u \text{ is the sum of } v \text{ and } w \text{ in the usual sense}\}$$

Show that ADD is not regular.

- (2) Let $L = \{1^{2^n} \mid n \geq 0\}$. Show that L cannot be regular.
 (3) $L = \{a^i b^j c^k \mid j \neq i \text{ or } j \neq k\}$

❶ **Prover** claims L is regular and fixes the value of the pumping length p .

❸ **Prover** writes

$$w = (xy)z = (a^p)b\boxed{}c\boxed{}$$

where xy is a string of a 's only

❷ **Falsifier** challenges **Prover** and chooses

$$w = a^p b \boxed{} c \boxed{}$$

Here $|w| = p + 2(\boxed{}) \geq p$.

❹ **Falsifier** forms

$$xy^k z = a^{p+(k-1)|y|} b \boxed{} c \boxed{}$$

where $k = 1 + \boxed{}/\boxed{}$. This gives $a^{p!+p} b \boxed{} c \boxed{}$ which is not in the language.