Theory of Computing:

9. *Turing Machine - 1*

Professor Imed Bouchrika
National School of Artificial Intelligence
imed.bouchrika@ensia.edu.dz

Outline :

- TM Architecture
- Examples
  - *{w | w in the form 0^*1^* }*
  - *{w | w contains 101}*
  - *$\{0^n1^n \mid n >= 0\}$*
  - *$\{a^nb^nc^n \mid n >= 0\}$*
  - *{w#w | w in $\{0,1\}^*$ }*
  - *$\{1^n x 1^m = 1^{n+m}\}$*
- Formalism for TM
- Classes of Languages

Computer Science & Programming

- Programming variables, computer science….

Chomsky Classification of Languages

- Lastly, Turing Machine will be explained next week

Type	Grammar	Language	Automaton
Type-3	Regular Grammar	Regular Languages	DFA/NFA
Type-2	Context-Free Grammar	Context Free Languages	PDA
Type-1	Context-Sensitive Grammar	Context-Sensitive Languages	Linear-bounded automaton
Type-0	Unrestricted grammar	Recursively enumerable language	Turing Machine

TM Architecture : Introduction

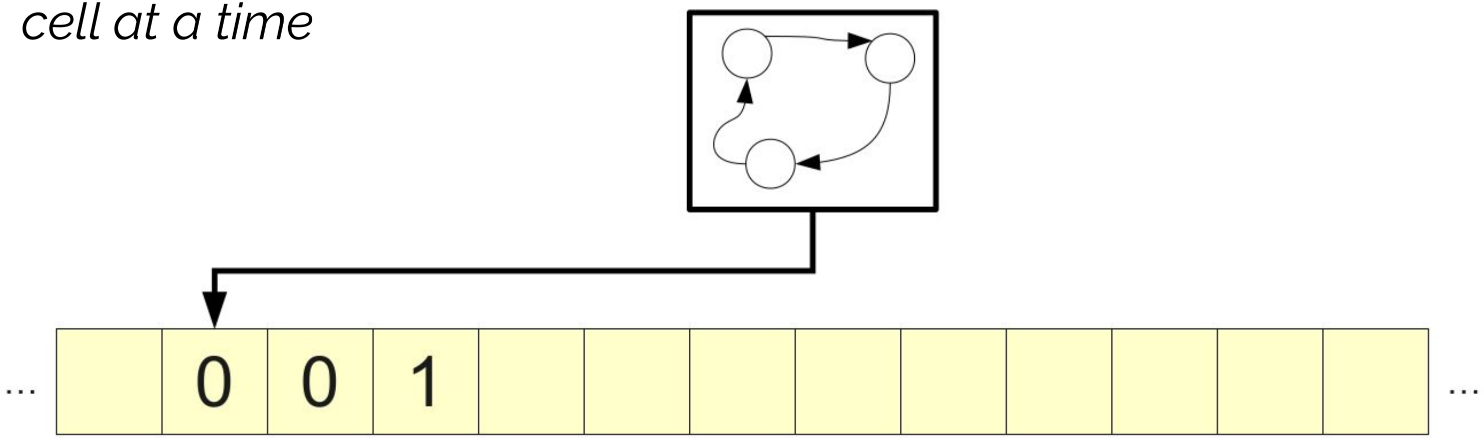
- Alan Turing aimed to design a computational machine which is :
  - Simple
  - Intuitive
  - Generic and
  - Formalizes the computation performed by a **human mind**

TM Architecture : Introduction

- Turing Machine was introduced in by its a British mathematician Alan Turing in 1936
- Turing machine is a much more accurate model of a general purpose computer with almost the same power.
    - It can execute any algorithm.

TM Architecture : Definition

- *A Turing machine is a finite automaton equipped with an infinite tape as its working memory.*
- *The machine has a tape head that can read and write a single memory cell at a time*



TM Architecture : Components

- A Turing Machine (TM) has three components:
  - An infinite tape divided into cells.
    - Each cell contains one symbol.
    - By Default, all cells are filled with the Blank Symbol : ⊔
    - The input string is placed on the tape at the left side.
    - Other symbols not from the language alphabet can be written to the tape
    - By default, the tape is infinite from the right side.

TM Architecture : Components

- A Turing Machine (TM) has three components:
  - A head that accesses one cell at a time:
    - It can both read from and write on the tape
    - It can move both left and right (based on the transitions)

TM Architecture : Components

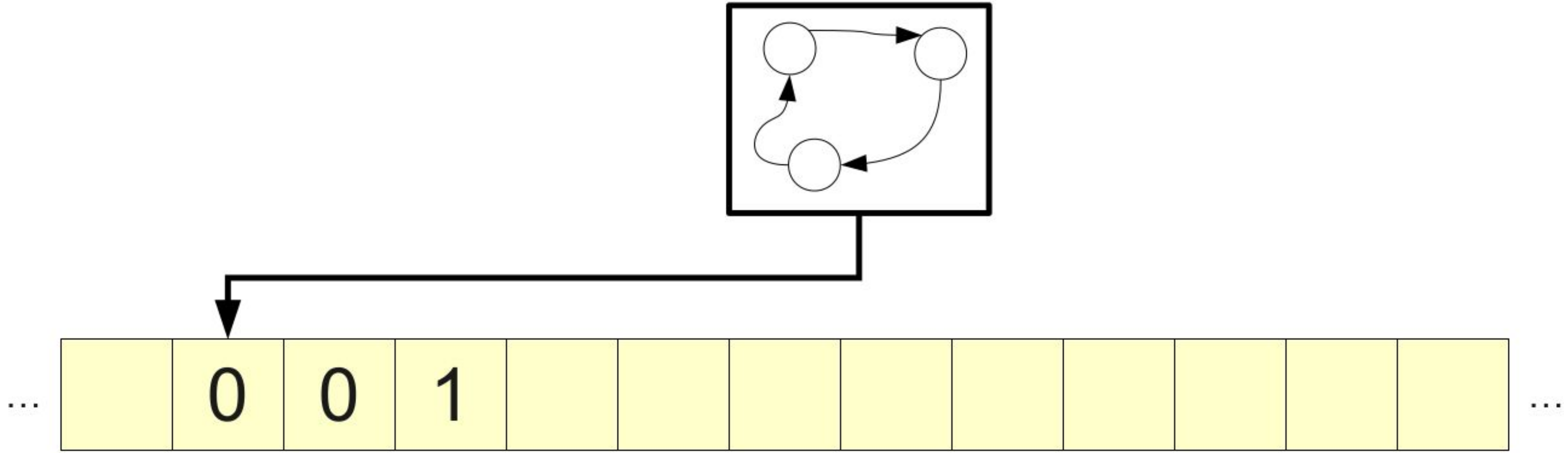
- A Turing Machine (TM) has three components:
    - A program memory or Controller for issuing commands :
        - Finite number of states
        - The transitions between the states

TM Architecture : Execution

- At each step , the Turing Machine :

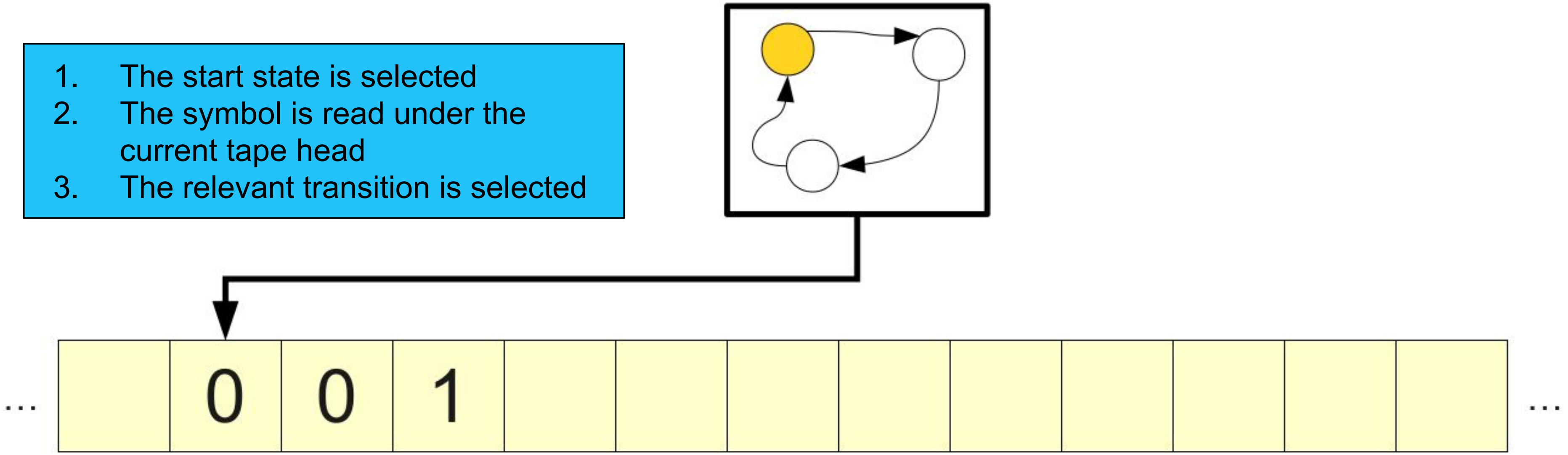
  1. Read the cell symbol from the Tape
  2. Decides which transition to make
  3. Write a Symbol to tape cell under the current tape head
  4. Changes the state
  5. Moves the tape head to the left or to the right.

TM Architecture : Execution & Simulation



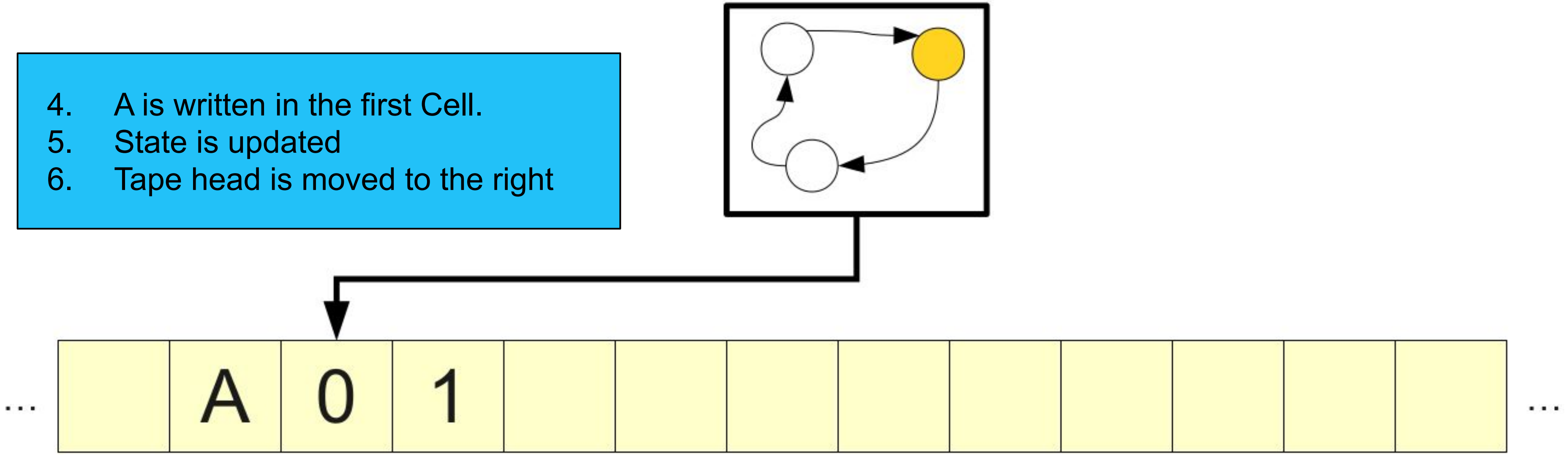
TM Architecture : Execution & Simulation

1. The start state is selected
2. The symbol is read under the current tape head
3. The relevant transition is selected

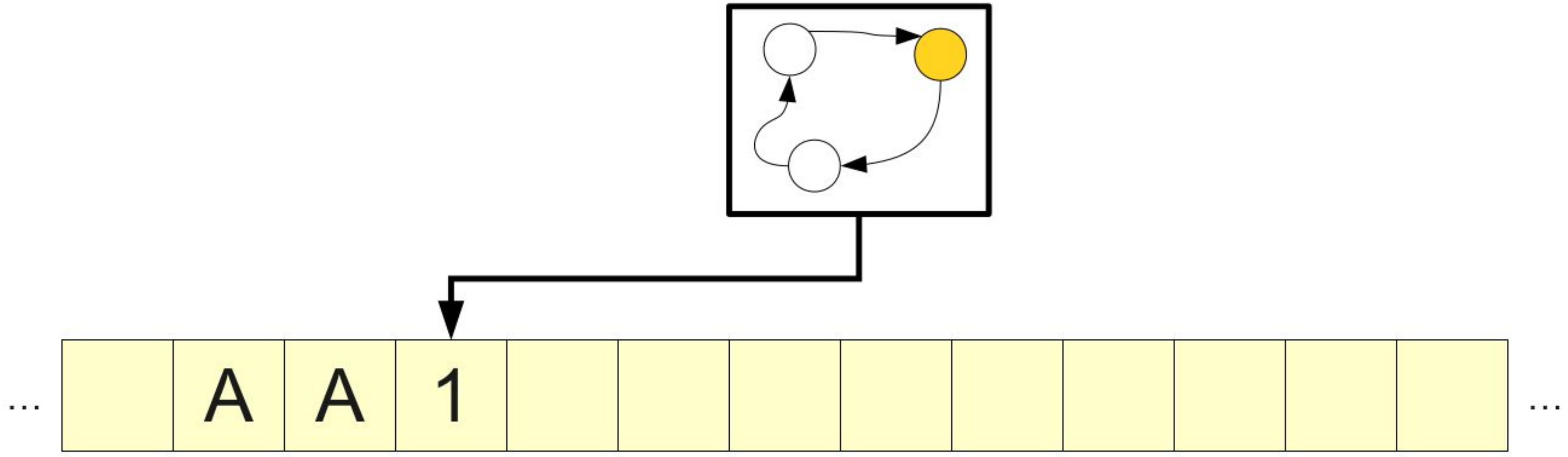


TM Architecture : Execution & Simulation

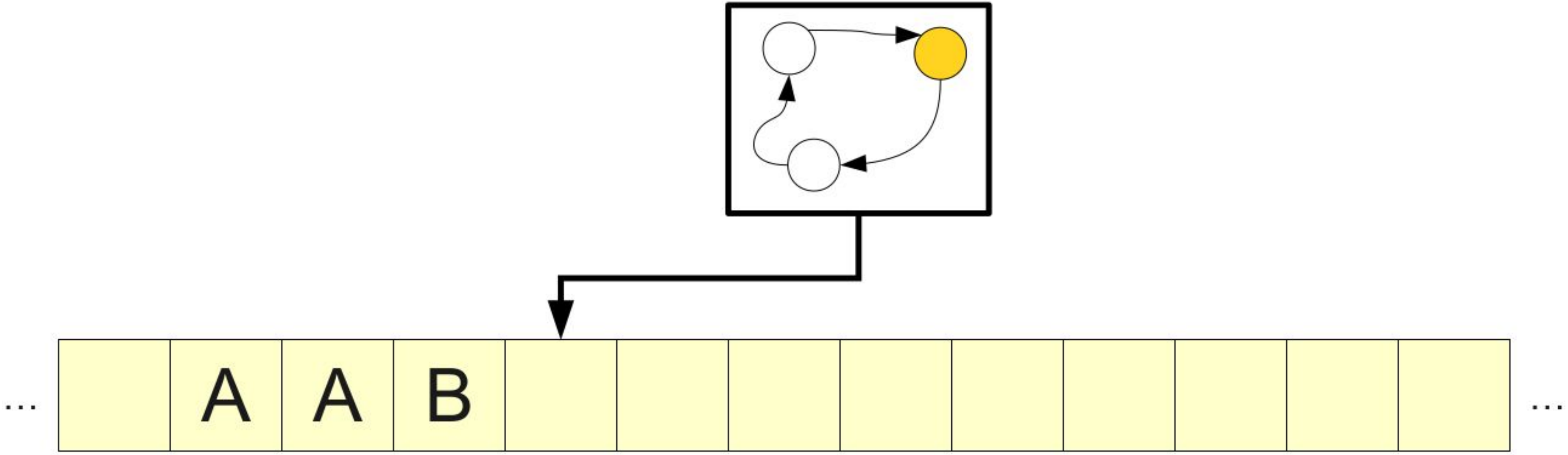
4. A is written in the first Cell.
5. State is updated
6. Tape head is moved to the right



TM Architecture : Execution & Simulation



TM Architecture : Execution & Simulation



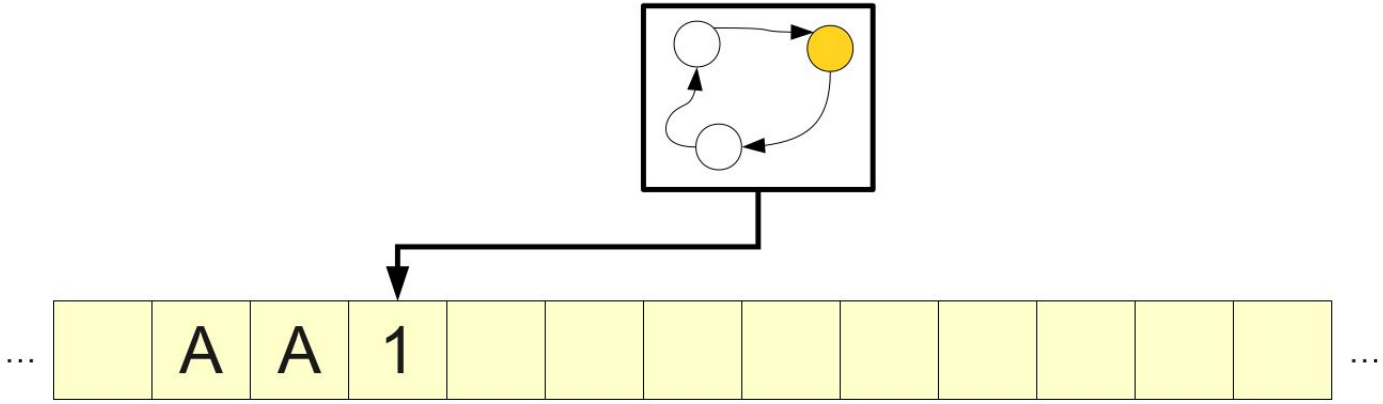
TM Architecture : Execution & Simulation

Move Right

Move Left

TM Architecture : Transitions

- Turing Machine is represented as a diagram like Finite State Machine
    - Except that each arrow is labeled with the following format:
        - A —>B, R : When you read A, replace it with B and move **right**



TM Architecture : Transitions

- Turing Machine is represented as a diagram like Finite State Machine
  - Except that each arrow is labeled with the following format:
    - A —> B, R : When you read A, replace it with B and move Right
    - A —> B, L : When you read A, replace it with B and move Left
    - A —> R : When you read A, replace it with A and move Right
    - A —> L : When you read A, replace it with A and move Left

TM Architecture : Transitions : Notations

- Depending on the textbook or lecture notes:
  - Transition : a $\rightarrow$: means when you read a, move right.
  - Transition : $\triangleright$ $\rightarrow$: means when you are at the start of the tape from the left side, move right. Assumption that the first cell on the left contain the $\triangleright$ symbol
  - Transition : ABR : when you read A, replace it with B, and move Right
  - Different notations for the blank symbol : $\sqcup$ or $\square$ or $\triangle$
  - There are other textbooks claiming the direction **S** symbol to stay in place ??

TM Architecture : States

- Turing machine has the following types of states
  - Single Start State
  - Intermediate States
  - Accept State : with the label as Accept and drawn in double lines
  - Reject State : with the label : reject , single line

TM Architecture : States

- Turing machine has the following types of states
  - Single Start State
  - Intermediate States
  - Accept State
  - Reject State : At any state if there is no relevant transition, it means, there is implicitly a transition to the reject state.

TM Architecture : Accepting vs. Rejecting vs …

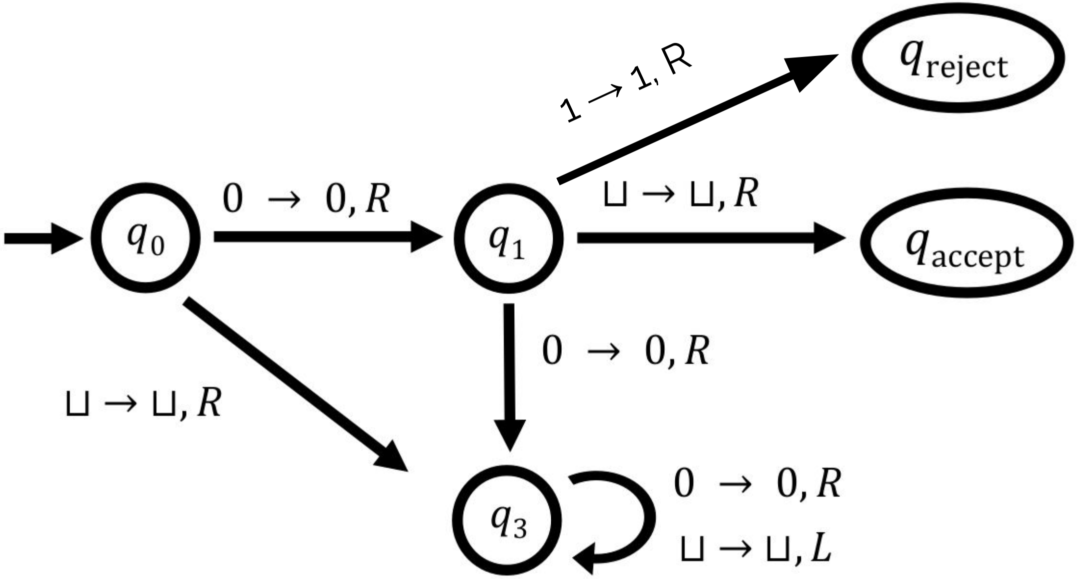
- The output for a Turing machine for a given input strings:
  - Halts
  - Keeps looping without halting :
    - TM keeps finding valid transitions between states even for a smaller input string

TM Architecture : Accepting vs. Rejecting vs …

- The output for a Turing machine for a given input strings:
  - Halts
    - Accept State : the word is accepted to be part of the language
    - Reject State : the word is not accepted within the language
  - Keeps looping without halting:
    - Undecidable

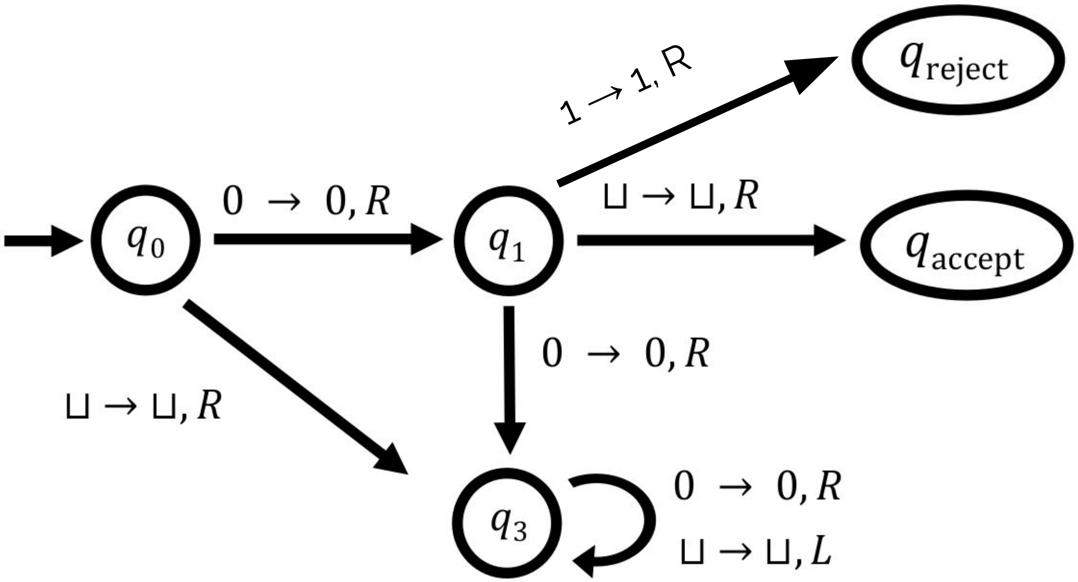
TM Architecture : Accepting vs. Rejecting vs ...

- What does this TM do in input : **000**



TM Architecture : Accepting vs. Rejecting vs ...

- What does this TM do in input : **000**
  - **Halt and Accept**
  - **Halt and Reject**
  - **Halt in State q3**
  - **Loop Forever**



TM Architecture : Machine Configuration

- At each step, the machine would have a configuration reflecting :
  - Current state
  - Position of the Tape Head
  - Content of the Tap

TM Architecture : Machine Configuration

- At each step, the machine would have a configuration reflecting :
  - Sipser prints the symbols in the tape whilst show the current state just before the head, Examples
    - q1 0000 : Head at State Q1 at the beginning of the tape
    - ␣y q5 **x**x␣ : Tape contains two cells containing ␣ and y, the tape head points at **x whilst the current state is q5**

TM Architecture : Machine Configuration

- A preferable way is to show the transitions table , whilst each row represents a configuration

Time	Configuration	State	Tape						
0	C_0	q_0	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
1	C_1	q_1	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
2	C_2	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
3	C_3	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
4	C_4	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
5	C_5	q_{rej}	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$

TM Architecture : Construction

- Creating and Innovative Process
    - Describe in English the algorithm containing the instructions :
        - How to move the head
        - What to write on the tape
    - Visualize your algorithm with the state diagram

TM Architecture : Construction

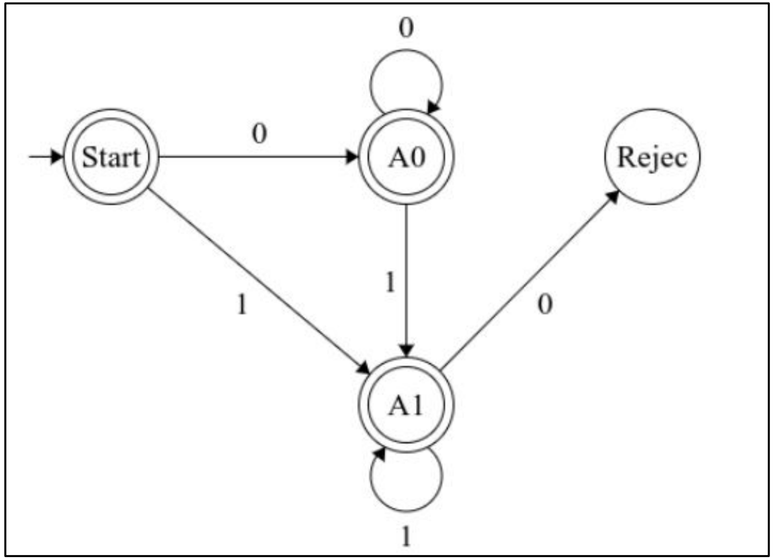
- Very important assumption :
  - Given words to be processed by a Turing machine, should never contain blank “space”

Examples for TM : 0^*1^*

- The DFA for the language 0^*1^* is given as :

Examples for TM : 0^*1^*

- The DFA for the language 0^*1^* is given as :

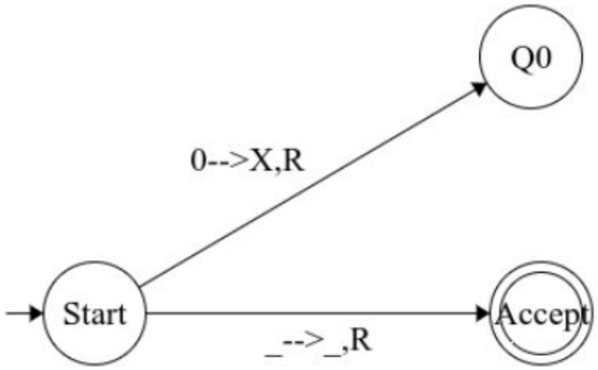


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

0	0	0	0	0	0	1	1	1	⊔

Start with the simple case
When the string is empty
Or : when first letter is 0

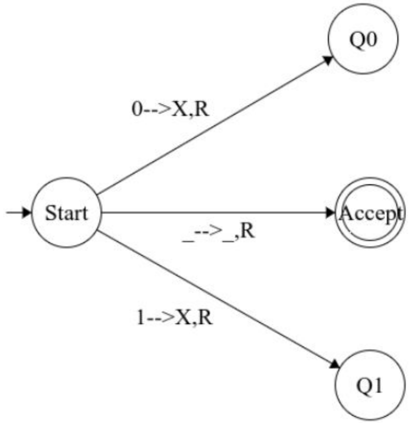


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	0	0	0	0	0	1	1	1	⊔

What if first letter is 1 ?

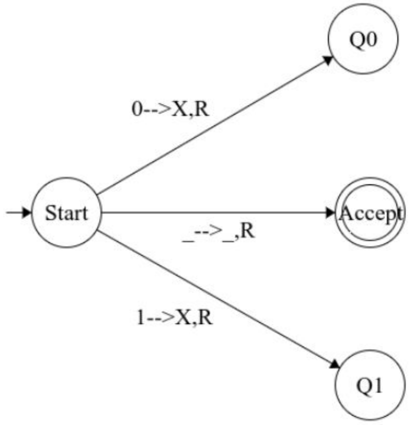


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	0	0	0	0	0	1	1	1	␣

If input string is only : 0

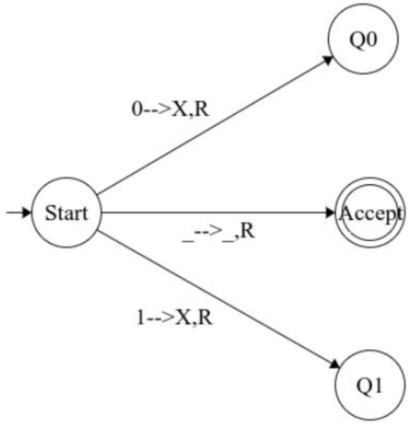


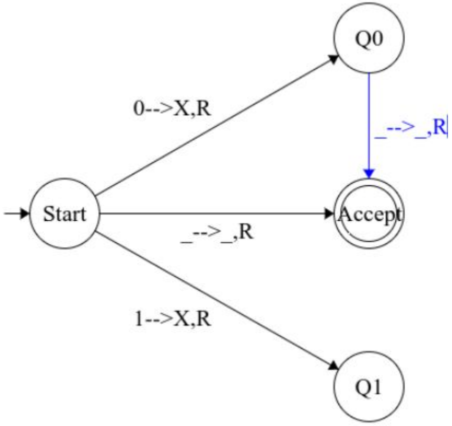
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	0	0	0	0	0	1	1	1	⊔

If input string is only : 0



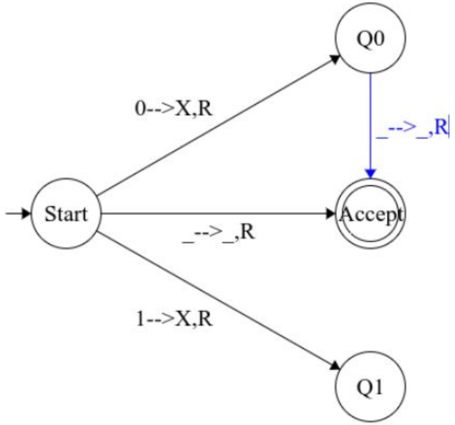


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	1	1	1	⊔

If input string is : 000…00

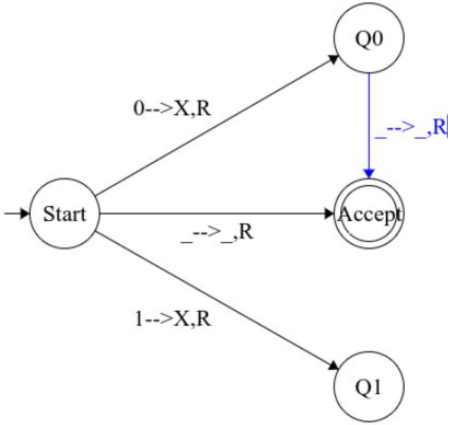


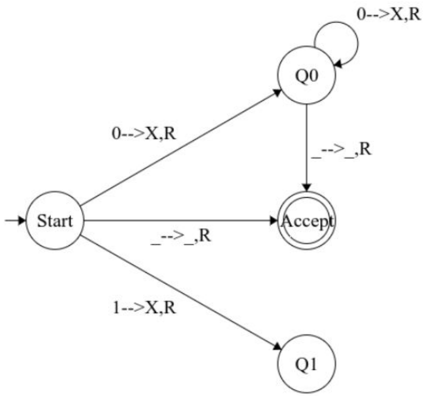
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	1	1	1	⊔

If input string is : 000…00



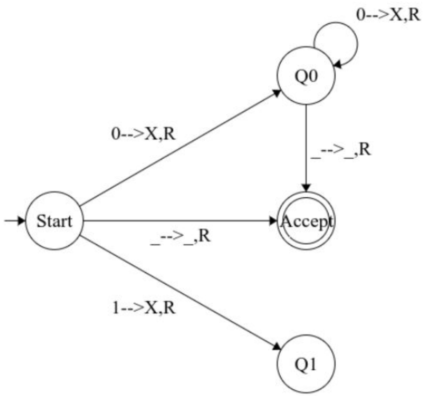


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	1	1	␣

If input string is : 00…01

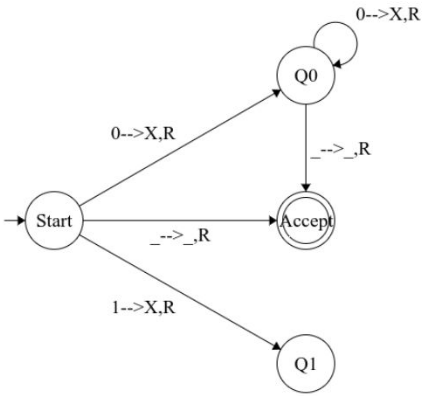


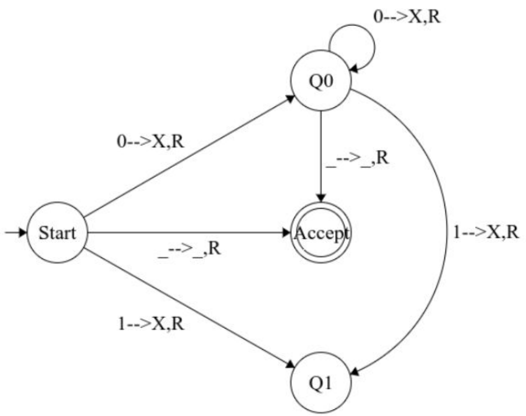
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	1	1	⊔

If input string is : 00…01



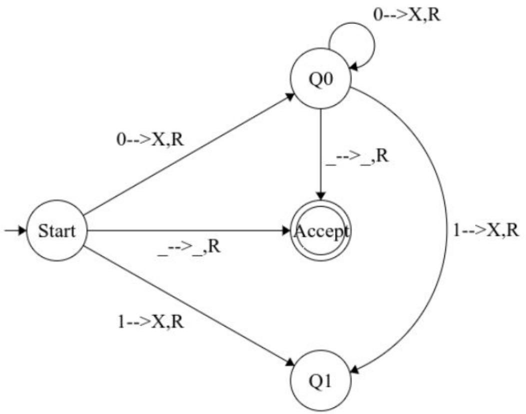


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	1	1	⊔

TM needs to terminate at Accept

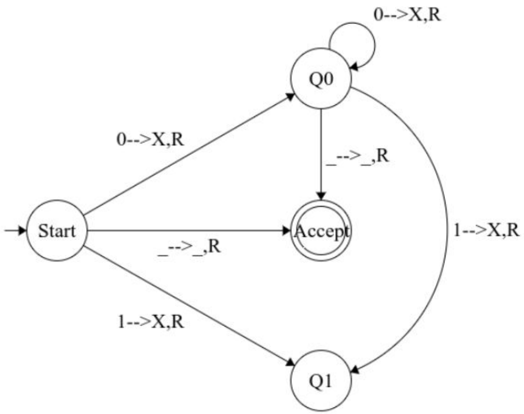


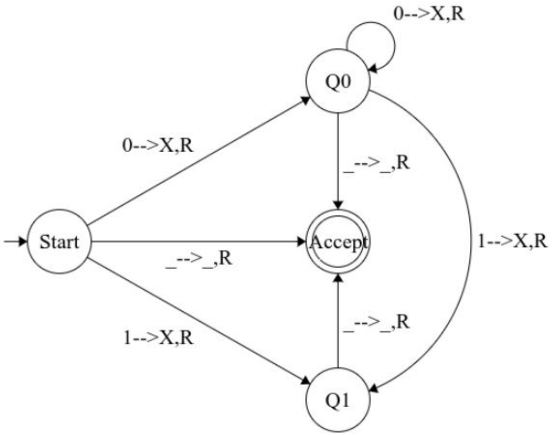
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	1	1	⊔

TM needs to terminate at Accept



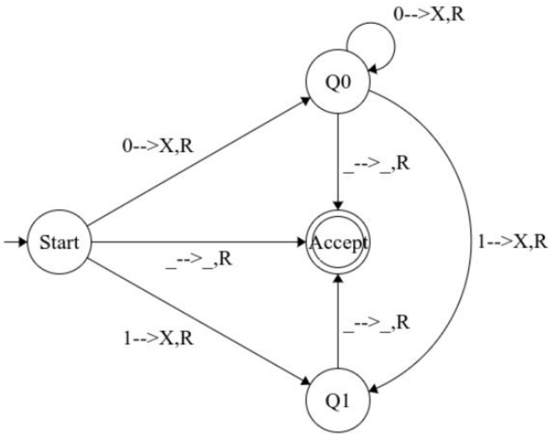


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	X	X	⊔

If input string is :
00…0111..11
Self-Loop is added

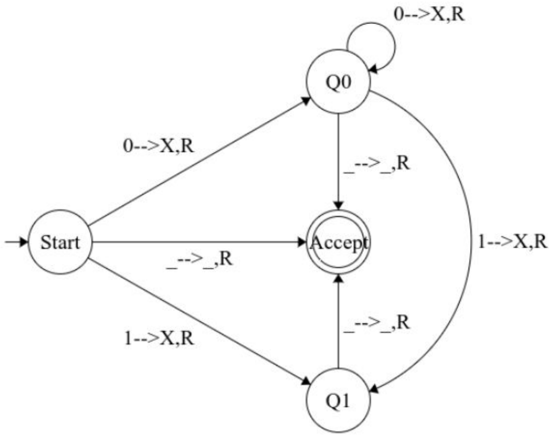


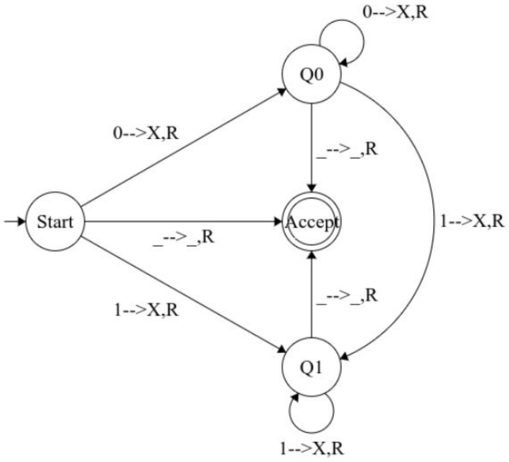
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	X	X	⊔

If input string is :
00…0111..11
Self-Loop is added



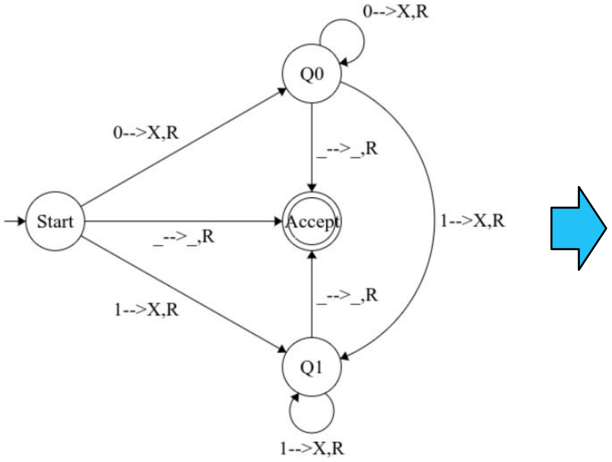


Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	X	X	⊔

If input string is : 00…010011..11

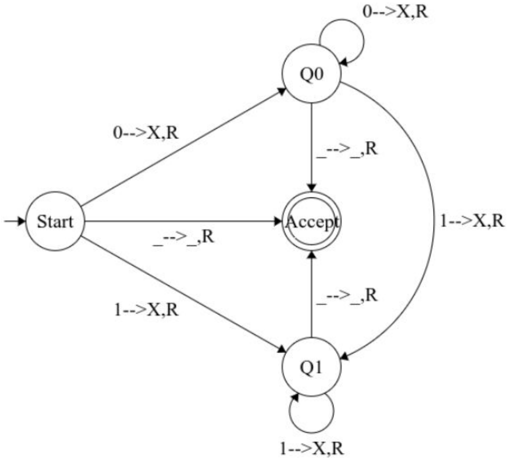


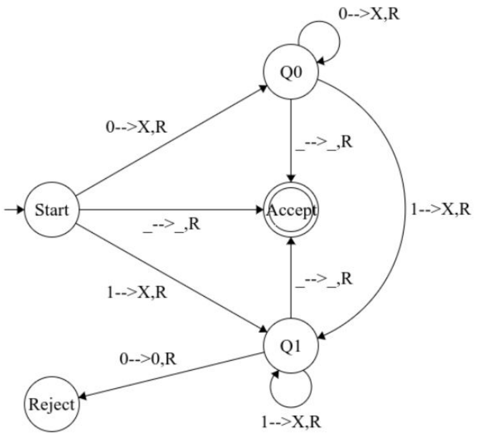
Examples for TM : 0^*1^*

- For the finite automaton of the Turing Machine :

X	X	X	X	X	X	X	X	X	␣

If input string is : 00…010011..11

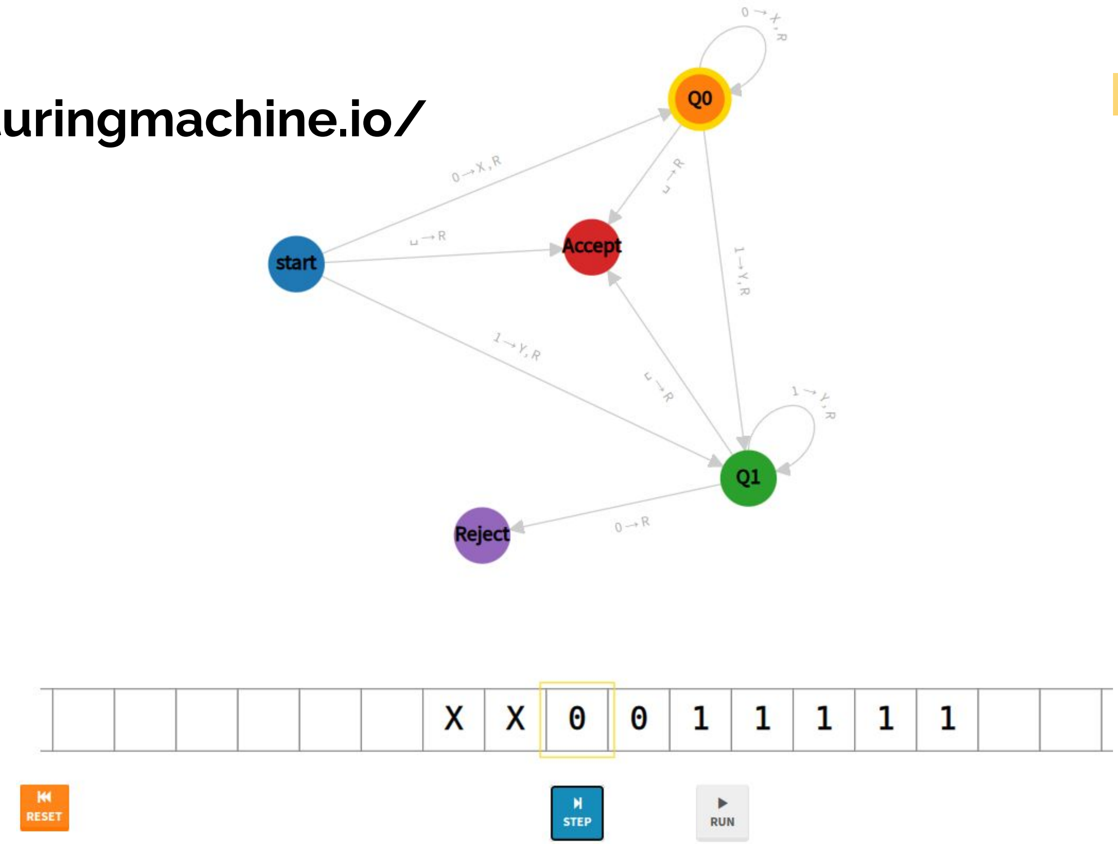




Examples for TM : 0^*1^*

Online Simulator : **https://turingmachine.io/**

```
input: '000011111'
blank: ' '
start state: start
table:
  start:
    0: {write: X, R: Q0}
    1: {write: Y, R: Q1}
    ' ': {R: Accept}
  Q0:
    0: {write: X, R: Q0}
    1: {write: Y, R: Q1}
    ' ': {R: Accept}
  Q1:
    1: {write: Y, R: Q1}
    0: {R: Reject}
    ' ': {R: Accept}
  Accept:
  Reject:
```

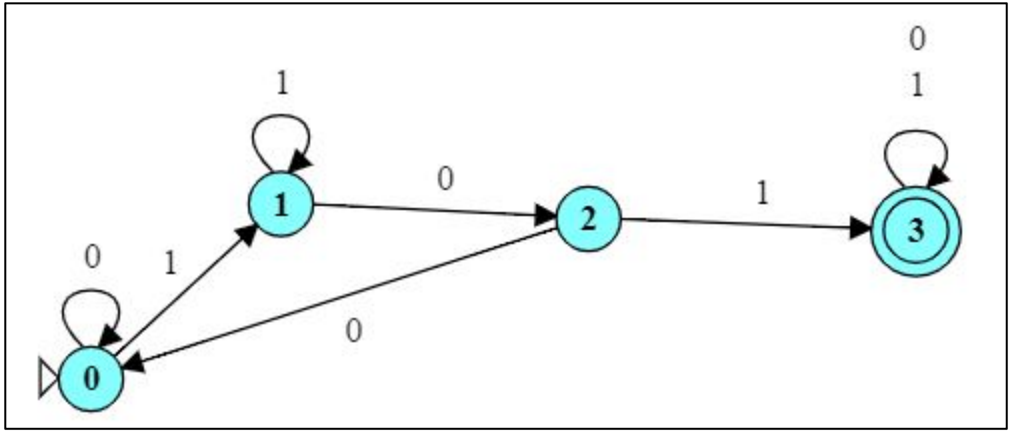


Examples for TM : Contains 101

- The Deterministic Finite Automaton for the language:

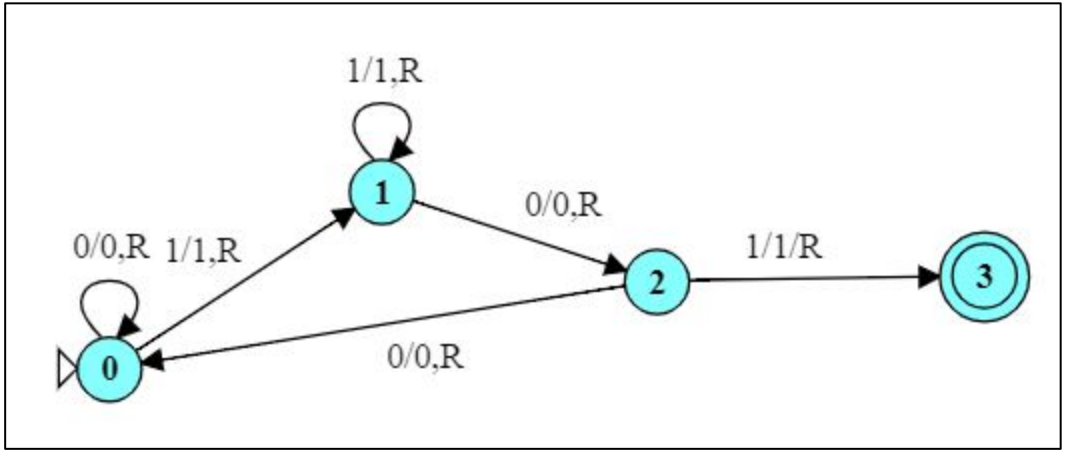
Examples for TM : Contains 101

- The Deterministic Finite Automaton for the language:



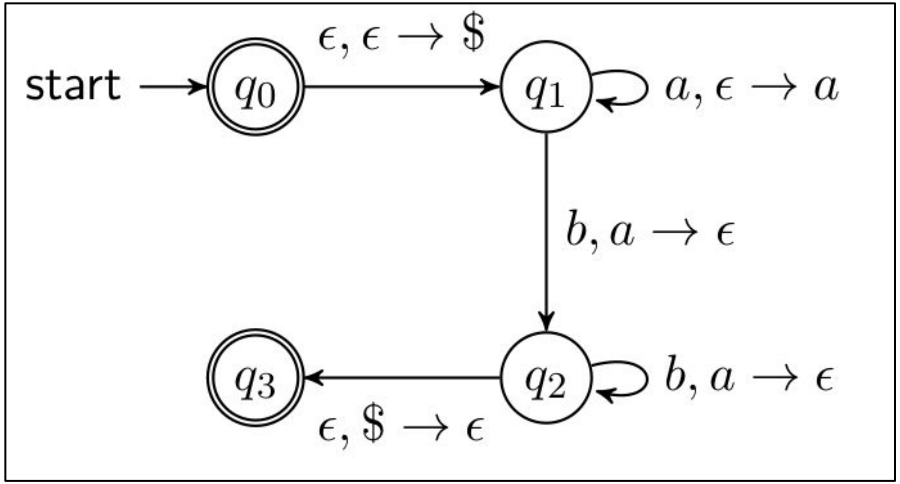
Examples for TM : Contains 101

- The Turing Machine for the language is given as:



Examples for TM : 0^n1^n

- The Pushdown Automaton for the language is given as
    - PDA uses a Stack.

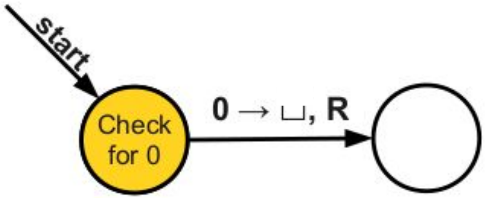


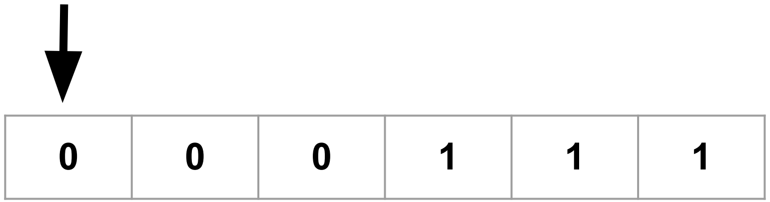
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - ?
- Some basic rules:
  - The string ε is in L.
  - Any string starting with 1 is not in L.
  - Any string ending with 0 is not in L.

Examples for TM : 0^n1^n

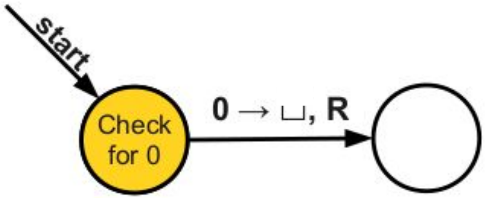
- The Algorithm for using Turing Machine :
    - The **initial** zero found, we change it to blank

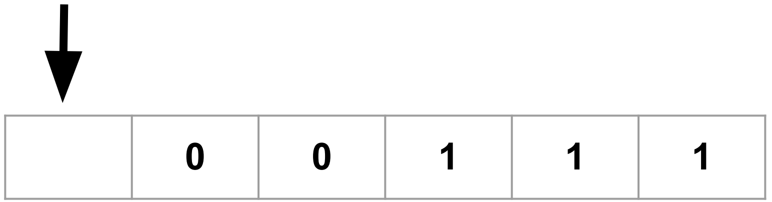




Examples for TM : 0^n1^n

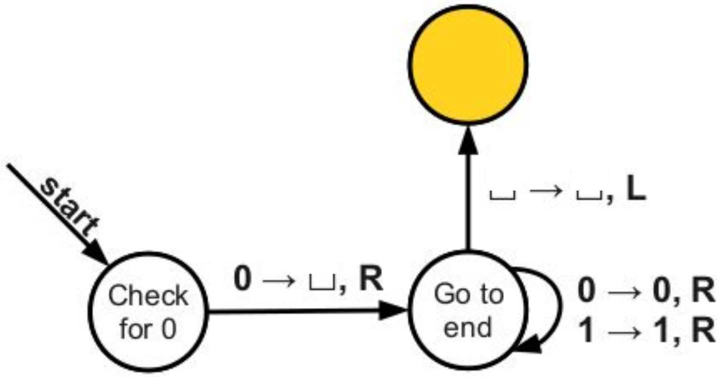
- The Algorithm for using Turing Machine :
  - The **initial** zero found, we change it to blank

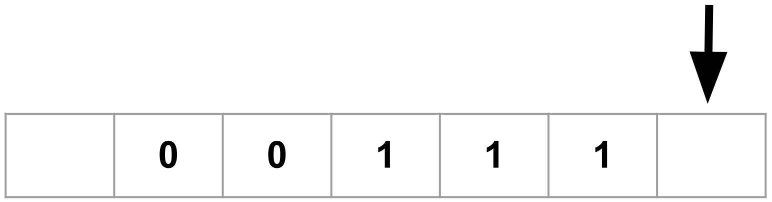




Examples for TM : 0^n1^n

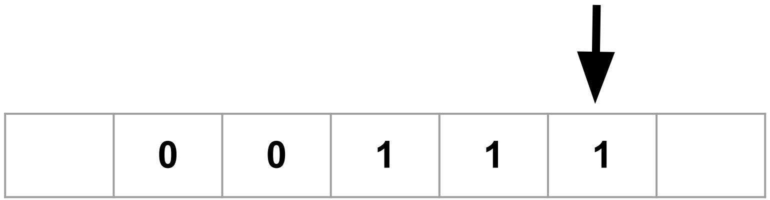
- The Algorithm for using Turing Machine :
  - We search by **skipping all zeros and ones until we reach blank at the extreme right**





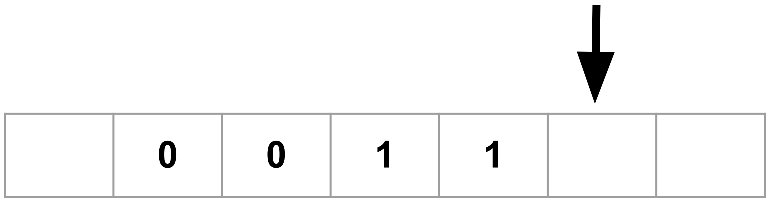
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
    - We search by **skipping all zeros and ones until we reach blank at the extreme right**
    - We move left to the one



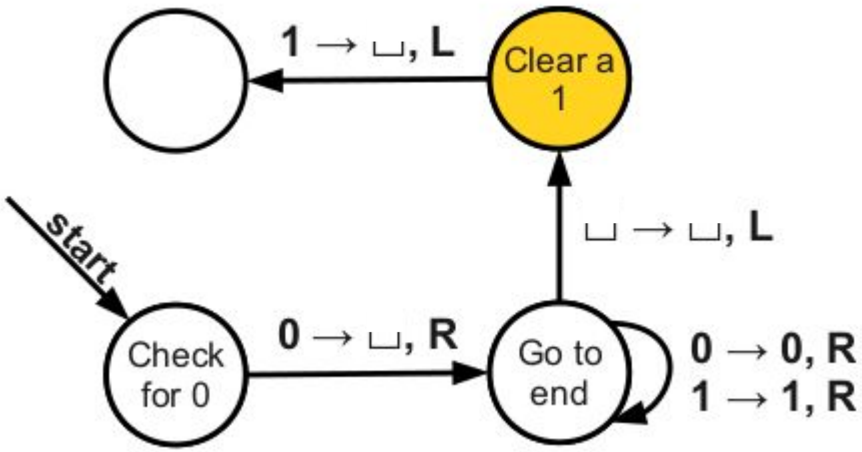
Examples for TM : 0^n1^n

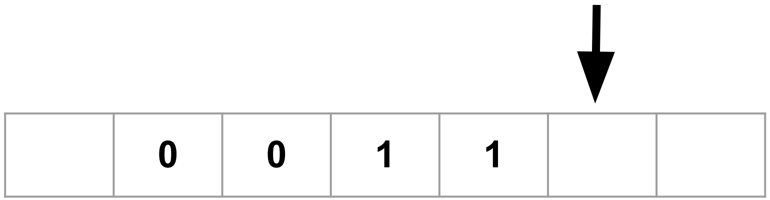
- The Algorithm for using Turing Machine :
  - We search by **skipping all zeros and ones until we reach blank at the extreme right**
  - We move left to the one
  - We replace it with Blank



Examples for TM : 0^n1^n

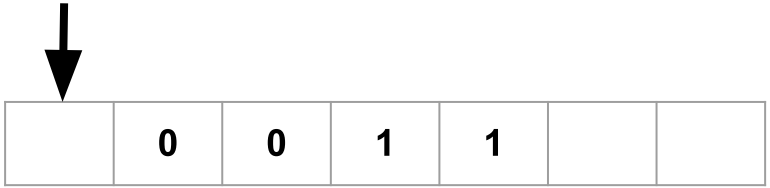
- The Algorithm for using Turing Machine :





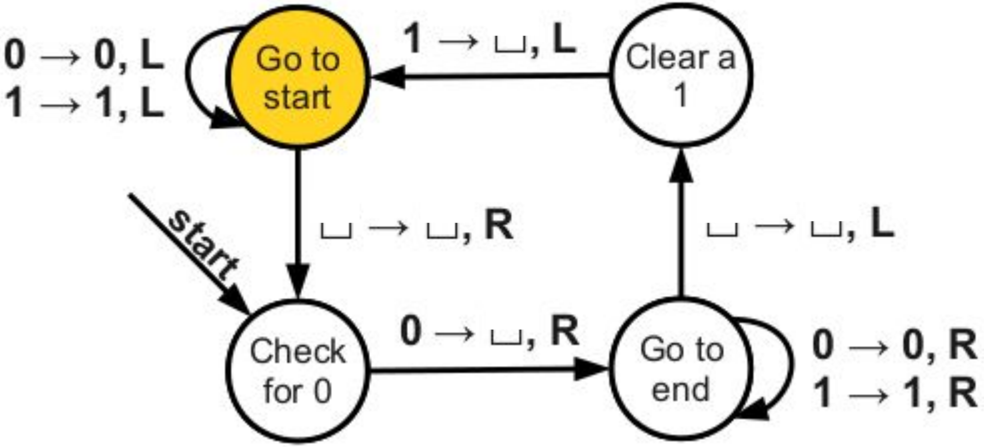
Examples for TM : 0^n1^n

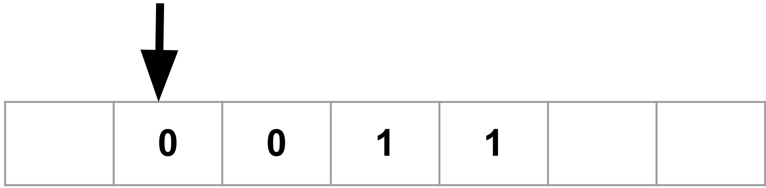
- The Algorithm for using Turing Machine :
  - We search by **skipping all zeros and ones until we reach blank at the extreme LEFT where you can move right**



Examples for TM : 0^n1^n

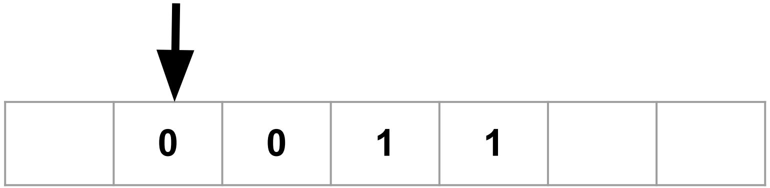
- The Algorithm for using Turing Machine





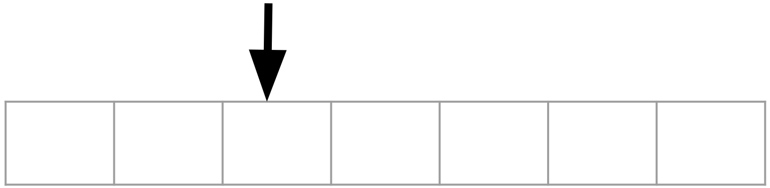
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - We search by **skipping all zeros and ones until we reach blank at the extreme LEFT where you can move right**
  - If at current cell with zero, recursively....



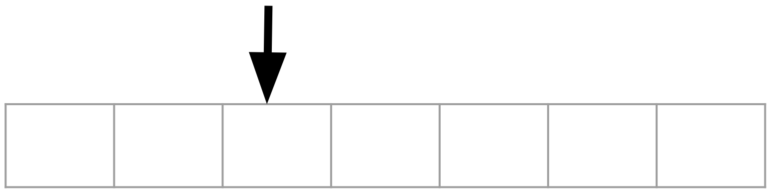
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - When to Stop ? Let's assume for an Accept:



Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - When to Stop ? Let’s assume for an Accept:
    - When there is a blank during the start state



Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - When to say whether a word is accepted ?
    - By DFA/NFA
    - By PDA
    - By TM

Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - When to say whether a word is accepted ?
    - By DFA/NFA
    - By PDA
    - By TM

1. You reach an accept state
2. You read all letters in the tape (given string)

Examples for TM : 0^n1^n

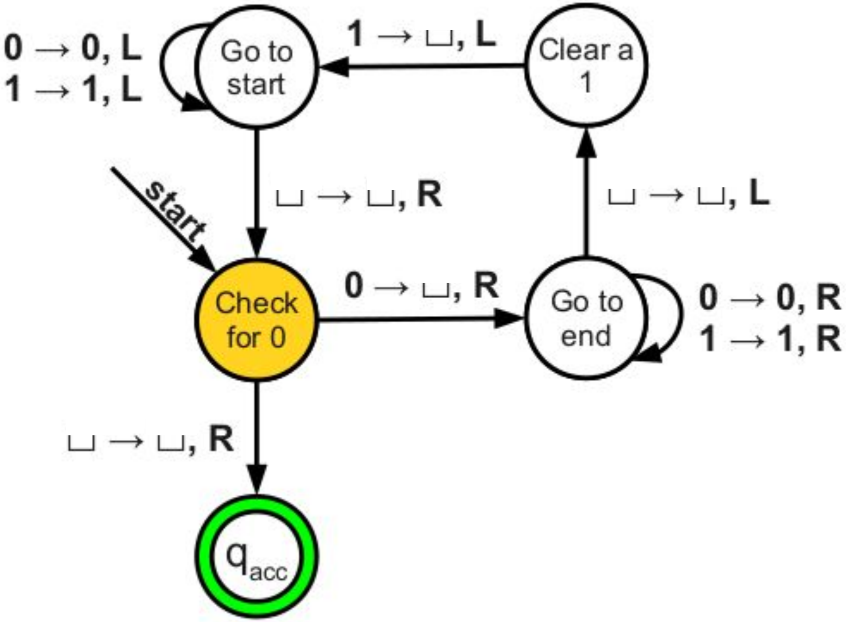
- The Algorithm for using Turing Machine :
  - When to say whether a word is accepted ?
    - By DFA/NFA
    - By PDA
    - By TM

1. You reach an accept state
2. You read all letters in the tape (given string)

1. You reach an accept state

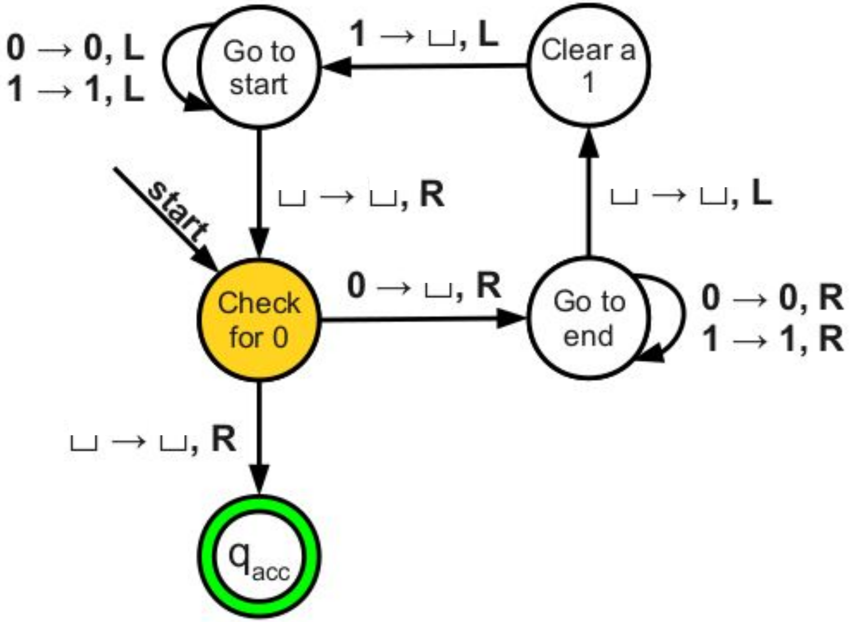
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :



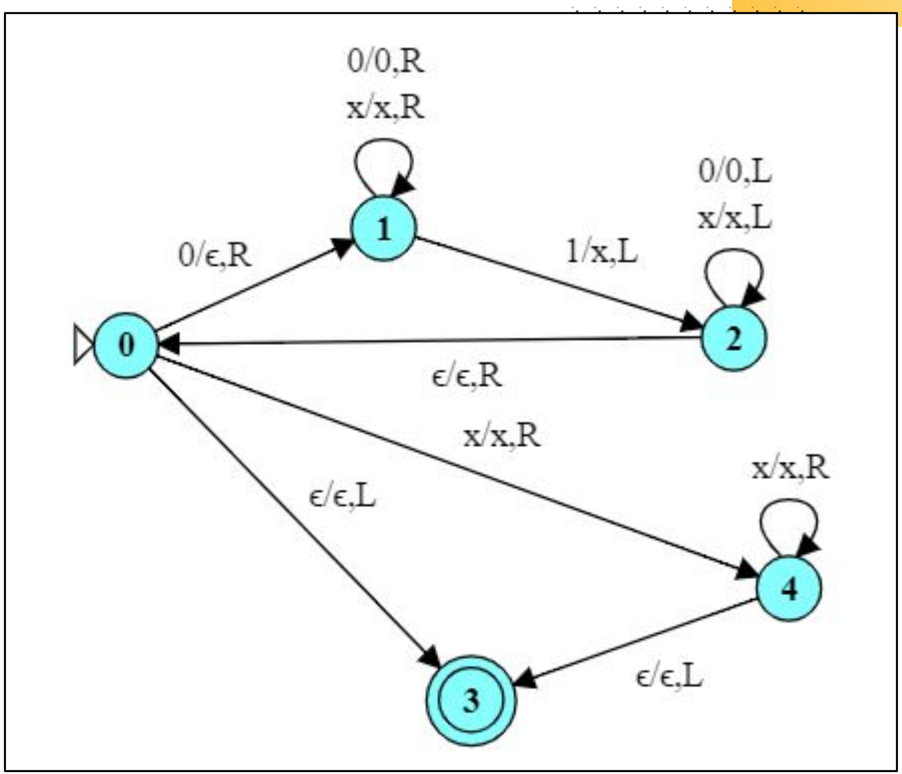
Examples for TM : 0^n1^n

- The Algorithm for using Turing Machine :
  - Following words:
    - 1
    - 01111
    - 001
  - Reject state and transitions are implicit



Examples for TM : 0^n1^n

- Another Possible solution

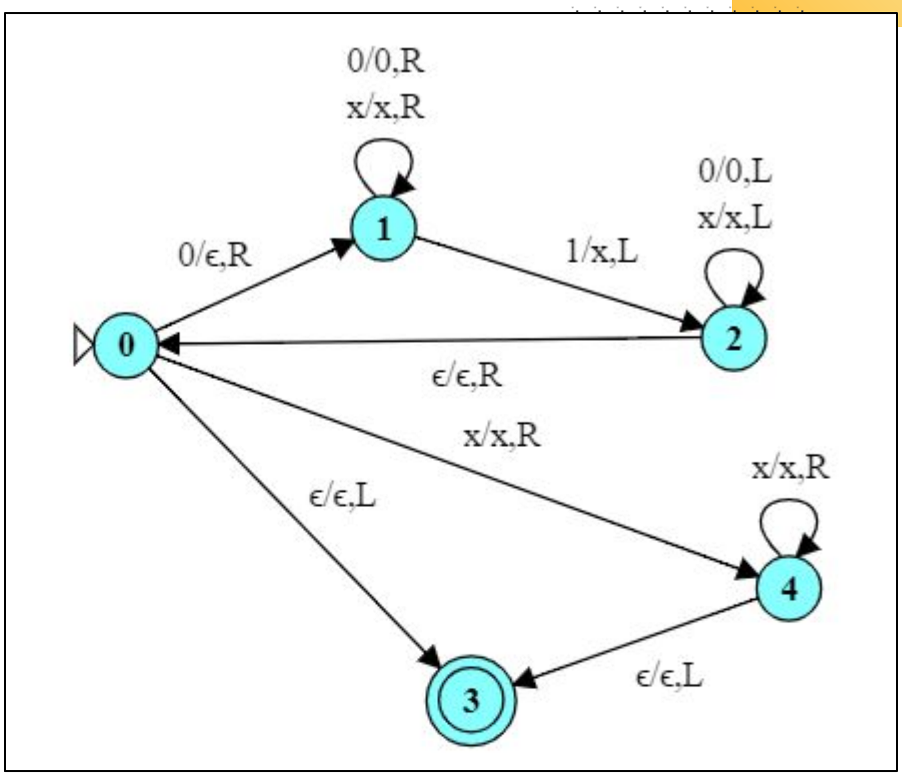


Examples for TM : 0^n1^n

- Another Possible solution

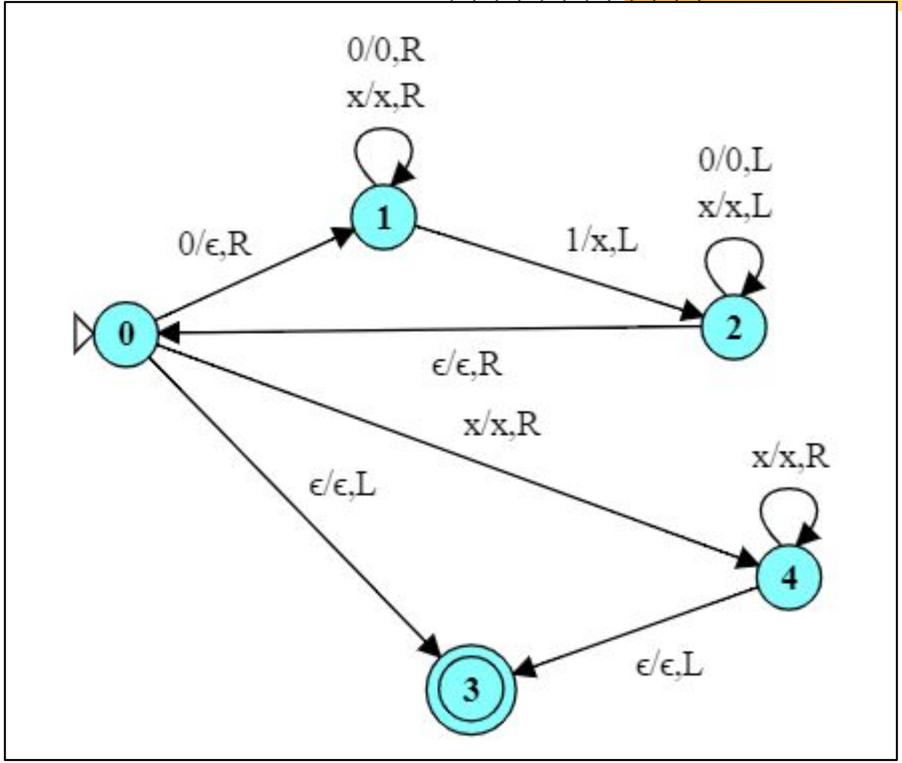
Don't be confused with the Epsilon of PDA/DFA

Here means the tape cell is blank...



Examples for TM : 0^n1^n

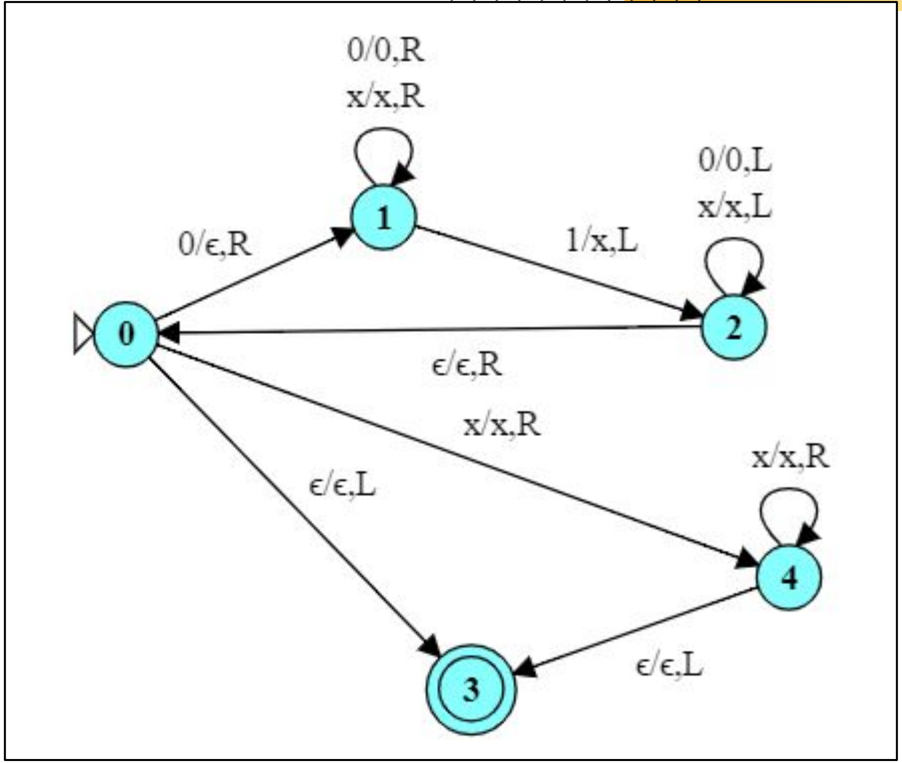
- Is the following word accepted :
  - **01111**
- Notations:
  - 0/x,R : when you read 0, replace it with X and move Right.



Examples for TM : 0^n1^n

- Is the following word accepted : **0011**

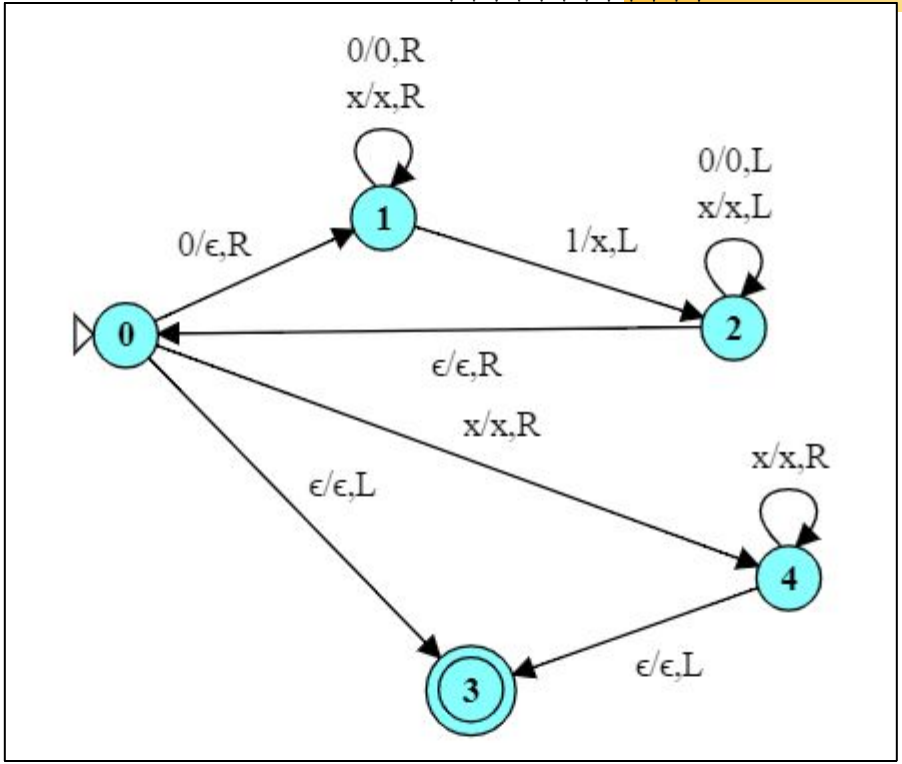
Step	State	0	0	1	1
0	0	O	0	1	1



Examples for TM : 0^n1^n

- Is the following word accepted : **0011**

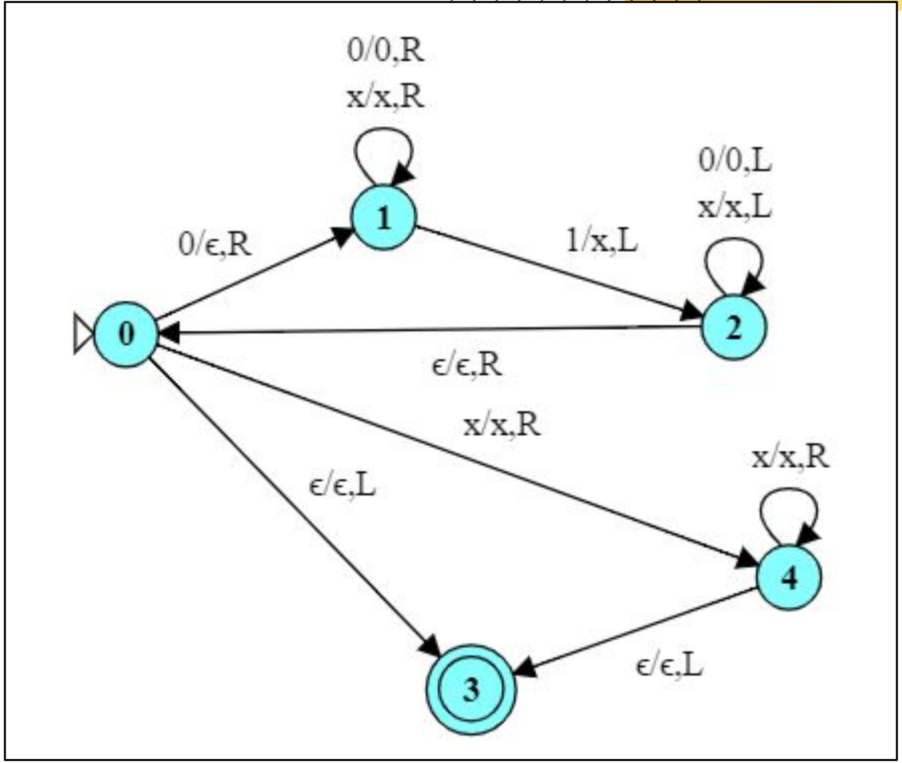
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	␣	0	1	1



Examples for TM : 0^n1^n

- Is the following word accepted : **0011**

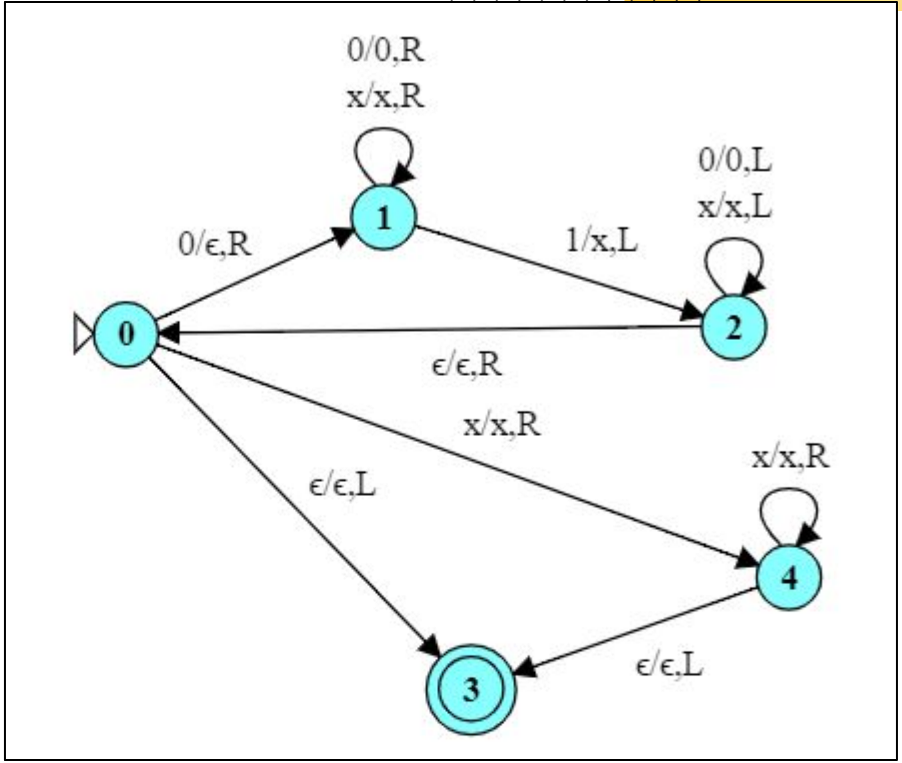
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	␣	0	1	1
2	1	␣	0	1	1



Examples for TM : 0^n1^n

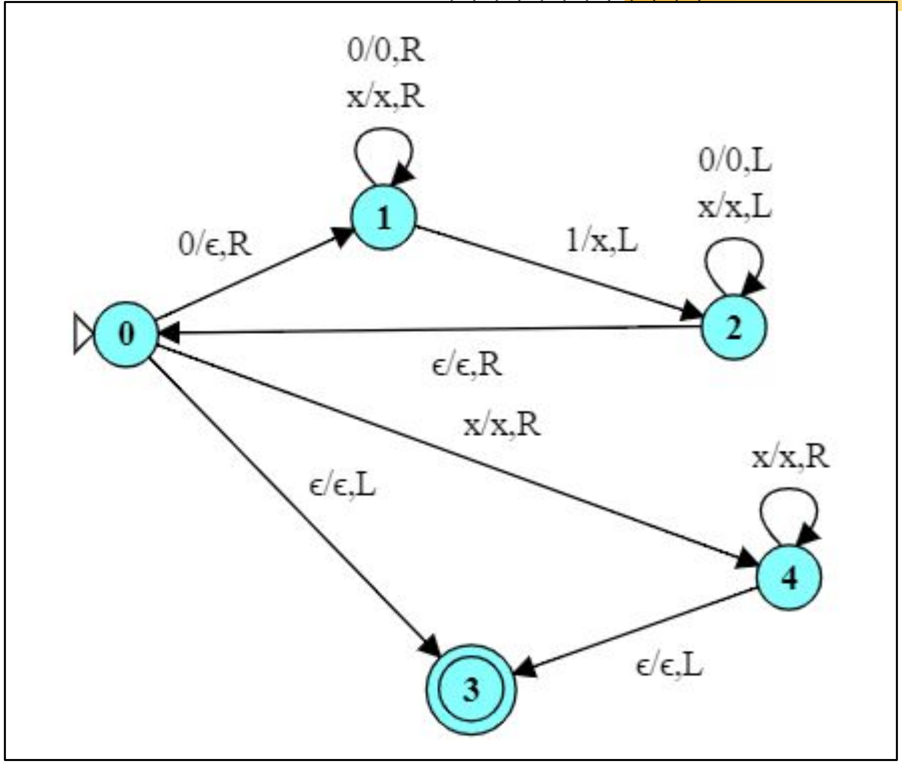
- Is the following word accepted : **0011**

Step	State	0	0	1	1
0	0	O	0	1	1
1	1	␣	0	1	1
2	1	␣	0	1	1
3	2	␣	0	x	1



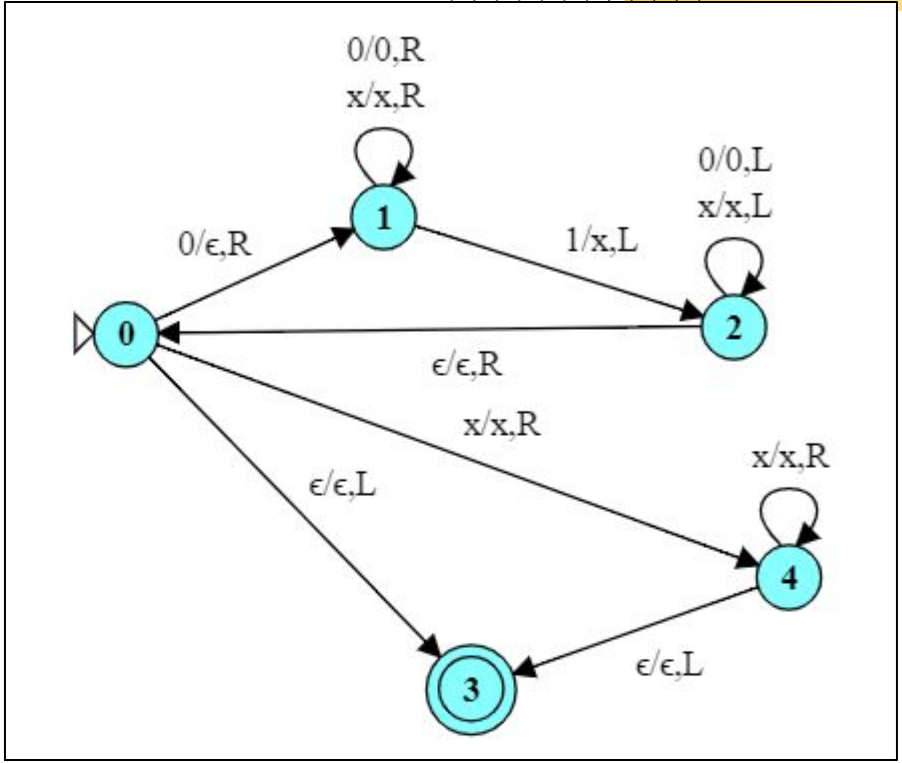
Examples for TM : $0^n 1^n$

Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1



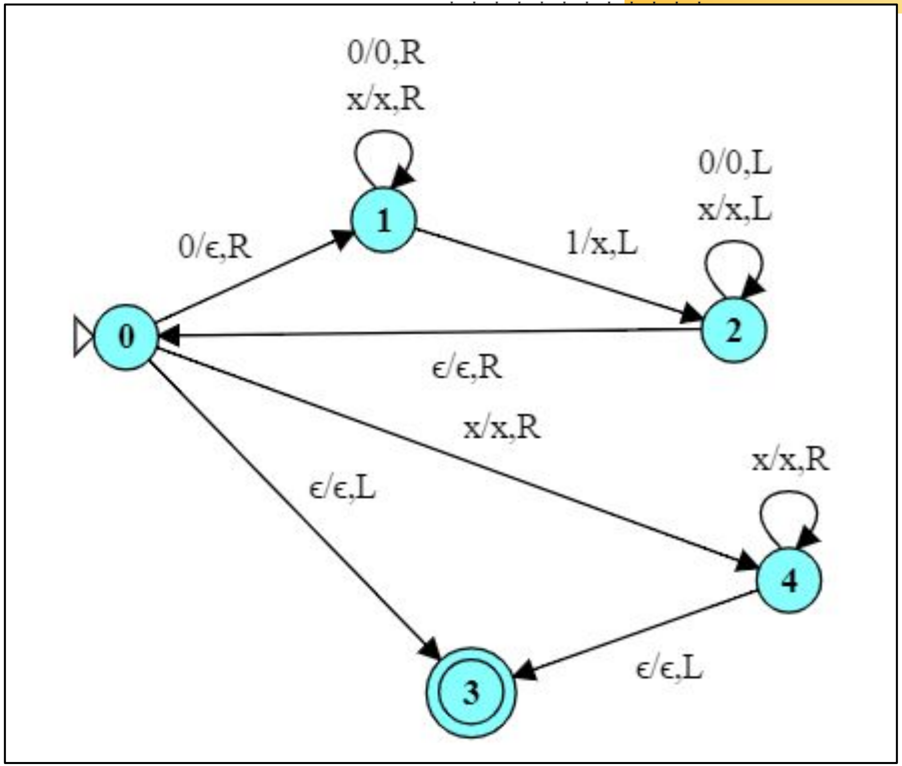
Examples for TM : 0^n1^n

Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1

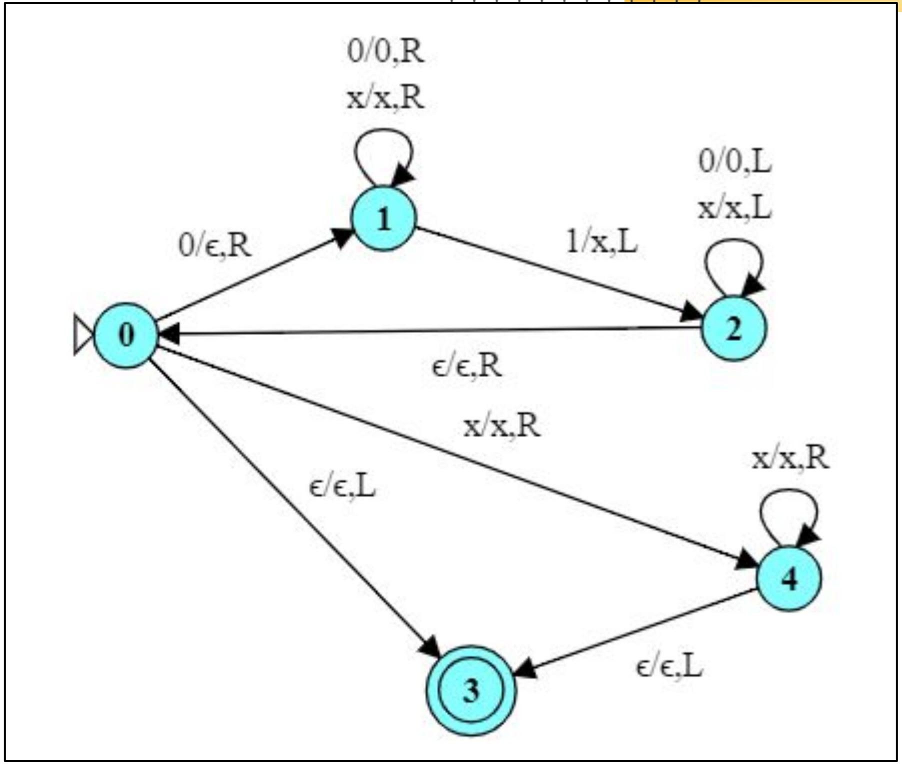


Examples for TM : 0^n1^n

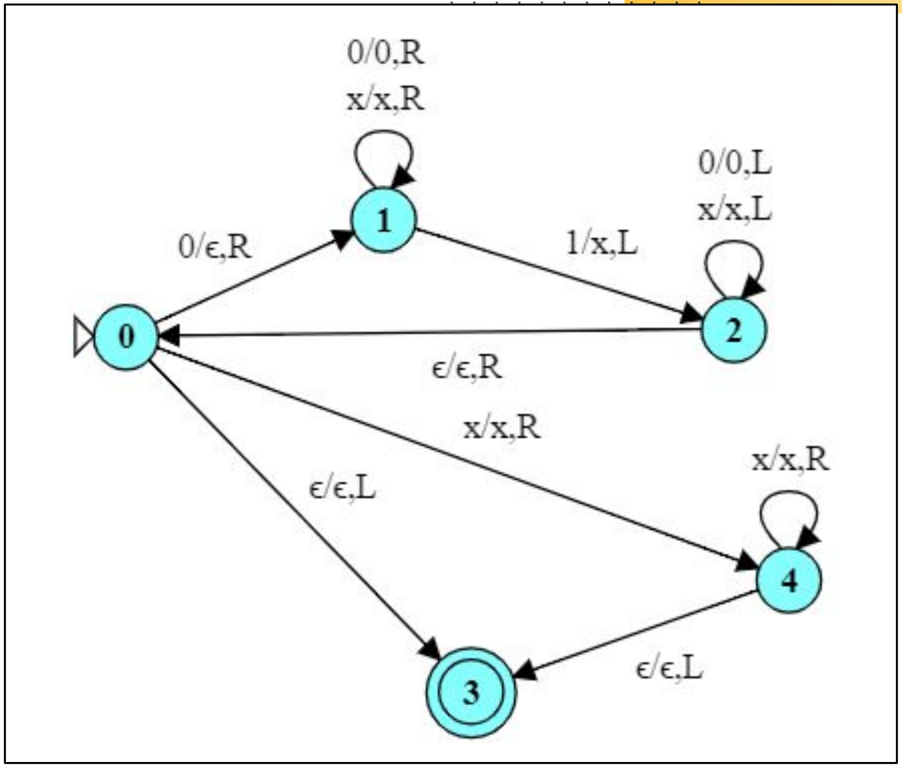
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1



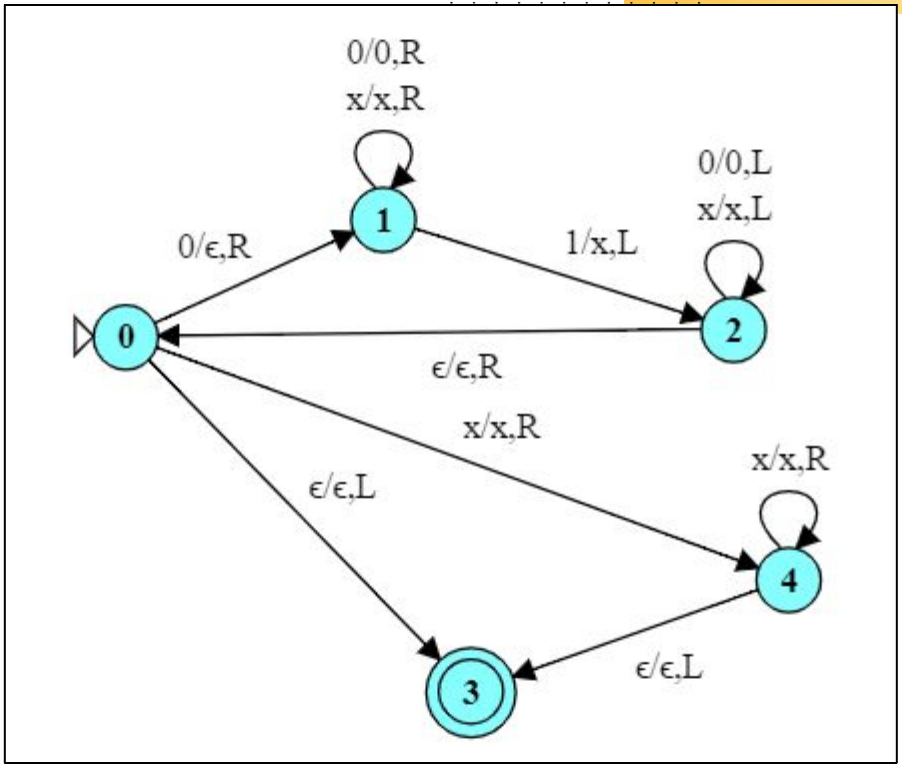
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1
7	1	⊔	⊔	x	1



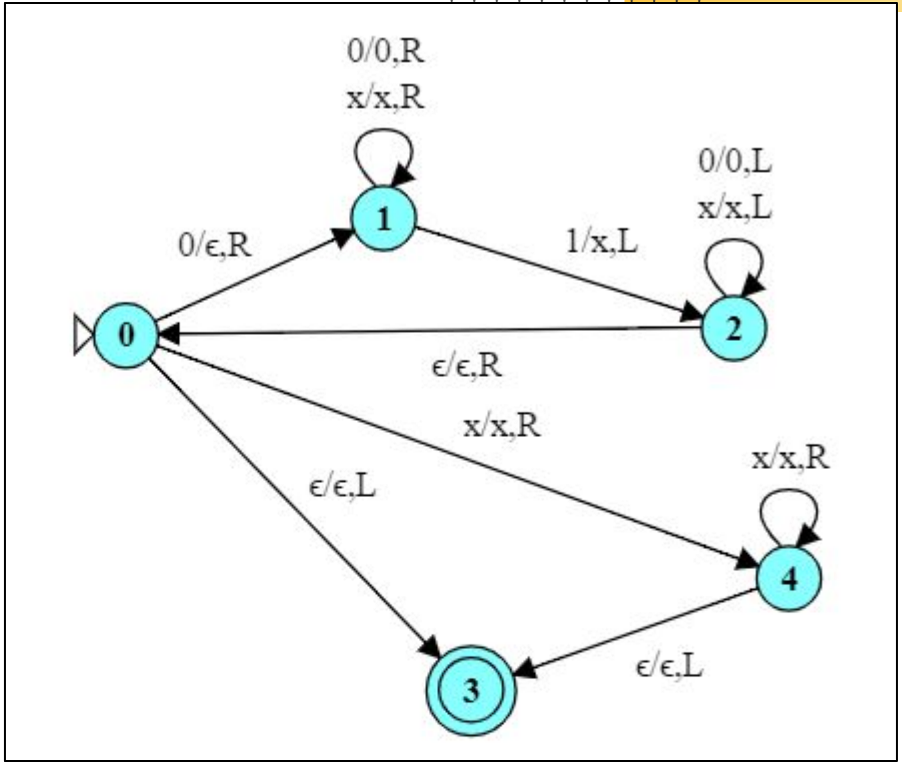
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1
7	1	⊔	⊔	x	1
8	2	⊔	⊔	x	x



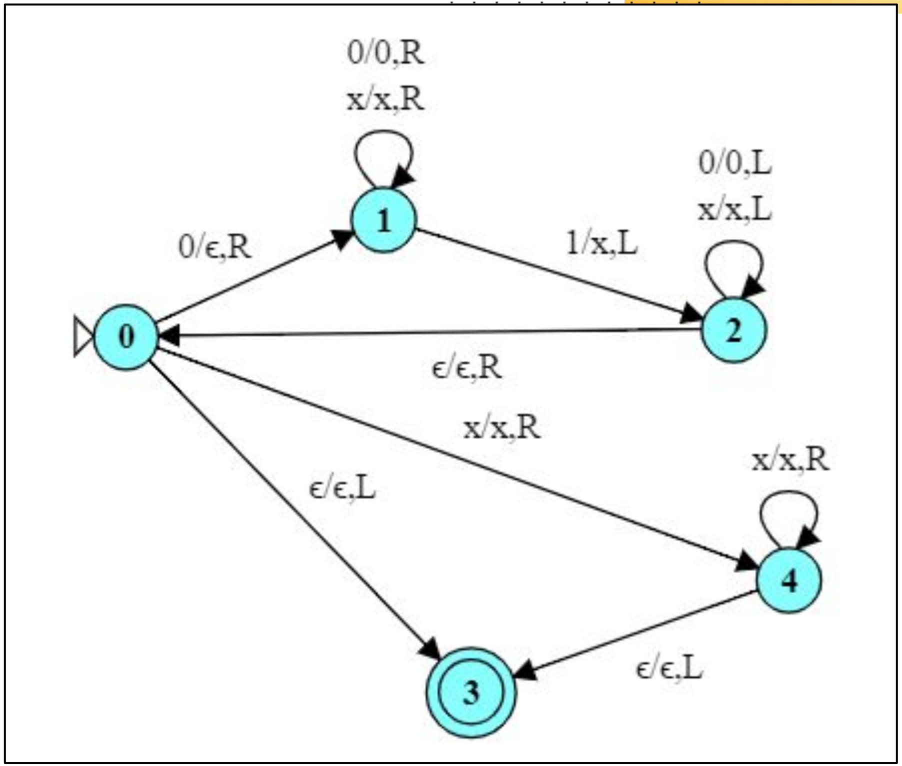
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1
7	1	⊔	⊔	x	1
8	2	⊔	⊔	x	x
9	2	⊔	⊔	x	x



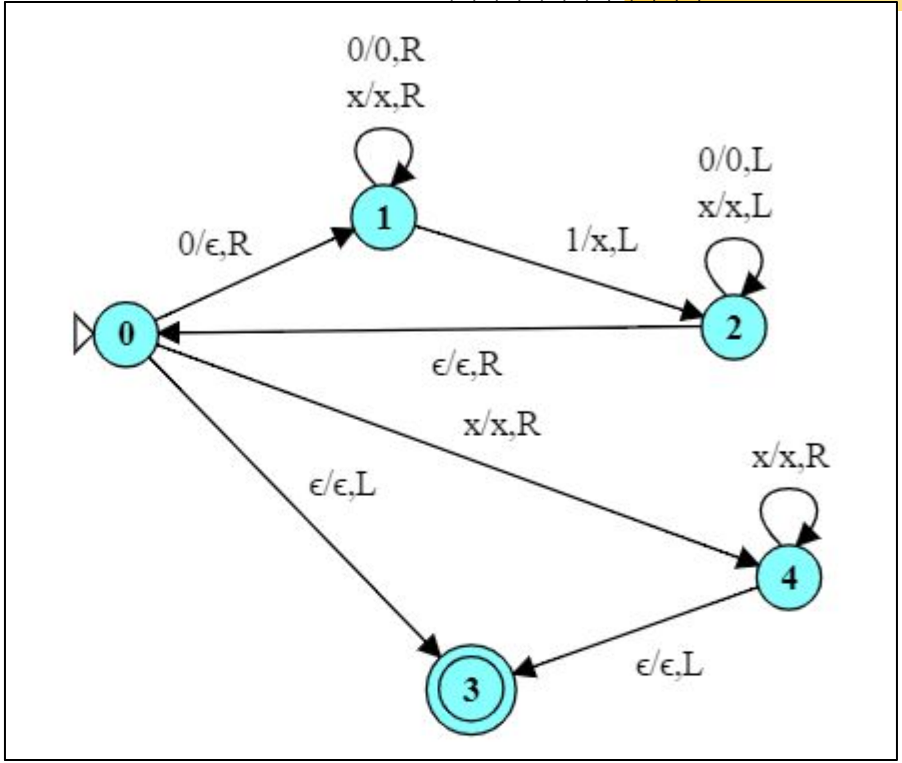
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1
7	1	⊔	⊔	x	1
8	2	⊔	⊔	x	x
9	2	⊔	⊔	x	x
10	0	⊔	⊔	x	x



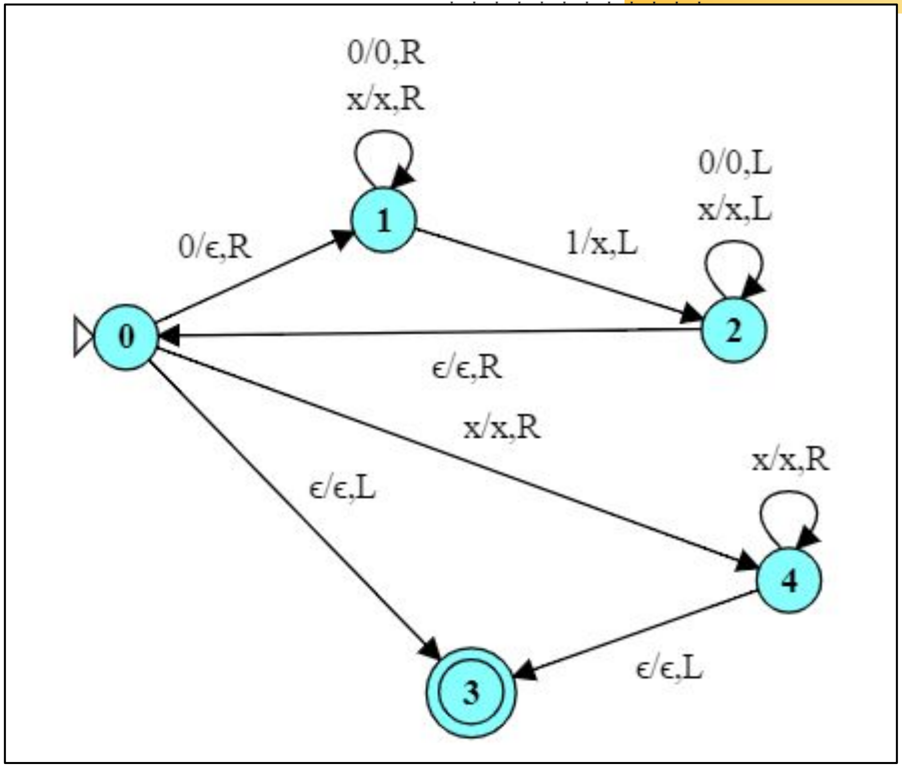
Step	State	0	0	1	1
0	0	O	0	1	1
1	1	⊔	0	1	1
2	1	⊔	0	1	1
3	2	⊔	0	x	1
4	2	⊔	0	x	1
5	0	⊔	0	x	1
6	1	⊔	⊔	x	1
7	1	⊔	⊔	x	1
8	2	⊔	⊔	x	x
9	2	⊔	⊔	x	x
10	0	⊔	⊔	x	x
11	4	⊔	⊔	x	x



Step	State	0	0	1	1	
0	0	O	0	1	1	
1	1	⊔	0	1	1	
2	1	⊔	0	1	1	
3	2	⊔	0	x	1	
4	2	⊔	0	x	1	
5	0	⊔	0	x	1	
6	1	⊔	⊔	x	1	
7	1	⊔	⊔	x	1	
8	2	⊔	⊔	x	x	
9	2	⊔	⊔	x	x	
10	0	⊔	⊔	x	x	
11	4	⊔	⊔	x	x	
12	4	⊔	⊔	x	x	

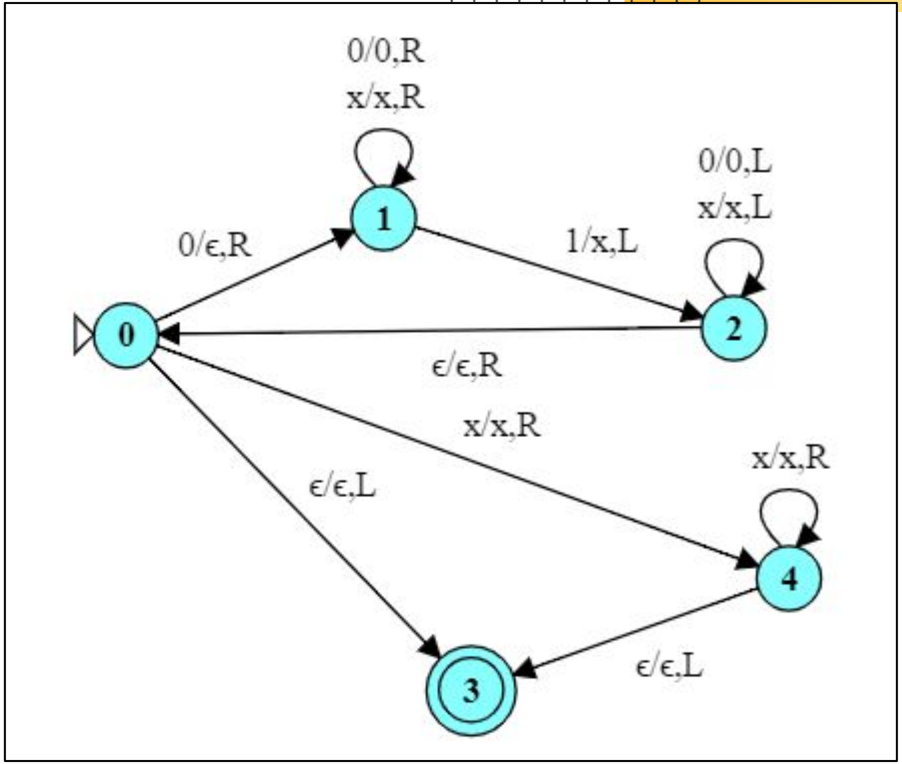


Step	State	0	0	1	1	
0	0	O	0	1	1	
1	1	⊔	0	1	1	
2	1	⊔	0	1	1	
3	2	⊔	0	x	1	
4	2	⊔	0	x	1	
5	0	⊔	0	x	1	
6	1	⊔	⊔	x	1	
7	1	⊔	⊔	x	1	
8	2	⊔	⊔	x	x	
9	2	⊔	⊔	x	x	
10	0	⊔	⊔	x	x	
11	4	⊔	⊔	x	x	
12	4	⊔	⊔	x	x	
13	3	⊔	⊔	x	x	



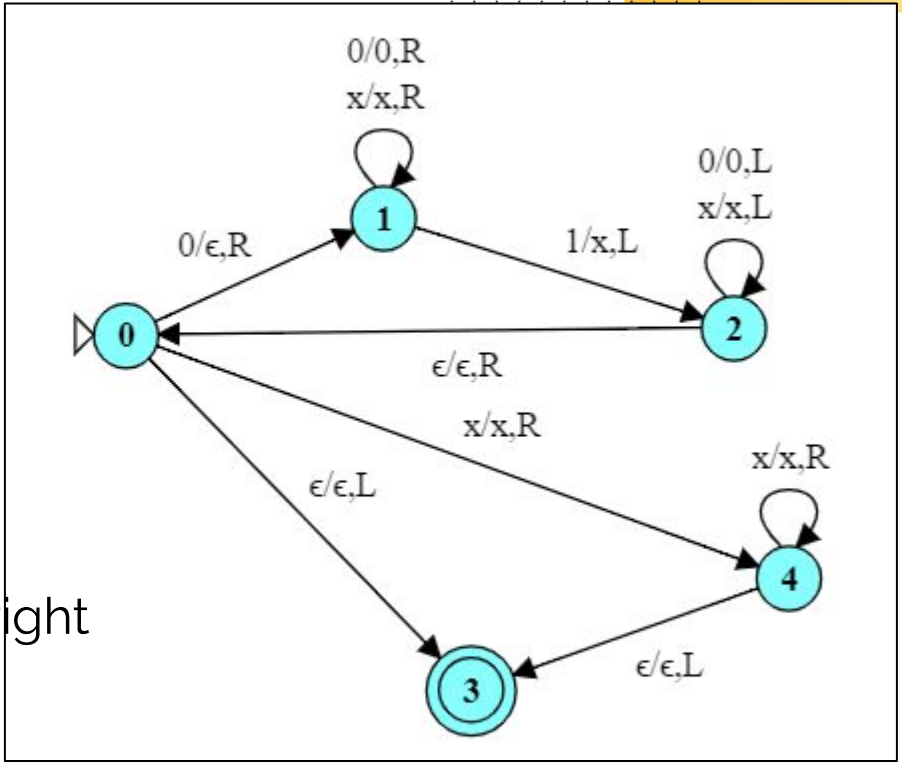
Examples for TM : 0^n1^n

- What's the Algorithm ?



Examples for TM : 0^n1^n

- What's the Algorithm :
  - Initial zero replace by blank
  - Skip right x and 0 till you find first 1
  - Replace 1 by x and move left
  - Skip left 0 and x until blank, move right
  - If no only x or spaces, accept.



Examples for TM : $a^nb^nc^n$

- This language is :
    - Not Regular, therefore, we cannot create the DFA
    - Not Context Free, therefore we cannot create a Pushdown Automaton
    - But, we can create the Turing Machine for this Language

Examples for TM : $a^n b^n c^n$

- The Algorithm :

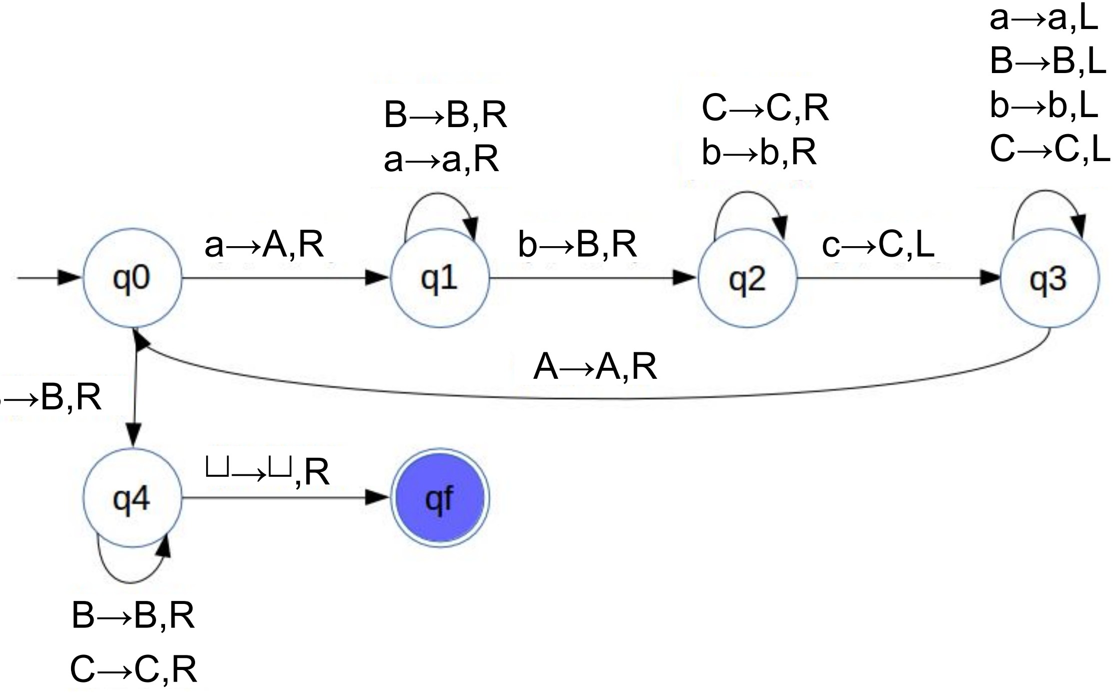
Examples for TM : $a^n b^n c^n$

- The Algorithm :
  - Replace a with A, skip right all a to find first b.
  - Replace b with B, skip right all b to find first c.
  - Replace c with C, Skip LEFT all a,A,b,B,C until A is found.
  - Keep repeating the process.

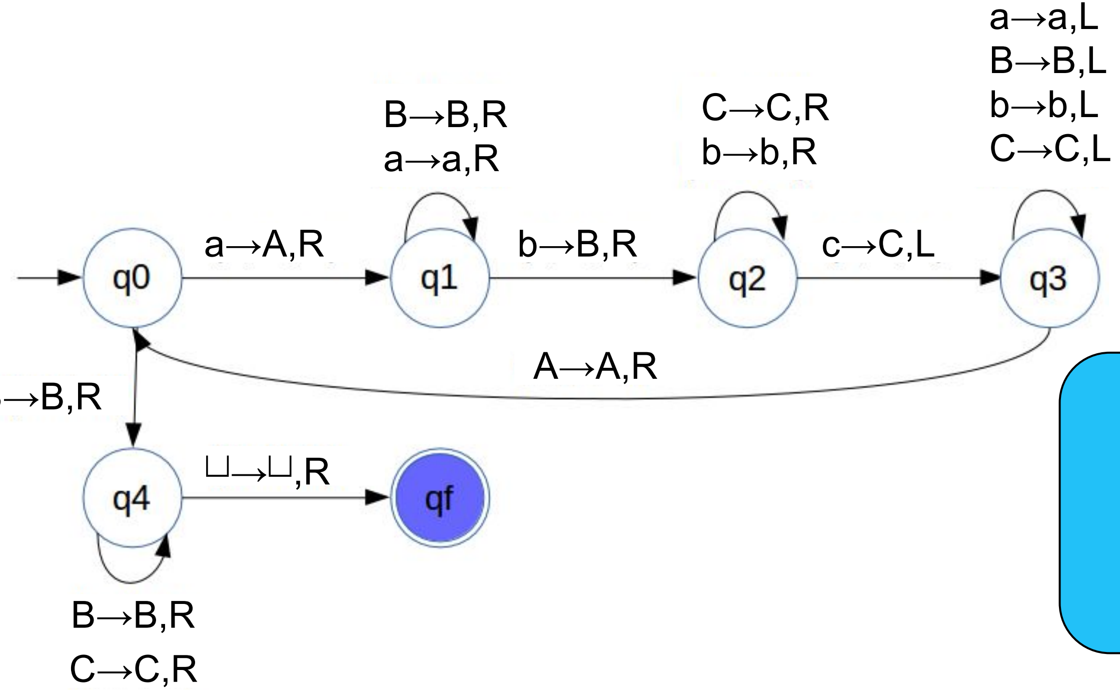
Examples for TM : $a^n b^n c^n$

- The Algorithm :
  - Replace a with A, skip right all a to find first b.
  - Replace b with B, skip right all b to find first c.
  - Replace c with C, Skip LEFT all a,A,b,B,C until A is found.
  - Keep repeating the process.
    - Until all letters are : A, B, and C on the tape.

Examples for TM : $a^n b^n c^n$



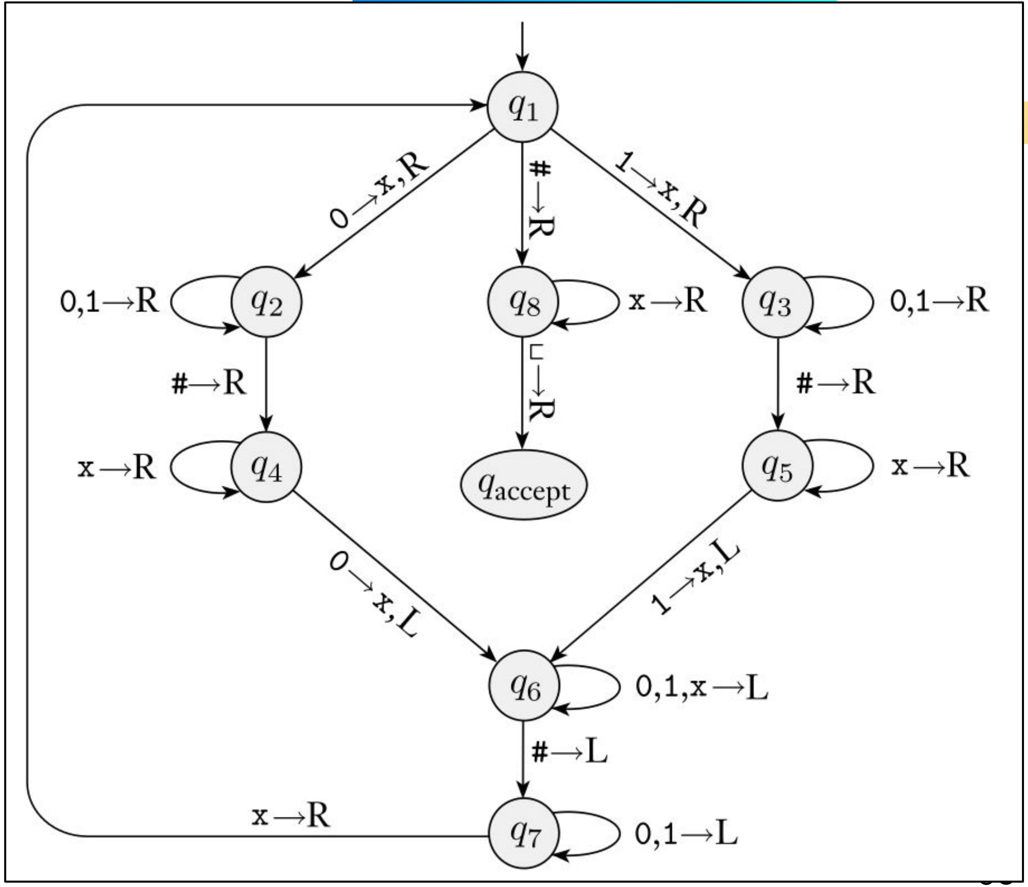
Examples for TM : $a^n b^n c^n$



Is the empty string accepted ?

Examples for TM : w#w

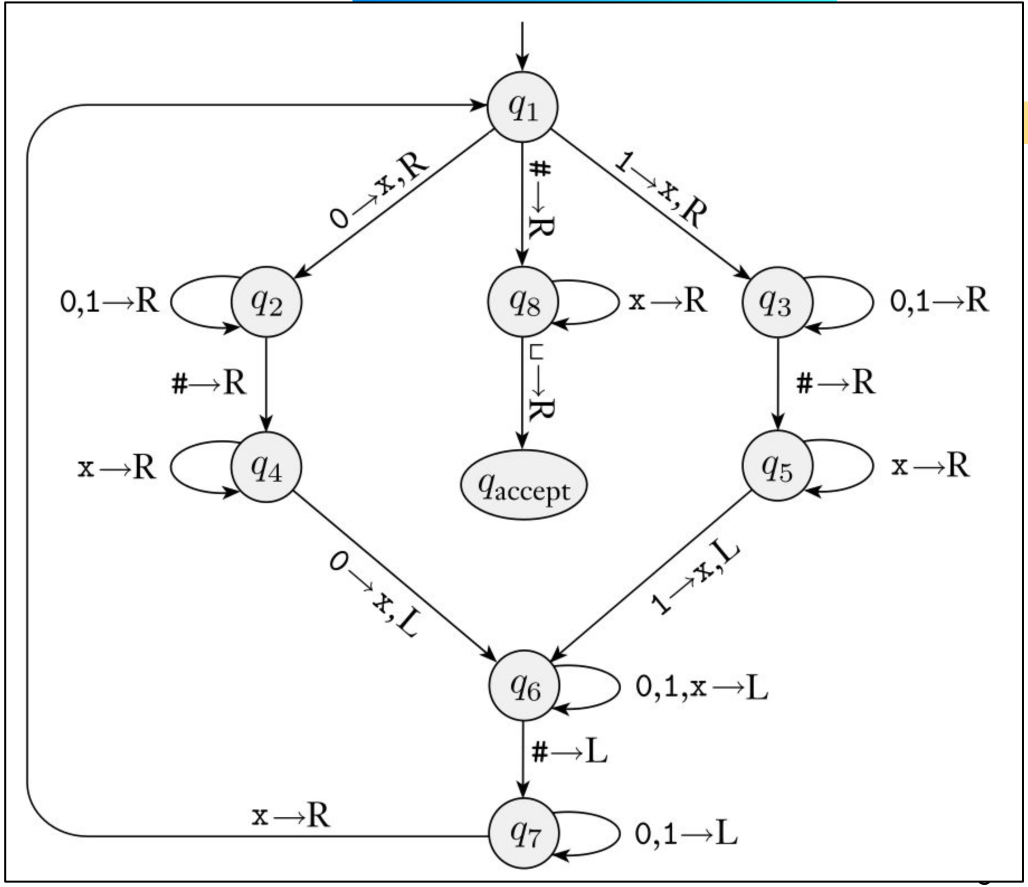
- What's the Algorithm for this ?



Examples for TM : w#w

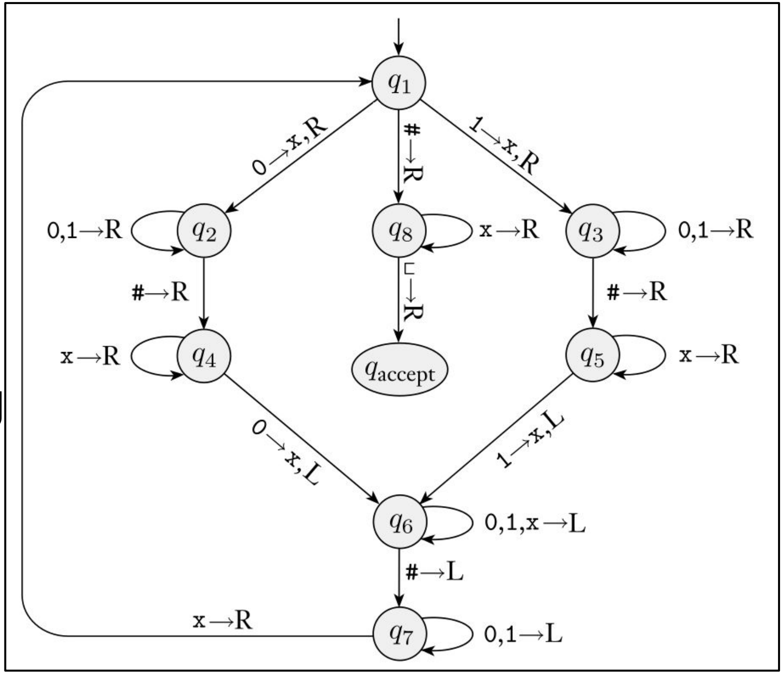
- What's the Algorithm for this ?

```
0 1 1 0 0 0 # 0 1 1 0 0 0 ␣ ...
x 1 1 0 0 0 # 0 1 1 0 0 0 ␣ ...
x 1 1 0 0 0 # x 1 1 0 0 0 ␣ ...
x 1 1 0 0 0 # x 1 1 0 0 0 ␣ ...
x x 1 0 0 0 # x 1 1 0 0 0 ␣ ...
x x x x x x # x x x x x x ␣ ...
                            accept
```



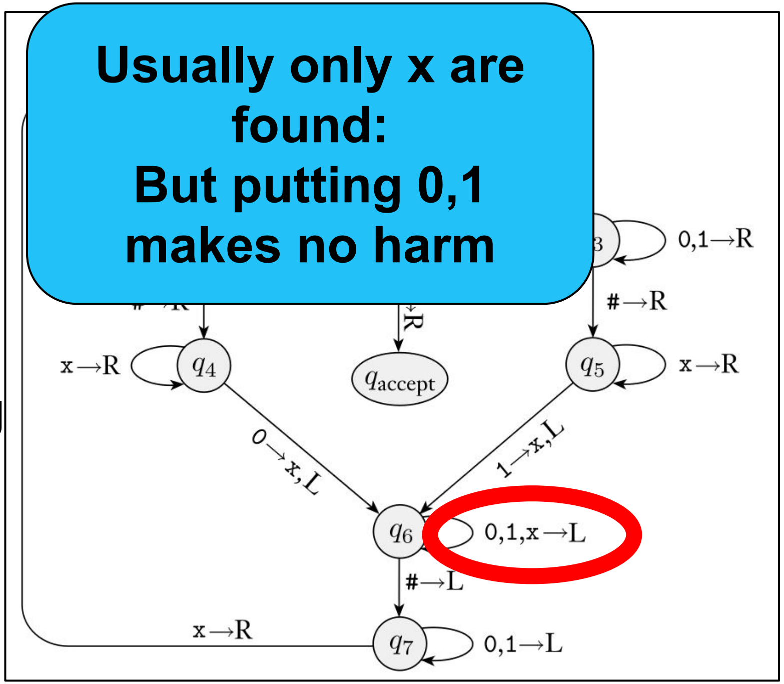
Examples for TM : w#w

- What's the Algorithm for this :
  - Initially:For a symbol (1 or 0) → x, R
  - Skip all 0 and 1 till #
  - Skip **only** x until the same initial symbol
  - Skip **LEFT** **ONLY** x until #, keep skipping 0 and 1
  - If x is found move right and go to initial



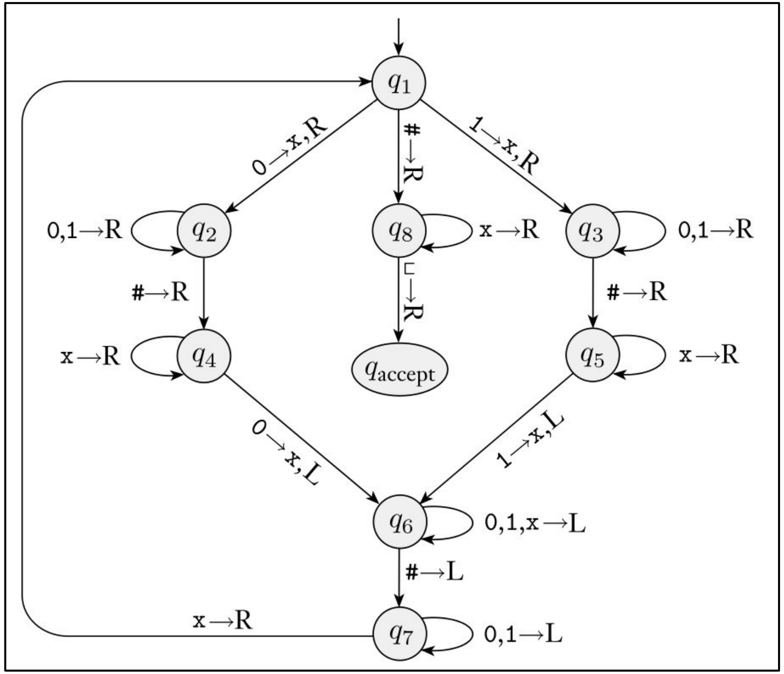
Examples for TM : w#w

- What's the Algorithm for this :
  - Initially:For a symbol (1 or 0) → x, R
  - Skip all 0 and 1 till #
  - Skip **only** x until the same initial symbol
  - Skip **LEFT** **<u>ONLY</u>** x until #, keep skipping 0 and 1
  - If x is found move right and go to initial



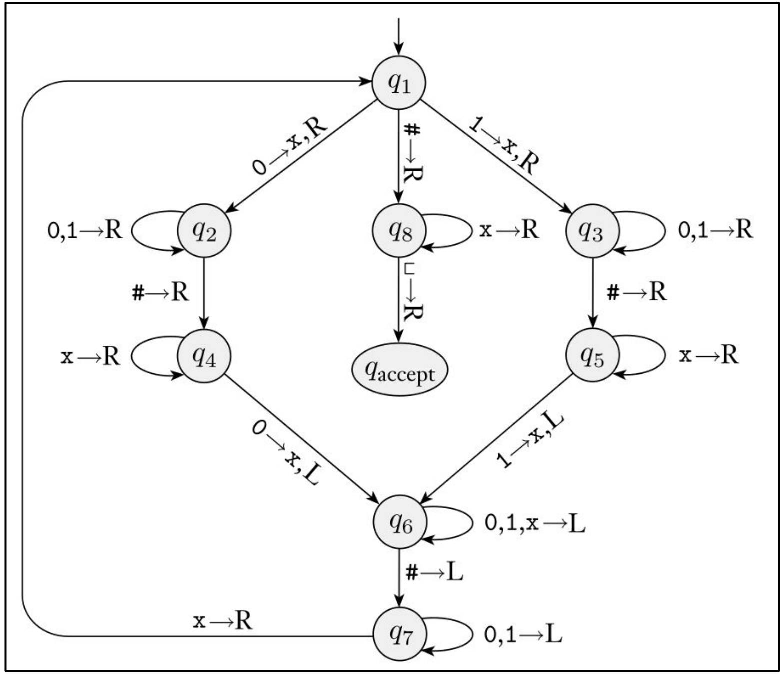
Examples for TM : w#w

- Btw: We are constructing only deterministic machine which is the norm
- There is nondeterministic Turing machine which has the same power as the normal TM.



Examples for TM : w#w

- Btw: We are constructing only deterministic machine which is the norm
- There is nondeterministic Turing machine which has the same power as the normal TM.



Examples for TM : ww

- How to construct the turing machine for this :

Examples for TM : ww

- How to construct the turing machine for this :

Algorithm for :
How to find the first word ?
Or
Middle of ww

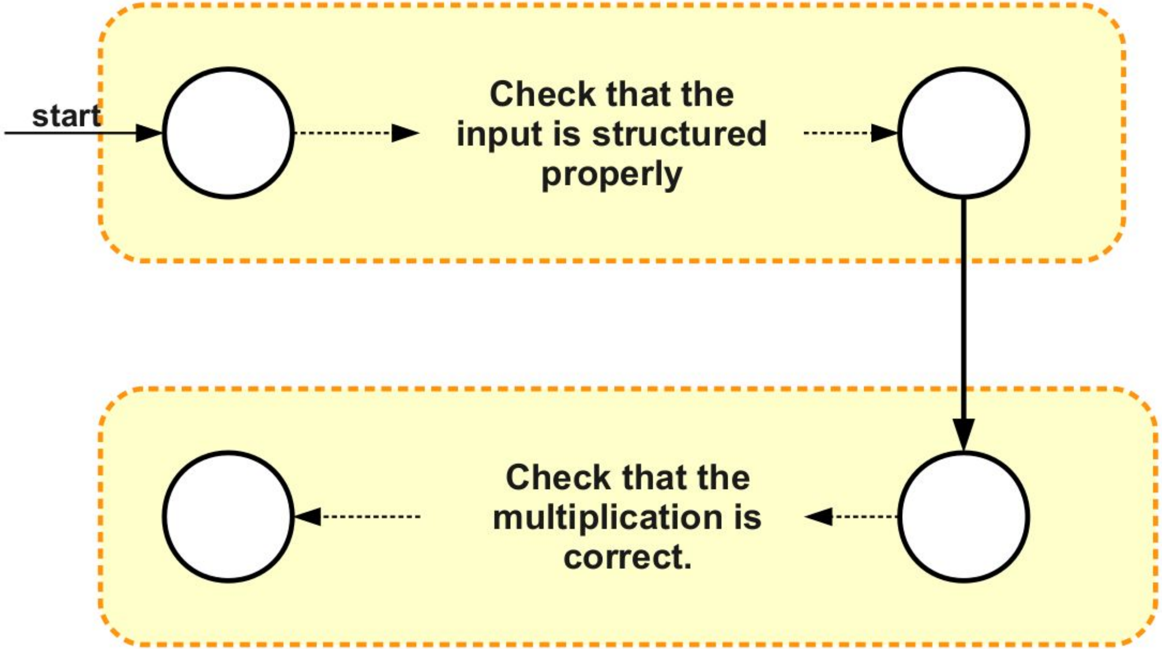
Examples for TM :
$1^n x 1^m = 1^{nm}$

- The language alphabet is : $\Sigma = \{1, \times, =\}$
- Is this :
  - Regular language ?
  - Context free language ?

Examples for TM : $1^n x 1^m = 1^{nm}$

- Remember : Turing Machine is an Abstract computer with the same power as a real computer.
  - Subroutines can be created as a set of states to perform some business logic. The set of states have a single entry state and single exit state
  - Complex tasks can be performed by breaking tasks into subroutines

Examples for TM :
$1^n x 1^m = 1^{nm}$

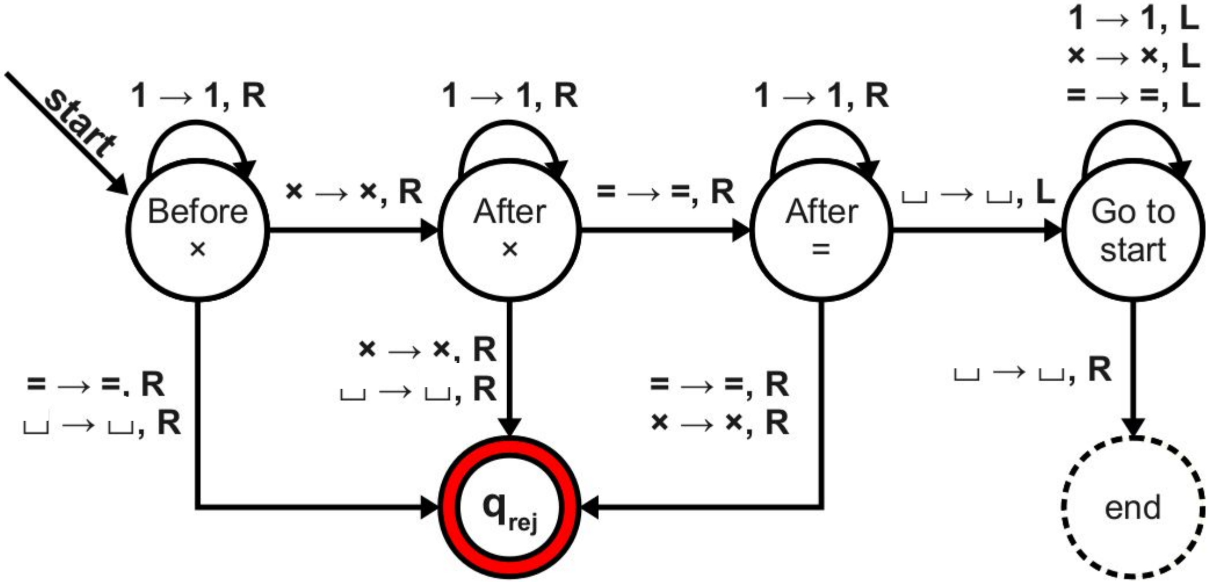


Examples for TM :
$1^n x 1^m = 1^{nm}$

- Let’s build a “**subroutine**” TM for the language $1^* x 1^* = 1^*$

Examples for TM : $1^n x 1^m = 1^{nm}$

- Let's build a "**subroutine**" TM for the language $1^*x1^*=1^*$



Examples for TM :
$1^n x 1^m = 1^{nm}$

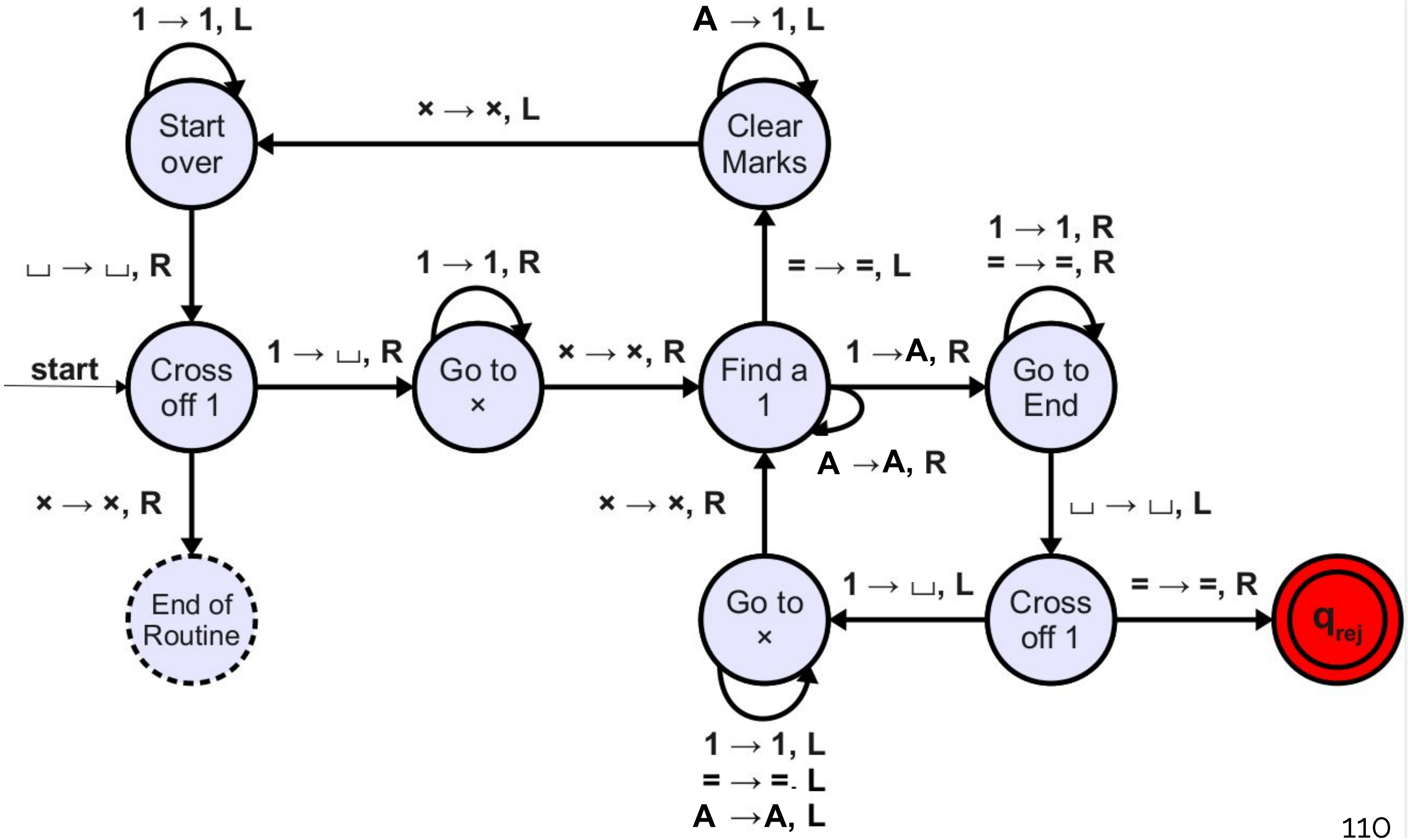
- How to make sure that : the number of 1s on the right hand side is correct ?
- Algorithm : ?

Examples for TM : $1^n x 1^m = 1^{nm}$

- Algorithm :
  - For the initial 1 found, replace with space.
  - Skip right till x,
  - For each 1 read, until =
    - Replace the 1 with A , move to find 1 on extreme **right,** replace with space
    - Get back till you find = and later A, and find 1 on the right.
    - If no 1 found , Replace all As with 1s.
  - Go to initial 1 on the extreme left. Repeat.
    - If no 1 is found before **x,** go to after **=,** there must be no 1 too.

Examples for TM :

$1^n x 1^m = 1^{nm}$



Formalism of Turing Machine

A ***Turing machine*** is a 7-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$, where Q, Σ, Γ are all finite sets and

1. Q is the set of states,
2. Σ is the input alphabet not containing the ***blank symbol*** $\sqcup$,
3. Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$,
4. $\delta \colon Q \times \Gamma \longrightarrow Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$ is the transition function,
5. $q_0 \in Q$ is the start state,
6. $q_{\text{accept}} \in Q$ is the accept state, and
7. $q_{\text{reject}} \in Q$ is the reject state, where $q_{\text{reject}} \neq q_{\text{accept}}$.

Classes of Languages for Turing Machine

- The collection of strings that M **accept**s is the language of M , or the language recognized by M
  - A language is called **Turing-recognizable** if some Turing machine recognizes it
  - Mainly : Accepting words that belong to the language.
  - For words not in the language:
    - Reject or Loop

Classes of Languages for Turing Machine

- **Turing-decidable** language or simply decidable if some Turing machine decides it
  - Halts and Accepts for words in the language
  - Halts and Reject for words not in the language
- Every Decidable language is also recognizable.

Classes of Languages for Turing Machine

- The collection of strings that M **accept**s is the language of M , or the language recognized by M
  - A language is called **Turing-recognizable** if some Turing machine recognizes it
  - Mainly : Accepting words that belong to the language.
  - For words not in the language:
    - Reject or **Loop**

Classes of Languages for Turing Machine

- **Terminologies:**
    - Turing Recognizable is called a **recursively enumerable language** in some other textbooks.
    - For turing decidable is called a **recursive language**

Classes of Languages for Turing Machine

- What about the following language :
  - {w#z | such that w is not equal to z and w,z $\in \Sigma^*$ }

Classes of Languages for Turing Machine

- What about the following language :
  - $\{w\#z \mid$ such that w is not equal to z and $w,z \in \{0,1\}^* \}$
  - $\{x_1\#x_2\#x_3\#x_4\#x_5\ldots \mid$ such that $x_1, x_2, x_3, x_4, x_5 \ldots$ are all distinct $\}$

Revision :

- DFA :
    - Construct the DFA (not NFA) for the following language :

Revision :

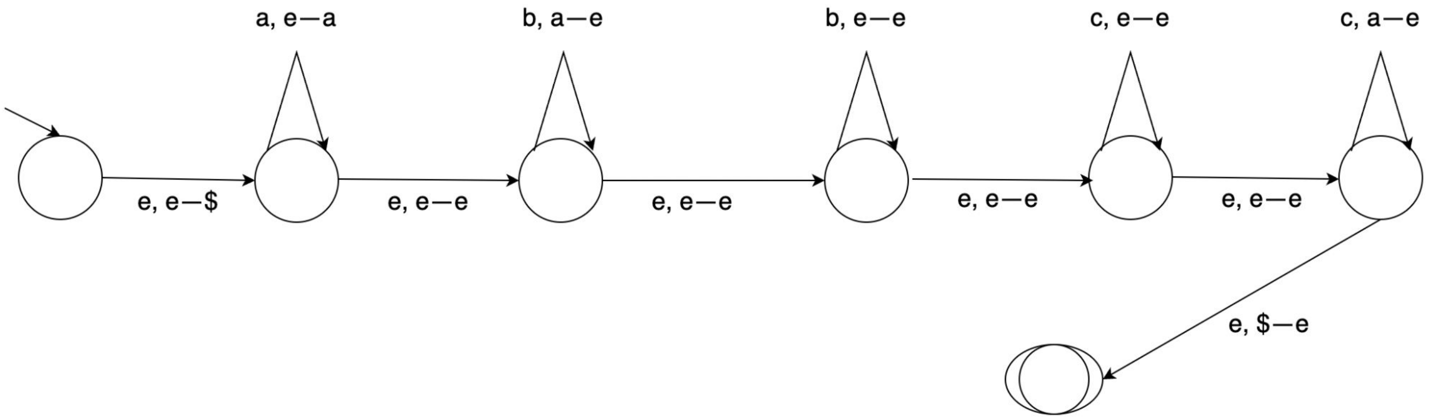
- NFA to DFA :
  - Steps.....

Revision :

- PDA :
    - I asked students about the language for 0^n1^m such that …

TD7 - Solutions

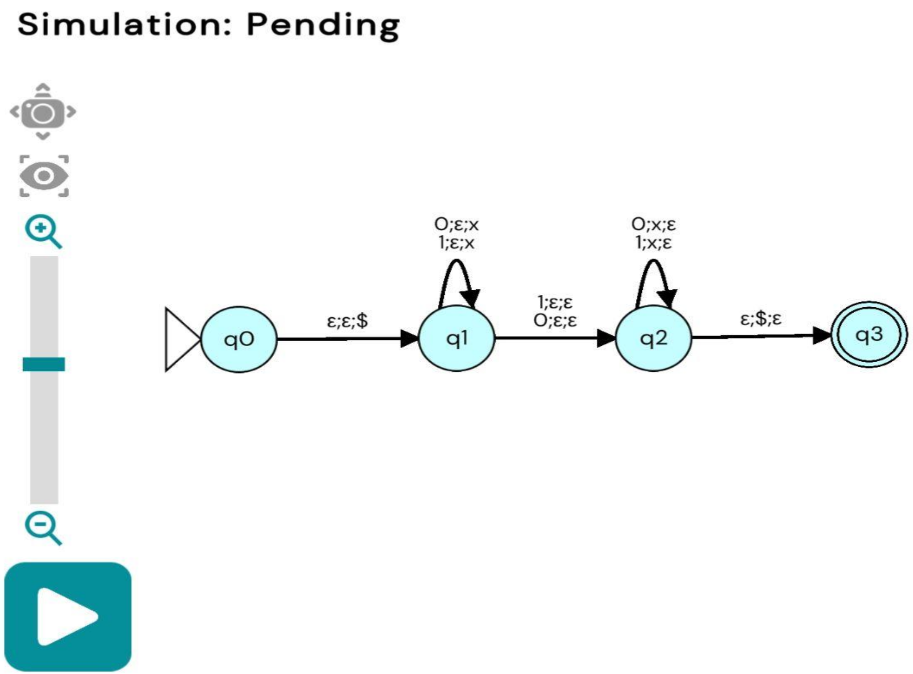
Draw a pushdown automaton for the following language L = $\{a^i b^j c^k \mid j+k \geq i\}$. Only one symbol is allowed to be pushed/popped to/from the stack at a time.

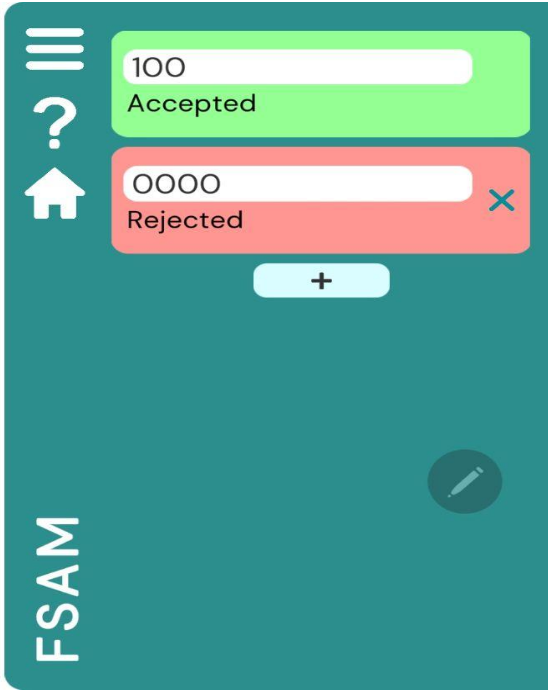


TD7 - Solutions

Give pushdown automata for the following languages:

{w| the length of w is odd and its middle symbol is a 0}

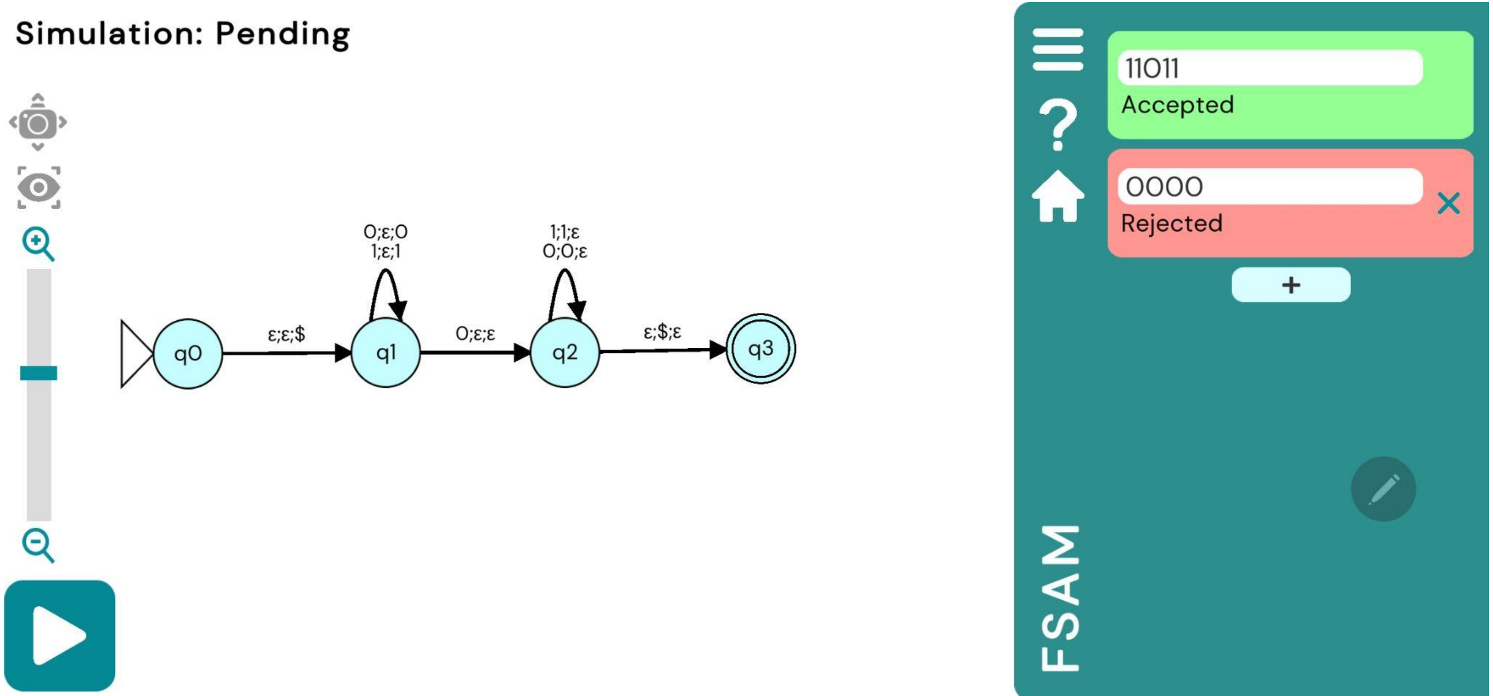




TD7 - Solutions

Give pushdown automata for the following languages:

L = {w| w = w^R , that is, w is a palindrome and |w| is odd}



TD7 - Solutions

Give pushdown automata for the following languages:

$L = \{0^m 1^n : m \geq n\}$

TD7 - Solutions

Give pushdown automata for the following languages:

$L = \{u0w1 : u \text{ and } w \in \{0, 1\}^* \text{ and } |u| = |w|\}$

TD7 - Solutions

Convert the following CFGs to an equivalent PDA:

a)

$E \rightarrow E+T \mid T$

$T \rightarrow T \times F \mid F$

$F \rightarrow (E) \mid a$

b)

$S \rightarrow aSa \mid bSb \mid aPb \mid bPa$

$P \rightarrow aP \mid bP \mid \varepsilon$