A TM accepts a language if it enters into a final state for any input string w. A language is recursively enumerable (generated by Type-0 grammar) if it is accepted by a Turing machine.

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.

Designing a Turing Machine

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

Example 1

Design a TM to recognize all strings consisting of an odd number of α’s.

***Solution***

The Turing machine **M** can be constructed by the following moves −

* Let **q1** be the initial state.
* If **M** is in **q1**; on scanning α, it enters the state **q2** and writes **B** (blank).
* If **M** is in **q2**; on scanning α, it enters the state **q1** and writes **B** (blank).
* From the above moves, we can see that **M** enters the state **q1** if it scans an even number of α’s, and it enters the state **q2** if it scans an odd number of α’s. Hence **q2** is the only accepting state.

Hence,

M = {{q1, q2}, {1}, {1, B}, δ, q1, B, {q2}}

where δ is given by −

|  |  |  |
| --- | --- | --- |
| **Tape alphabet symbol** | **Present State ‘q1’** | **Present State ‘q2’** |
| α | BRq2 | BRq1 |

Example 2

Design a Turing Machine that reads a string representing a binary number and erases all leading 0’s in the string. However, if the string comprises of only 0’s, it keeps one 0.

***Solution***

Let us assume that the input string is terminated by a blank symbol, B, at each end of the string.

The Turing Machine, **M**, can be constructed by the following moves −

* Let **q0** be the initial state.
* If **M** is in **q0**, on reading 0, it moves right, enters the state **q1** and erases 0. On reading 1, it enters the state **q2** and moves right.
* If **M** is in **q1**, on reading 0, it moves right and erases 0, i.e., it replaces 0’s by B’s. On reaching the leftmost 1, it enters **q2** and moves right. If it reaches B, i.e., the string comprises of only 0’s, it moves left and enters the state **q3**.
* If **M** is in **q2**, on reading either 0 or 1, it moves right. On reaching B, it moves left and enters the state **q4**. This validates that the string comprises only of 0’s and 1’s.
* If **M** is in **q3**, it replaces B by 0, moves left and reaches the final state **qf**.
* If **M** is in **q4**, on reading either 0 or 1, it moves left. On reaching the beginning of the string, i.e., when it reads B, it reaches the final state **qf**.

Hence,

M = {{q0, q1, q2, q3, q4, qf}, {0,1, B}, {1, B}, δ, q0, B, {qf}}

where δ is given by −

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tape alphabet symbol** | **Present State ‘q0’** | **Present State ‘q1’** | **Present State ‘q2’** | **Present State ‘q3’** | **Present State ‘q4’** |
| 0 | BRq1 | BRq1 | ORq2 | - | OLq4 |
| 1 | 1Rq2 | 1Rq2 | 1Rq2 | - | 1Lq4 |
| B | BRq1 | BLq3 | BLq4 | OLqf | BRqf |