PL: if A is regular, then there exists p that, for any s in A and |s| > p, then

there exists a partition s=xyz, satisfying condition:

1. for each i>=0, xy^iz in A
2. |y| > 0
3. |xy| < p

PL => all regular languages are infinite

F  All finite languages are regular

Every DFA contains a loop

T  DFA runs on input of arbitrary length, there must be a loop

Every DFA contains a loop from which a final state is reachable

F  excludes DFAs for finite languages.

L = {a^n b^n | n>=0} is not regular

T  pumping lemma

Any subset of that L is not regular

F  empty subset

An infinite subset of that L is not regular

T  pumping lemma

if that L is a subset of L', then L' is not regular

F  ∑*

if L1 union L2 is regular then so are L1 and L2

F  L1 union L2 =∑*
if $L_1$ intersection $L_2$ is regular then so are $L_1$ and $L_2$

F  $L_1$ and $L_2$ disjoint

If $L_1$ and $L_2$ are regular, then $L_1$ union $L_2$ is regular

T  closure property

If $L_1$ and $L_2$ are regular, then $L_1$ intersection $L_2$ is regular

T  closure property

Application of Pumping Lemma.

$\Sigma = \{0,1,+,=\}$

ADD = \{a=b+c | a,b,c are binary integers and a is sum of b and c\}.

Solution:
a=10^p, b=1^p, c= 1

$|xy| < p$ and $|y| > 1 \Rightarrow x=\varepsilon \ y=10^i \ or \ x=10^i \ y=0^i$

Proof by closure properties of regular expression

If $L$ intersects $L'$ ($L'$ is regular) is not regular, then $L$ is not regular.

$\Sigma = \{0,1\}\, \text{L= \{the number of 0’s and the number of 1’s are equal\}}$

$L$ intersects $L'=${$0^i1^j | i,j \geq 0$} = {$0^i1^j | i \geq 0$}

$L'$ is regular, and $L$ intersects $L'$ is not regular $\Rightarrow L$ is not regular.
Many elements of programming languages are regular, e.g. Identifiers: the first being a letter of the alphabet or an underline, and the remaining being any letter of the alphabet, any numeric digit, or the underline int/float keywords.

A C program is not regular.

main(){return (...(0)...);}