Recursively Enumerable and Recursive Languages

Recall

Definition (class 18.pdf)

Definition 10.4, Linz, 6th, page 279

Let S be a set of strings.

An **enumeration procedure** for S is a Turing Machine that generates all strings of S one by one

and

each string is generated in finite time.

Definition:

A language is **recursively enumerable** if some Turing machine accepts it

Let L be a recursively enumerable language

and M the Turing Machine that accepts it

For string w :

if $w \in L$ then M halts in a final state

if $w \notin L$ then M halts in a non-final state

or loops forever

Definition (11.2, page 287):

A language is **recursive**
if some Turing machine accepts it
and halts on any input string

In other words:

A recursive language has a
membership algorithm.

Let L be a recursive language

and M the Turing Machine that accepts it

L is a recursive language if there is a Turing Machine M such that

For any string w :

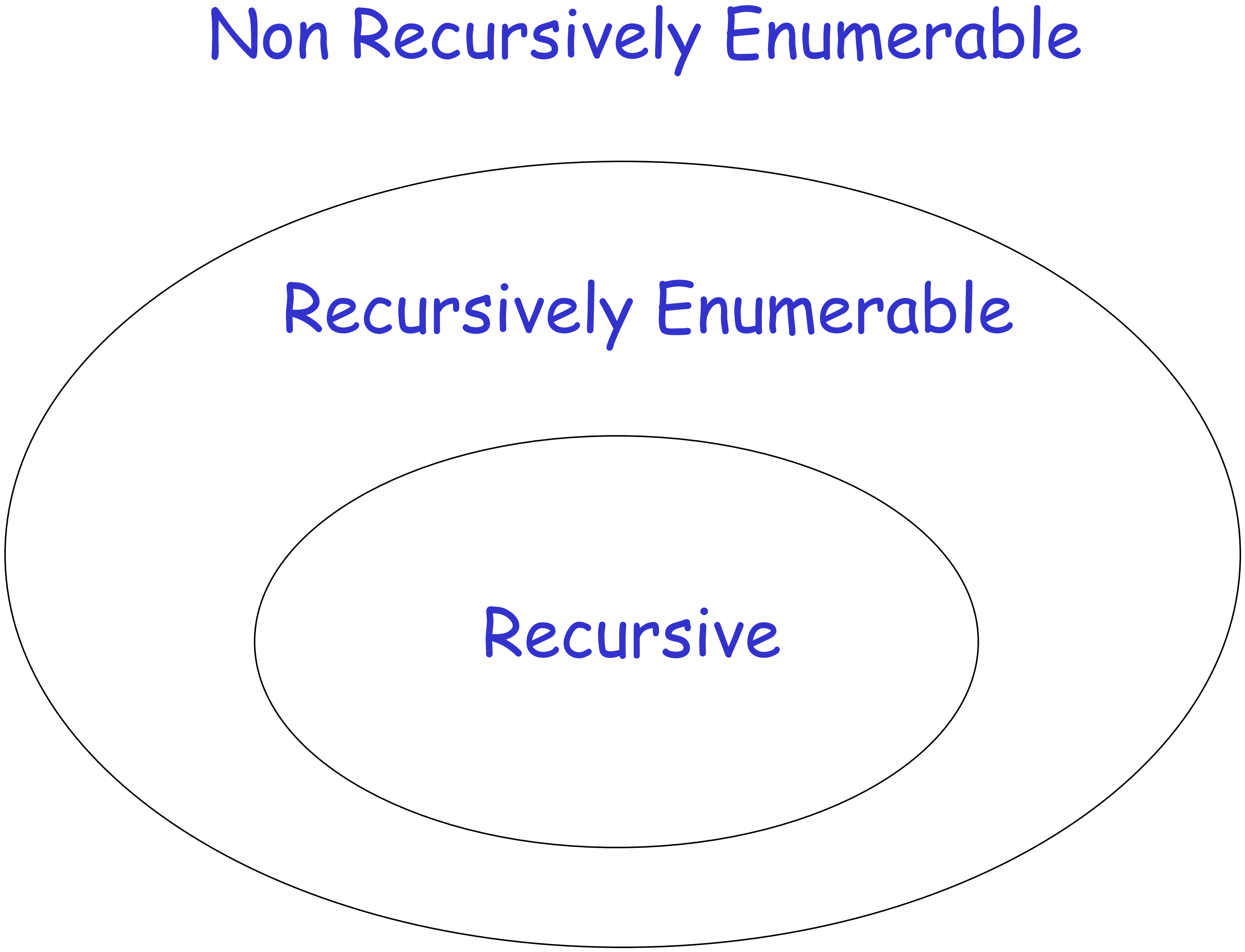
if $w \in L$ then M halts in a final state

if $w \notin L$ then M halts in a non-final state

We will prove:

1. There is a specific language which is not recursively enumerable (not accepted by any Turing Machine)

2. There is a specific language which is recursively enumerable but not recursive


Non Recursively Enumerable
Recursively Enumerable
Recursive

We will prove:

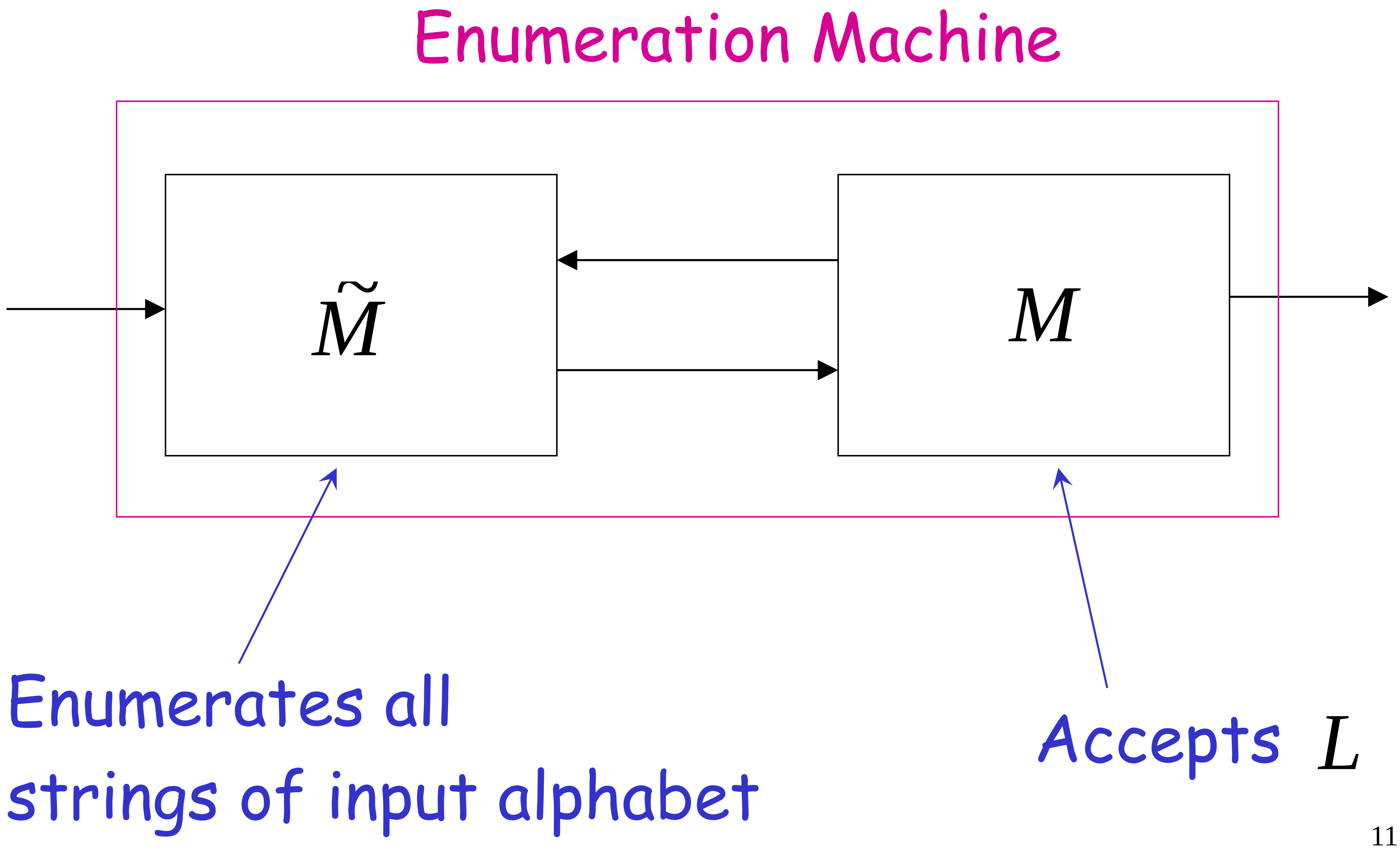
1. If a language is recursive then there is an enumeration procedure for it.
Linz 6th, page 287.

2. A language is recursively enumerable if and only if there is an enumeration procedure for it.
Linz 6th, page 287.

Theorem:

if a language L is recursive then there is an enumeration procedure for it

Proof:


Enumeration Machine
$\tilde{M}$
M
Enumerates all strings of input alphabet
Accepts L

If the alphabet is $\{a,b\}$ then $\tilde{M}$ can enumerate strings as follows:

$$a$$
$$b$$
$$aa$$
$$ab$$
$$ba$$
$$bb$$
$$aaa$$
$$aab$$
$$\ldots\ldots$$

Enumeration procedure

```
Repeat:
    $\tilde{M}$ generates a string $w$
    $M$ checks if $w \in L$
        YES: print $w$ to output
        NO: ignore $w$
```

End of Proof

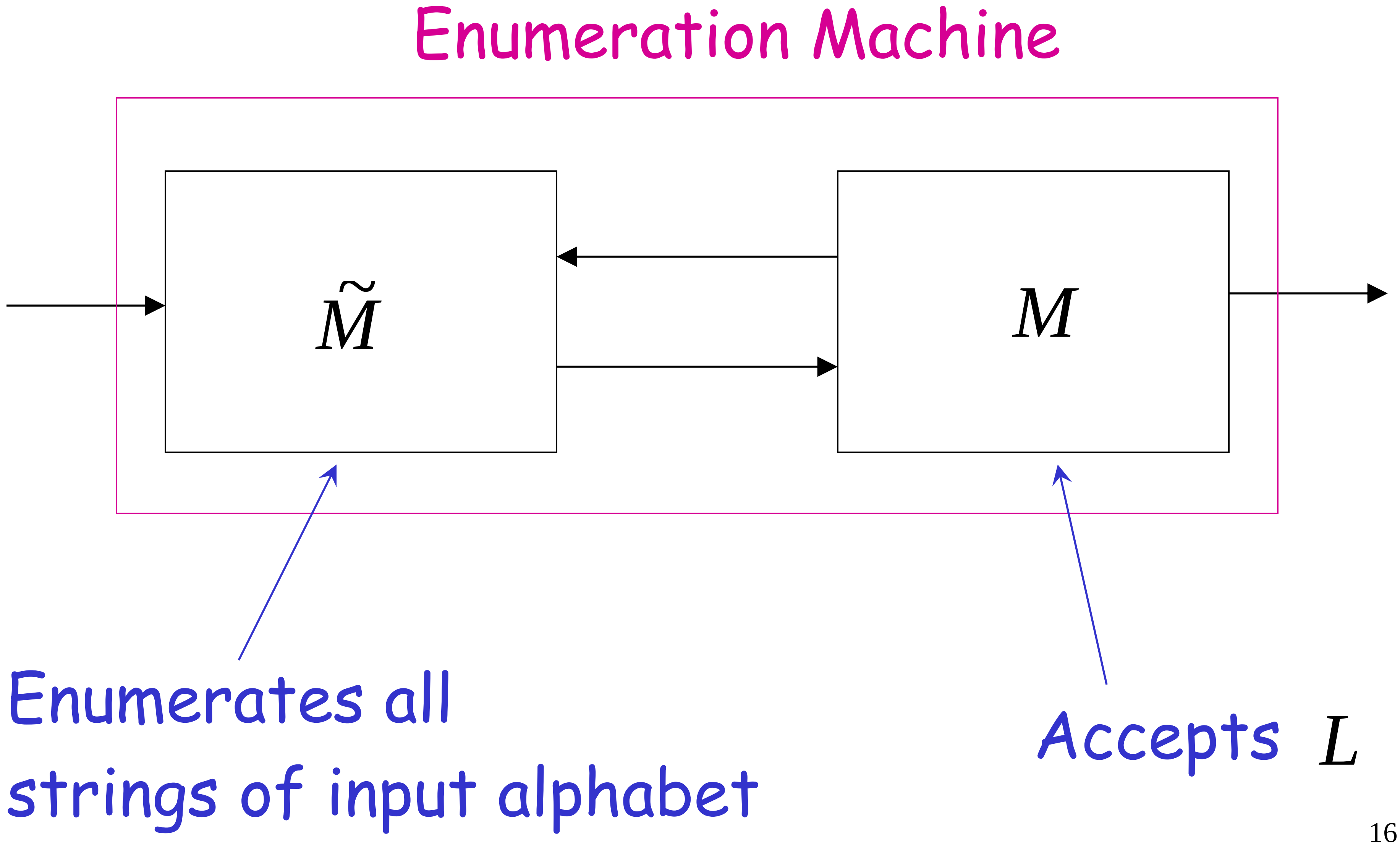
Example: $L = \{b, ab, bb, aaa, \ldots\}$

$\tilde{M}$	$L(M)$	Enumeration Output
a		
b	b	b
aa		
ab	ab	ab
ba		
bb	bb	bb
aaa	aaa	aaa
aab		
$\ldots\ldots$	$\ldots\ldots$	$\ldots\ldots$

Theorem:

if language L is recursively enumerable, then there is an enumeration procedure for it

Proof:


Enumeration Machine
$\tilde{M}$
M
Enumerates all strings of input alphabet
Accepts L

If the alphabet is $\{a, b\}$ then $\tilde{M}$ can enumerate strings as follows:

a

b

aa

ab

ba

bb

aaa

aab

NAIVE APPROACH

Enumeration procedure

Repeat: $\tilde{M}$ generates a string w

M checks if $w \in L$

YES: print w to output

NO: ignore w

Problem: If $w \notin L$

machine M may loop forever

BETTER APPROACH

$\tilde{M}$ Generates first string w_1

M executes first step on w_1

$\tilde{M}$ Generates second string w_2

M executes first step on w_2

second step on w_1

$\tilde{M}$ Generates third string w_3

M executes first step on w_3

second step on w_2

third step on w_1

And so on............

Step in string

$$\begin{tikzcd}
w_1 & w_2 & w_3 & w_4 & ʌ \\
1 \arrow[r] & 1 \arrow[dl] & 1 \arrow[dl] & 1 \arrow[dl] & \\
2 \arrow[urr] & 2 \arrow[dl] & 2 \arrow[dl] & 2 & \\
3 \arrow[uurrr] & 3 & 3 & 3 &
\end{tikzcd}$$

ʌ

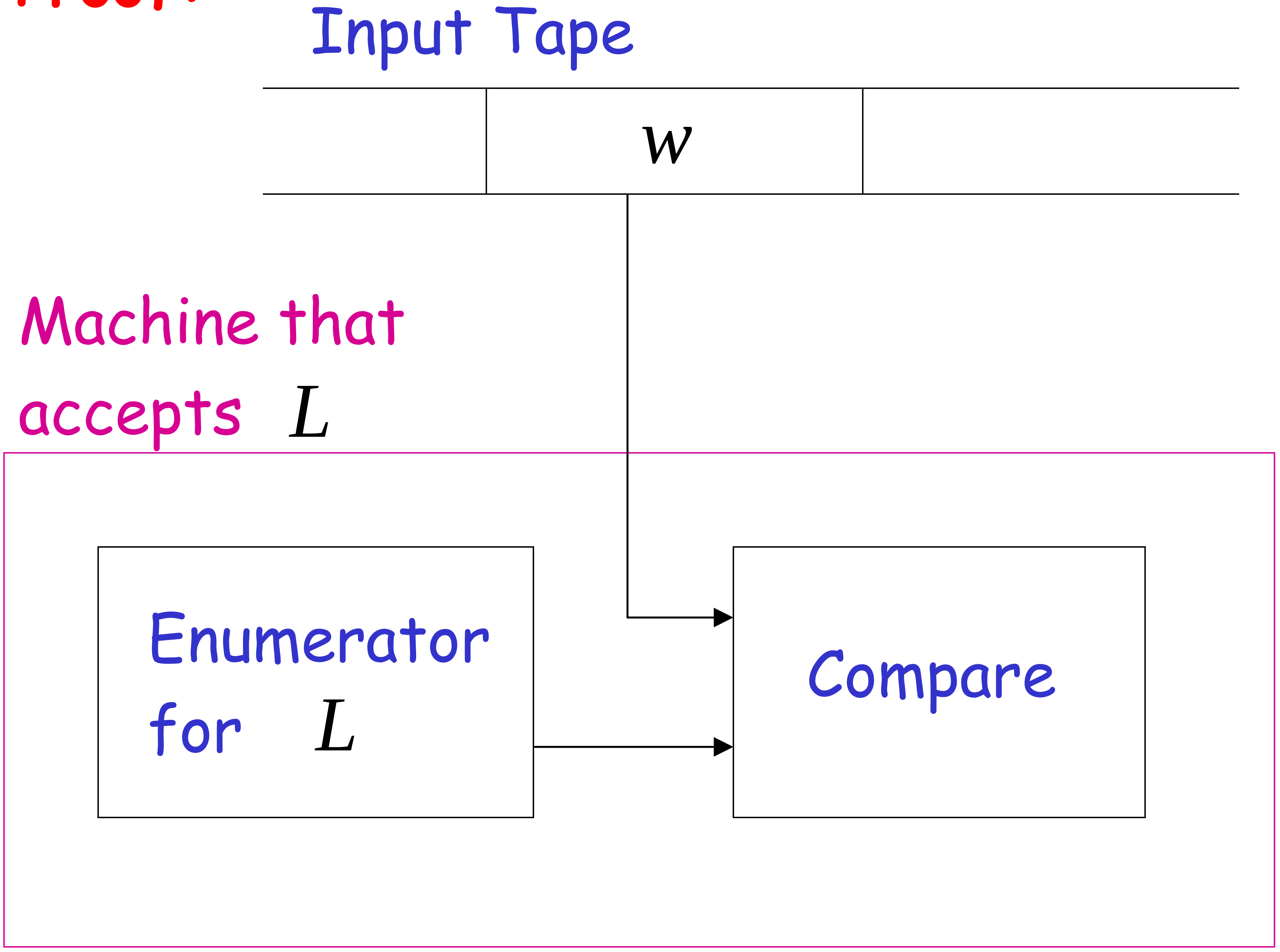
If for any string w_i the machine M halts in a final state, then it prints w_i on the output

End of Proof

Theorem:

If for language L
there is an enumeration procedure,
then L is recursively enumerable

Proof:


Input Tape
w
Machine that accepts L
Enumerator for L
Compare

Turing machine that accepts L

For input string w

Repeat:

- Using the enumerator, generate the next string of L

- Compare generated string with w

  If same, accept and exit loop

End of Proof

We have proven:

A language is recursively enumerable
if and only if
there is an enumeration procedure for it

A Language which is not Recursively Enumerable

We want to find a language that is not Recursively Enumerable

This language is not accepted by any Turing Machine

Consider alphabet $\{a\}$

Strings: $a, \; aa, \; aaa, \; aaaa, \; \mathrm{K}$

$a^1 \quad a^2 \quad a^3 \quad a^4 \quad \Lambda$

Consider Turing Machines
that accept languages over alphabet $\{a\}$

They are countable:

$$M_1,\ M_2,\ M_3,\ M_4,\ \ldots$$

Example language accepted by M_i

$$L(M_i) = \{aa, aaaa, aaaaaa\}$$

$$L(M_i) = \{a^2, a^4, a^6\}$$

Alternative representation

	a^1	a^2	a^3	a^4	a^5	a^6	a^7	ʌ
$L(M_i)$	0	1	0	1	0	1	0	ʌ

	a^1	a^2	a^3	a^4	ʌ
$L(M_1)$	0	1	0	1	ʌ
$L(M_2)$	1	0	0	1	ʌ
$L(M_3)$	0	1	1	1	ʌ
$L(M_4)$	0	0	0	1	ʌ

Consider the language

$$L = \{a^i : a^i \in L(M_i)\}$$

L consists from the 1's in the diagonal

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \Lambda \\
\hline
L(M_1) & 0 & 1 & 0 & 1 & \Lambda \\
L(M_2) & 1 & 0 & 0 & 1 & \Lambda \\
L(M_3) & 0 & 1 & \textcircled{1} & 1 & \Lambda \\
L(M_4) & 0 & 0 & 0 & \textcircled{1} & \Lambda
\end{array}
$$

$$L = \{a^3, a^4, \mathrm{K}\}$$

Consider the language $\overline{L}$

$$L = \{a^i : a^i \in L(M_i)\}$$

$$\overline{L} = \{a^i : a^i \notin L(M_i)\}$$

$\overline{L}$ consists from the 0's in the diagonal

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \wedge \\
\hline
L(M_1) & \textcircled{0} & 1 & 0 & 1 & \wedge \\
L(M_2) & 1 & \textcircled{0} & 0 & 1 & \wedge \\
L(M_3) & 0 & 1 & 1 & 1 & \wedge \\
L(M_4) & 0 & 0 & 0 & 1 & \wedge
\end{array}
$$

$$\overline{L} = \{a^1, a^2, \mathrm{K}\}$$

Theorem:

Language $\overline{L}$ is not recursively enumerable

Proof:

Assume for contradiction that

$\overline{L}$ is recursively enumerable

There must exist some machine M_k that accepts $\overline{L}$

$$L(M_k) = \overline{L}$$

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \wedge \\
\hline
L(M_1) & \textcircled{0} & 1 & 0 & 1 & \wedge \\
L(M_2) & 1 & \textcircled{0} & 0 & 1 & \wedge \\
L(M_3) & 0 & 1 & 1 & 1 & \wedge \\
L(M_4) & 0 & 0 & 0 & 1 & \wedge
\end{array}
$$

Question: $M_k = M_1$?

$$\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\
\hline
L(M_1) & \textcircled{0} & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & \textcircled{0} & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & 1 & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & 1 & \cdots
\end{array}$$

$$a^1 \in L(M_k)$$

Answer: $M_k \neq M_1$

$$a^1 \notin L(M_1)$$

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	$\textcircled{0}$	1	0	1	$\cdots$
$L(M_2)$	1	$\textcircled{0}$	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Question: $M_k = M_2$?

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\
\hline
L(M_1) & \boxed{0} & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & \boxed{0} & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & 1 & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & 1 & \cdots
\end{array}
$$

$$a^2 \in L(M_k)$$

Answer: $M_k \neq M_2$

$$a^2 \notin L(M_2)$$

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\
\hline
L(M_1) & \textcircled{0} & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & \textcircled{0} & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & 1 & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & 1 & \cdots
\end{array}
$$

Question: $M_k = M_3$?

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\
\hline
L(M_1) & 0 & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & 0 & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & 1 & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & 1 & \cdots
\end{array}
$$

$$a^3 \notin L(M_k)$$

Answer: $M_k \neq M_3$

$$a^3 \in L(M_3)$$

Similarly: $M_k \neq M_i$ for any i

Because either:

$a^i \in L(M_k)$ $a^i \notin L(M_k)$

or

$a^i \notin L(M_i)$ $a^i \in L(M_i)$

Therefore, the machine M_k cannot exist

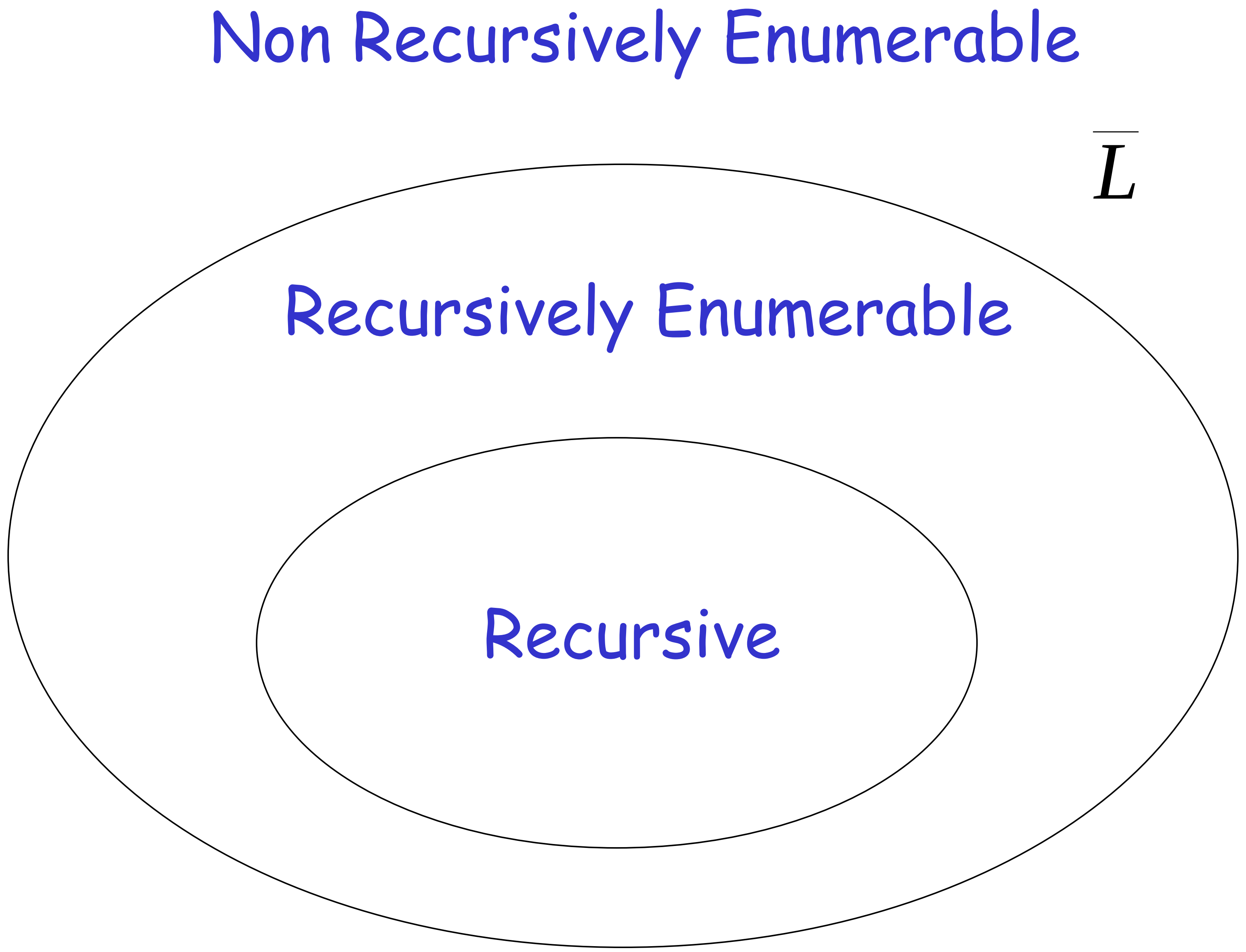
Therefore, the language $\overline{L}$ is not recursively enumerable

End of Proof

Observation:

There is no algorithm that describes $\overline{L}$

(otherwise $\overline{L}$ would be accepted by some Turing Machine)


Non Recursively Enumerable
$\overline{L}$
Recursively Enumerable
Recursive

A Language which is Recursively Enumerable and not Recursive

We want to find a language which

Is recursively enumerable

There is a Turing Machine that accepts the language

But not recursive

The machine doesn't halt on some input

We will prove that the language

$$L = \{a^i : a^i \in L(M_i)\}$$

Is recursively enumerable

but not recursive

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \wedge \\
\hline
L(M_1) & 0 & 1 & 0 & 1 & \wedge \\
L(M_2) & 1 & 0 & 0 & 1 & \wedge \\
L(M_3) & 0 & 1 & \boxed{1} & 1 & \wedge \\
L(M_4) & 0 & 0 & 0 & \boxed{1} & \wedge
\end{array}
$$

$$L = \{a^3, a^4, \mathrm{K}\}$$

Theorem:

The language $L = \{a^i : a^i \in L(M_i)\}$

is recursively enumerable

Proof:

We will give a Turing Machine that accepts L

Turing Machine that accepts L

For any input string w

- Compute i, for which $w = a^i$
- Find Turing machine M_i

  (using the enumeration procedure for Turing Machines)

- Simulate M_i on input a^i
- If M_i accepts, then accept w

End of Proof

Observation:

Recursively enumerable

$$L = \{a^i : a^i \in L(M_i)\}$$

Not recursively enumerable

$$\overline{L} = \{a^i : a^i \notin L(M_i)\}$$

(Thus, also not recursive)

Theorem:

The language $L = \{a^i : a^i \in L(M_i)\}$

is not recursive

Proof:

Assume for contradiction that L is recursive

Then $\overline{L}$ is recursive:

Take the Turing Machine M that accepts L

M halts on any input:

If M accepts then reject

If M rejects then accept

Therefore:

$\overline{L}$ is recursive

But we know:

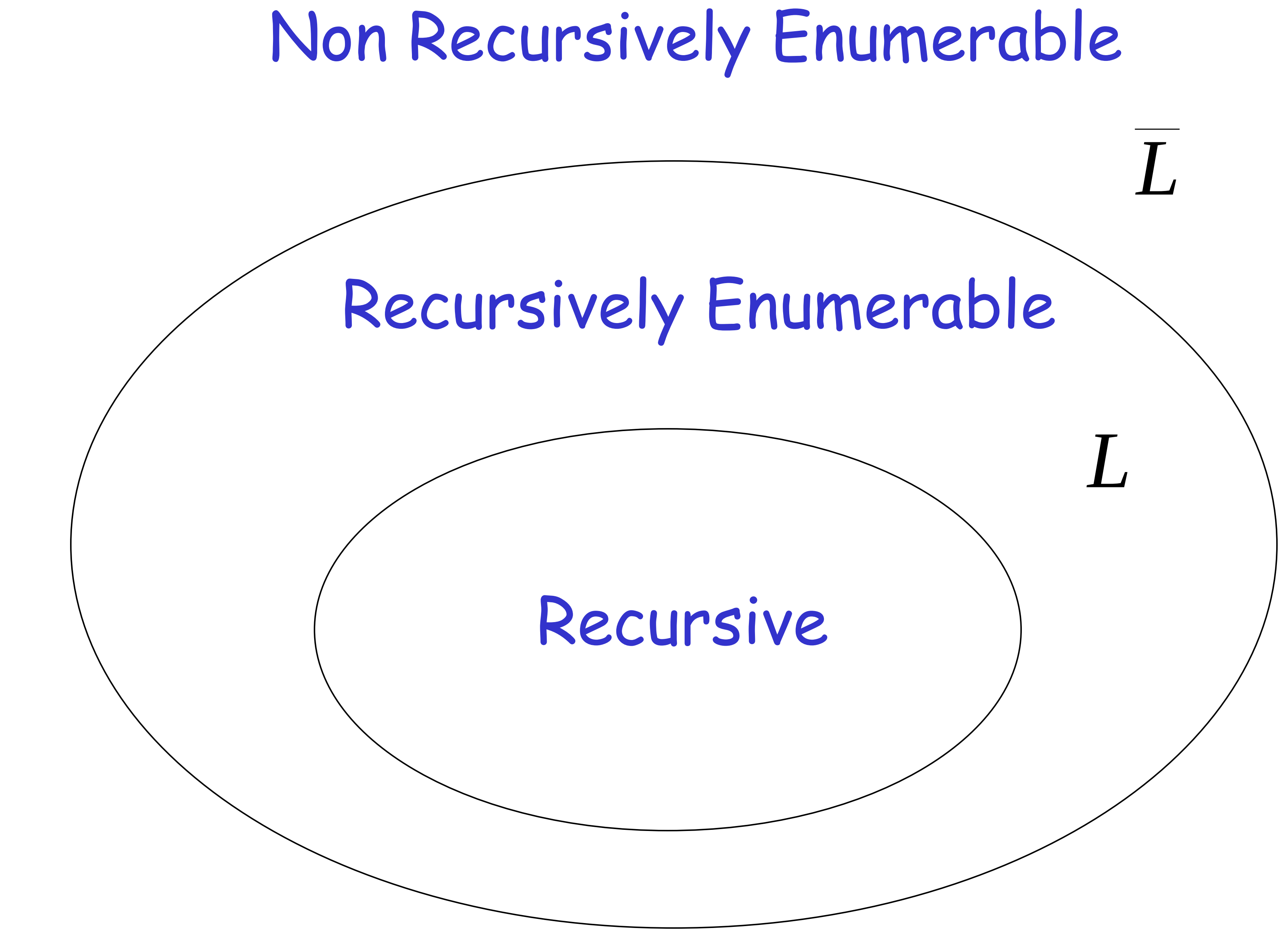
$\overline{L}$ is not recursively enumerable

thus, not recursive

CONTRADICTION!!!!

Therefore, L is not recursive

End of Proof


Non Recursively Enumerable
$\overline{L}$
Recursively Enumerable
L
Recursive