Turing Machine (TM)

TM was invented in 1936 by **Alan Turing**. It is an accepting device which accepts Recursive Enumerable Language generated by type 0 grammar.

There are various features of the Turing Machine:

1. It has an external memory which remembers arbitrary long sequence of input.
2. It has unlimited memory capability.
3. The model has a facility by which the input at left or right on the tape can be read easily.
4. The machine can produce a certain output based on its input. Sometimes it may be required that the same input has to be used to generate the output. So in this machine, the distinction between input and output has been removed. Thus, a common set of alphabets can be used for the Turing Machine.

Formal definition of Turing Machine

A TM can be formally described/expressed as a 7-tuple $(Q, \sum, T, q_0, F, B, \delta)$ where:

Q: the finite set of states.

$\sum$: the finite set of input symbols/alphabet (symbols which are part of input alphabet).

T: the tape symbol/alphabet.

$\mathbf{q_0}$: the initial state.

F: a set of final states. If any state of F is reached, input string is accepted.

B: a blank symbol used as an end marker for input (every cell is filled with B except input alphabet initially).

δ: a transition/mapping function which maps $Q \times T \rightarrow Q \times T \times \{L, R\}$.

The mapping function shows the mapping from states of finite automata and input symbol on the tape to the next states, external symbols and the direction for moving the tape head. This is known as a triple or a program for Turing Machine.

$(q_0, a) \rightarrow (q_1, A, R)$

That means in q_0 state, if we read symbol 'a' then it will go to state q_1, replaced a by X and move ahead right (R stands for right).

Comparison with the previous automaton

The following table shows a comparison of how a Turing Machine differs from Finite Automaton and Pushdown Automaton:

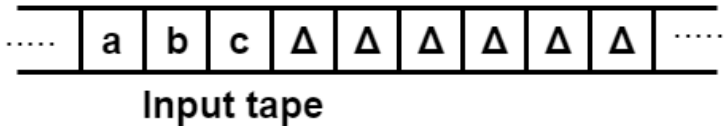
Machine	Stack Data Structure	Deterministic?
Finite Automaton	N.A	Yes
Pushdown Automaton	Last In First Out(LIFO)	No
Turing Machine	Infinite tape	Yes

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

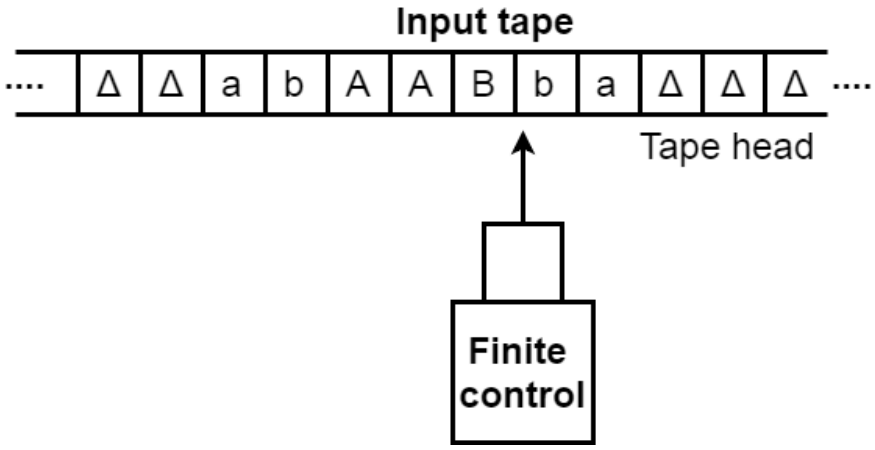
Basic Model of Turing Machine

The Turning Machine is a mathematical model which can be modelled with the help of the following representation:

1. The input tape is having an infinite number of cells, each cell containing one input symbol and thus the input string can be placed on tape. The empty tape is filled by blank characters.



2. The finite control and the tape head which is responsible for reading the current input symbol. After reading an input symbol, it is replaced with another symbol, its internal state is changed, and it moves from one cell to the right or left.
3. A finite set of states through which machine has to undergo.
4. Finite set of symbols called external symbols which are used in building the logic of Turing Machine.
5. If the TM reaches the final state, the input string is accepted, otherwise rejected.



Designing a Turing Machine

The basic guidelines of designing a Turing Machine have been explained below with the help of a couple of examples.

Language accepted by Turing Machine

The Turing Machine accepts all the language, if they enter into a final state for any input string w, even though they are recursively enumerable (generated by Type-0 grammar) if they are accepted by Turing Machines. Recursive means repeating the same set of rules for any number of times and enumerable means a list of elements. The TM also accepts the computable functions, such as addition, multiplication, subtraction, division, power function, and many more. A TM decides a language if it accepts it and enters into a rejecting state for any input not in the language. A language is recursive if it is decided by a Turing Machine. There may be some cases where a TM does not stop. Such TM accepts the language, but it does not decide it.

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

Example-
Construct a Turing Machine for language L = {w w^R | w ∈ {0, 1}}.

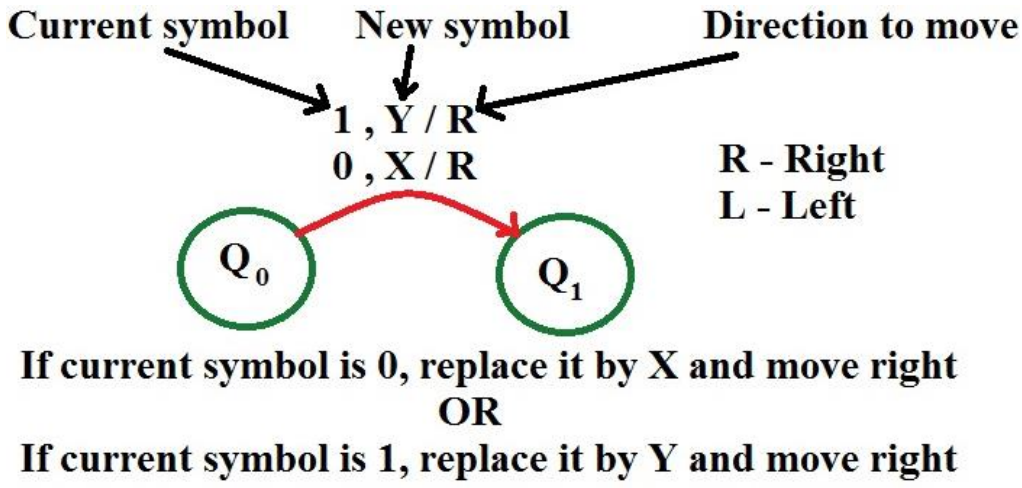
The language L = {w w^R | w ∈ {0, 1}} represents a kind of language where you use only 2 characters, i.e., 0 and 1. The first part of language can be any string of 0 and 1. The second part is the reverse of the first part. Combining both these parts out string will be formed. Any such string which falls in this category will be accepted by this language. The beginning and end of string is marked by $ sign. For example, if first part w = 1 1 0 0 1 then second part w^R = 1 0 0 1 1. It is clearly visible that w^R is the reverse of w, so the string 1 1 0 0 1 1 0 0 1 1 is a part of given language.

Examples –
Input: 0 0 1 1 1 1 0 0
Output: Accepted

Input: 1 0 1 0 0 1 0 1
Output: Accepted

Input: 1 1 1 0 1 1 1 0
Output: Rejected

Basic Representation:



Assumption: We will replace 0 by Y and 1 by X.
Approach Used –
First check the first symbol, if it's 0 then replace it by Y and by X if it's 1. Then go to the end of string. So last symbol is same as first. We replace it also by X or Y depending on it. Now again come back to the position next to the symbol replace from the starting and repeat the same process as told above.

One important thing to note is that since w^R is reverse of w of both of them will have equal number of symbols. Every time replace a symbol from beginning of string, replace a corresponding symbol from the end.

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

Step-1:
If symbol is 0 replace it by Y and move right, Go to state q_2.
If symbol is 1 replace it by X and move right, Go to state q_1.

Step-2:
If symbol is 0 replace it by 0 and move right, remain on same state.
If symbol is 1 replace it by 1 and move right, remain on same state.
If symbol is X replace it by X and move right, Go to state q_3.
If symbol is Y replace it by Y and move right, Go to state q_3.
If symbol is $ replace it by $ and move right, Go to state q_3.

Step-3:
If symbol is 1 replace it by X and move left, Go to state q_4.
If symbol is 0 replace it by Y and move left, Go to state q_5.

Step-4:
If symbol is 1 replace it by 1 and move left.
If symbol is 0 replace it by 0 and move left.
Remain on same state.

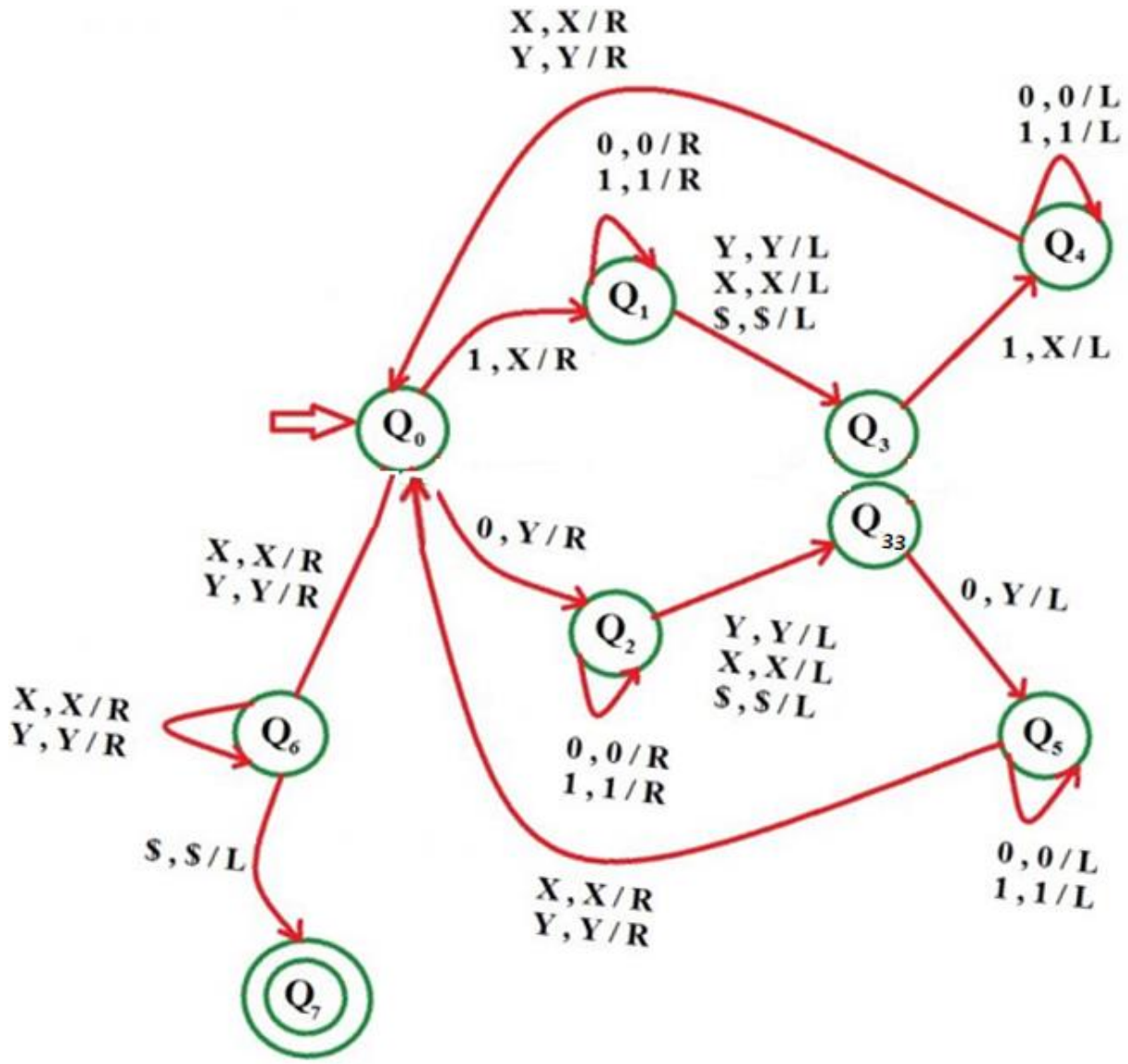
Step-5:
If symbol is X replace it by X and move right.
If symbol is Y replace it by Y and move right.
Go to state q_0.

Step-6:
If symbol is X replace it by X and move right.
If symbol is Y replace it by Y and move right.
Go to state q_6.
ELSE
Go to step 1

Step-7:
If symbol is X replace it by X and move right, Remain on same state.
If symbol is Y replace it by Y and move right, Remain on same state.
If symbol is $ replace it by $ and move left, String is ACCEPTED, Go to final state q_7.

The same TM can be represented by Transition Diagram:

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.



Example-
Construct a Turing Machine for the language L = $\{0^n 1^n\}$ where n >= 1.

Solution:

- Q = $\{q_0, q_1, q_2, q_3\}$ where q_0 is initial state.
- T = {0, 1, X, Y, B} where B represents blank.
- $\sum$ = {0,1}
- F = $\{q_3\}$

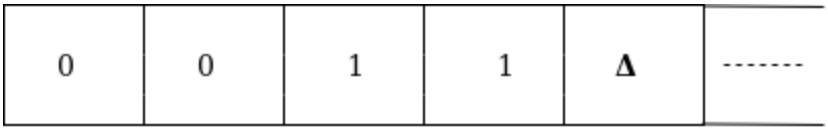
We have already solved this problem by PDA. In PDA, we have a stack to remember the previous symbol. The main advantage of the Turing Machine is we have a tape head which can be moved forward or backward, and the input tape can be scanned.

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

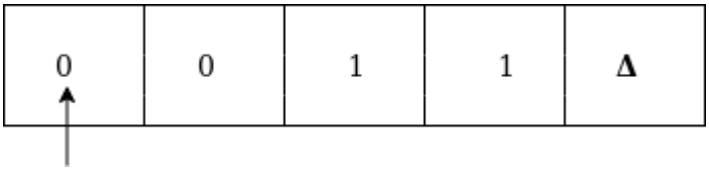
The simple logic which we will apply is read out each '0' mark it by A and then move ahead along with the input tape and find out 1 convert it to B. Now, repeat this process for all 0's and 1's.

Now, we will see how this Turing Machine work for 0011.

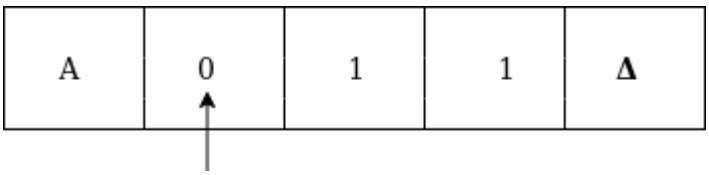
The simulation for 0011 can be shown as below:



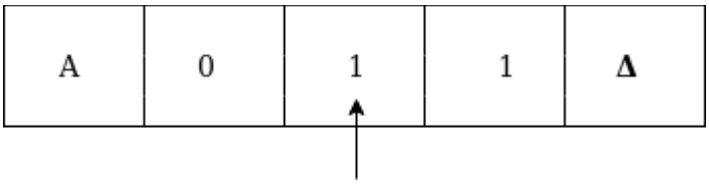
Initially, state is q_0 and head points to 0 as:



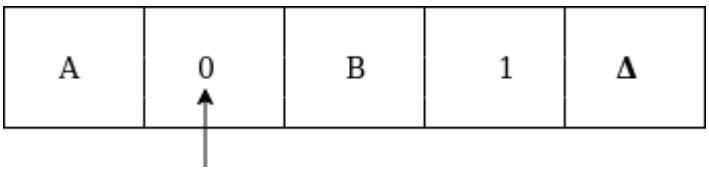
The move will be $\delta(q_0, 0) = \delta(q_1, A, R)$ which means it will go to state q_1, replaced 0 by A and head will move to the right as:



The move will be $\delta(q_1, 0) = \delta(q_1, 0, R)$ which means it will not change any symbol, remain in the same state and move to the right as:

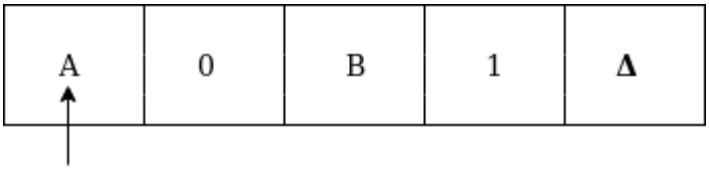


The move will be $\delta(q_1, 1) = \delta(q_2, B, L)$ which means it will go to state q_2, replaced 1 by B and head will move to left as:

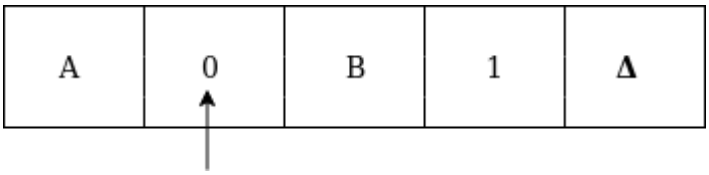


Now move will be $\delta(q_2, 0) = \delta(q_2, 0, L)$ which means it will not change any symbol, remain in the same state and move to left as:

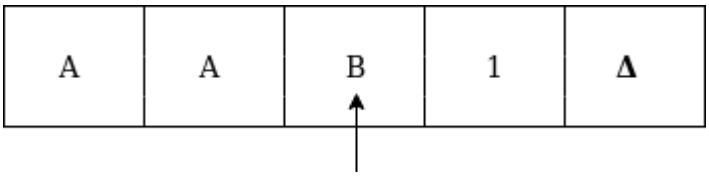
(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.



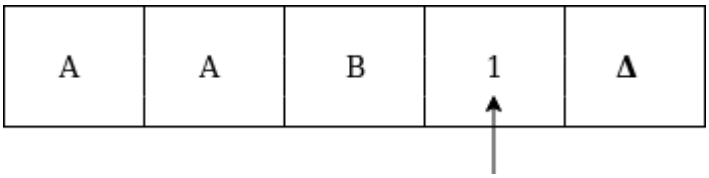
The move will be $\delta(q_2, A) = \delta(q_0, A, R)$, it means will go to state q_0, replaced A by A and head will move to the right as:



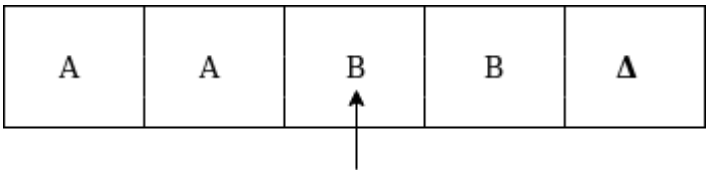
The move will be $\delta(q_0, 0) = \delta(q_1, A, R)$ which means it will go to state q_1, replaced 0 by A, and head will move to right as:



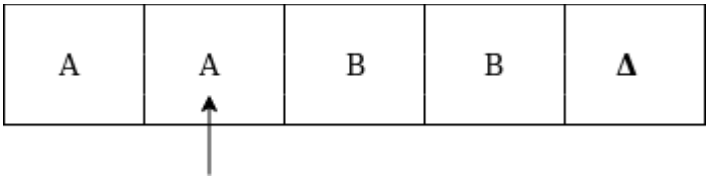
The move will be $\delta(q_1, B) = \delta(q_1, B, R)$ which means it will not change any symbol, remain in the same state and move to right as:



The move will be $\delta(q_1, 1) = \delta(q_2, B, L)$ which means it will go to state q_2, replaced 1 by B and head will move to left as:

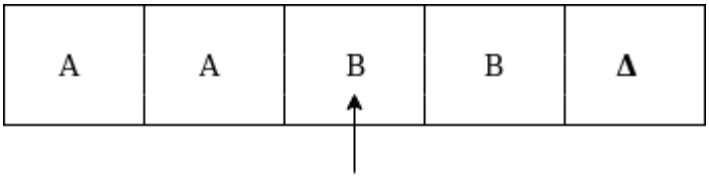


The move $\delta(q_2, B) = (q_2, B, L)$ which means it will not change any symbol, remain in the same state and move to left as:

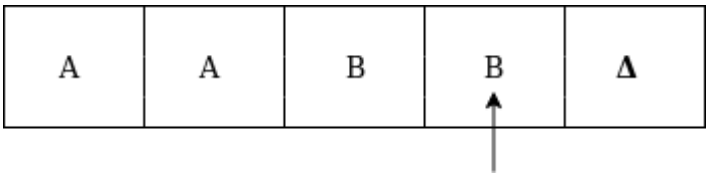


Now immediately before B is A that means all the 0's are market by A. So we will move right to ensure that no 1 is present. The move will be $\delta(q_2, A) = (q_0, A, R)$ which means it will go to state q_0, will not change any symbol, and move to right as:

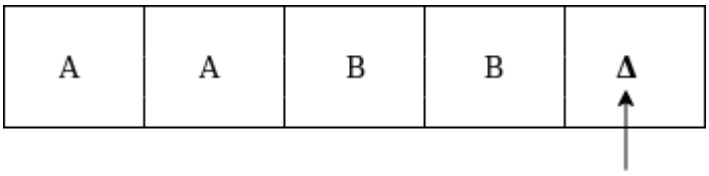
(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.



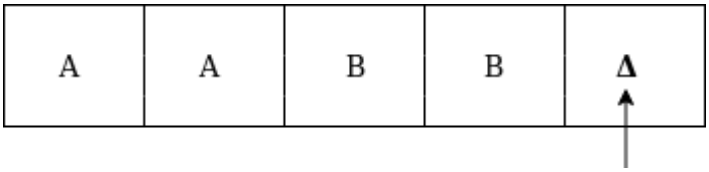
The move $\delta(q_0, B) = (q_3, B, R)$ which means it will go to state q_3, will not change any symbol, and move to right as:



The move $\delta(q_3, B) = (q_3, B, R)$ which means it will not change any symbol, remain in the same state and move to right as:



The move $\delta(q_3, \Delta) = (q_4, \Delta, R)$ which means it will go to state q_4 which is the HALT state and HALT state is always an accept state for any TM.



HW. Represent a Transition Diagram for the same TM above.

Example-
Construct a Turing Machine for language $L = \{0^{2n} 1^n \mid n >= 0\}$

Solution
The language $L = \{0^{2n}1^n \mid n >= 0\}$ represents a kind of language where we use only 2 symbols, i.e., 0 and 1. In the beginning language has some number of 0's followed by exactly half number of 1's. Any such string which falls in this category will be accepted by this language.

For example,

Input: 001
Output: YES

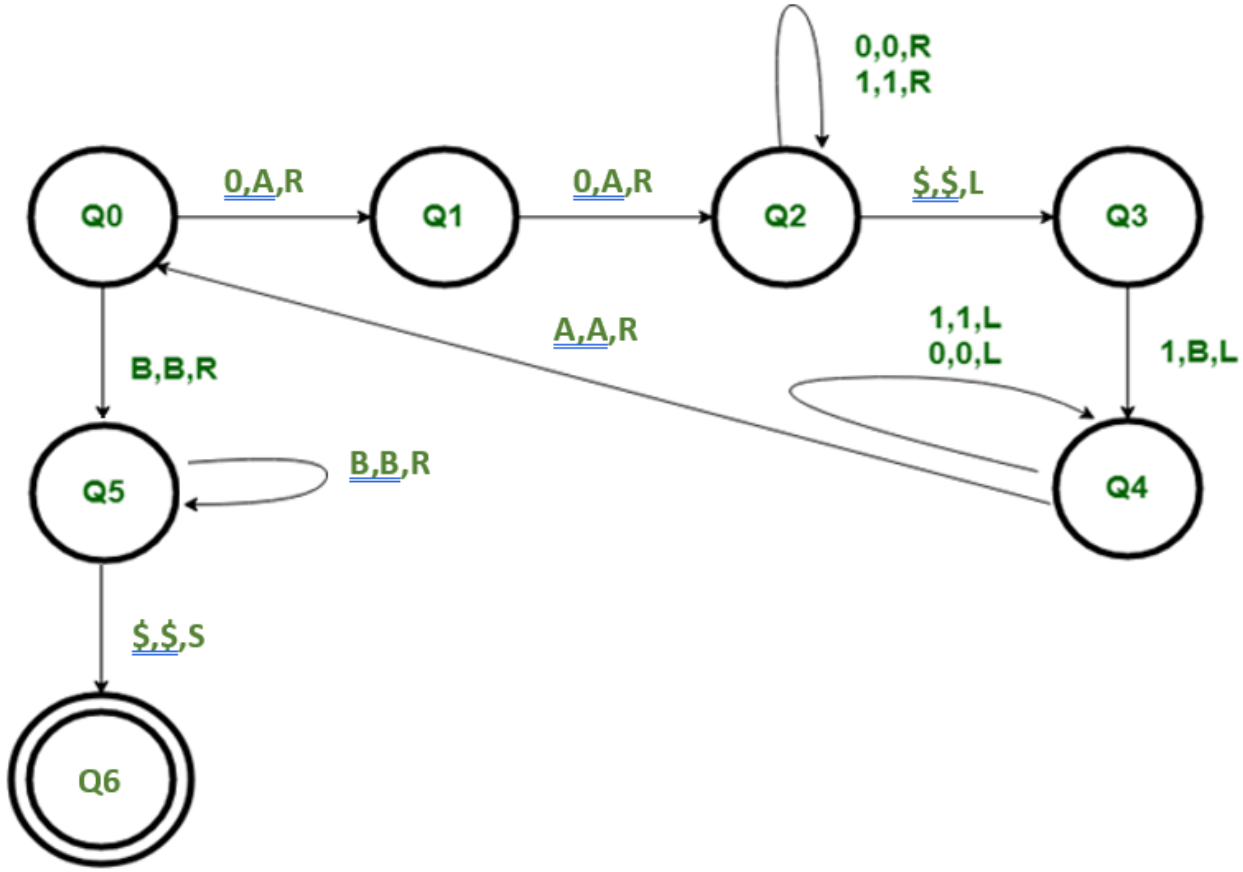
Input: 00001
Output: NO

Input: ε or empty string
Output: YES

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

Start of Computation:
The tape contains the input string w, the tape head is on the leftmost symbol of w, and the Turing Machine is in the start state q_0.

Basic Idea:
The tape head reads the leftmost symbol of the string, which is 0 and makes it **A** then the next left most 0 is made **A** after this we traverse to the rightmost 1 of the string and make it **B**. In this way we have reduced the string to $0^{2n-2}1^{n-1}$. If the string belongs to language L then at end empty string will be left and hence gets accepted by the machine.



Example-
Construct a Turing Machine which accepts the language of **aba** over $\sum = \{a, b\}$.

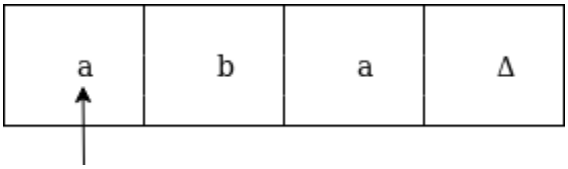
Solution
We will assume that on input tape the string 'aba' is placed like this:

a	b	a	Δ	-------

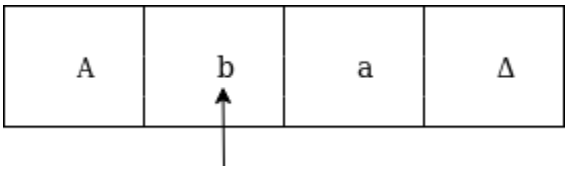
(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

The tape head will read out the sequence up to the Δ characters. If the tape head is readout 'aba' string, then TM will halt after reading Δ.

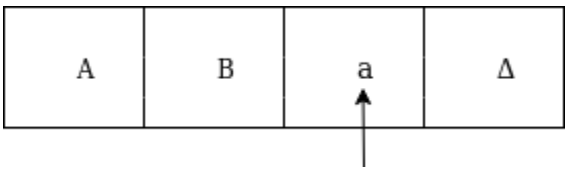
Now, we will see how this Turing Machine will work for aba. Initially, state is q_0 and head points to a as:



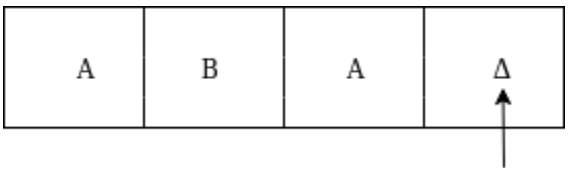
The move will be $\delta(q_0, a) = \delta(q_1, A, R)$ which means it will go to state q_1, replaced a by A and head will move to right as:



The move will be $\delta(q_1, b) = \delta(q_2, B, R)$ which means it will go to state q_2, replaced b by B and head will move to right as:



The move will be $\delta(q_2, a) = \delta(q_3, A, R)$ which means it will go to state q_3, replaced a by A and head will move to right as:



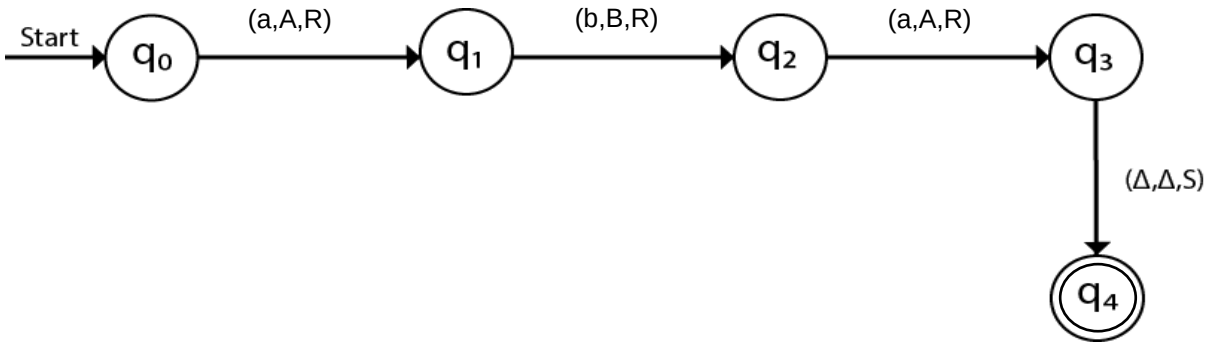
The move $\delta(q_3, \Delta) = (q_4, \Delta, S)$ which means it will go to state q_4 which is the HALT state and HALT state is always an accept state for any TM.

The same TM can be represented by Transition Table:

States	a	b	Δ
q_0	(q_1, A, R)	–	–
q_1	–	(q_2, B, R)	–
q_2	(q_3, A, R)	–	–
q_3	–	–	(q_4, Δ, S)
q_4	–	–	–

(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.

The same TM can be represented by Transition Diagram:



(*) أستاذ مساعد، قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة الموصل.