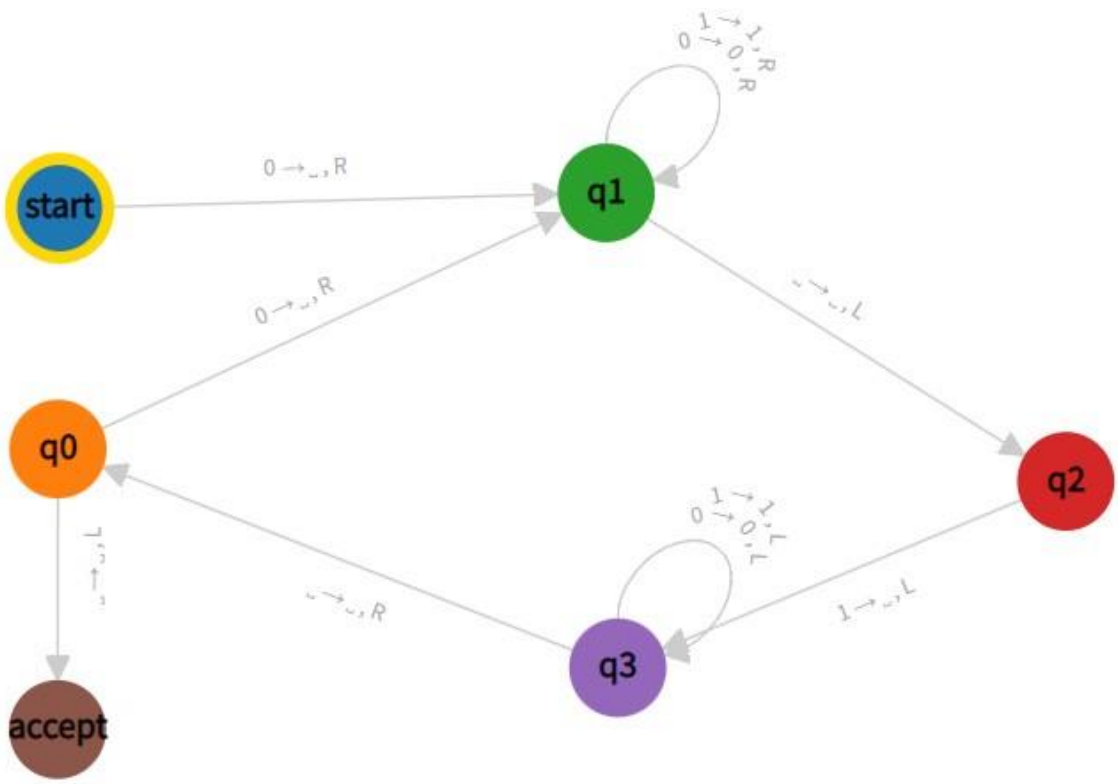
Student Information

Name : Gürhan İlhan Adıgüzel

ID : 2448025

Answer 1

Turing Machine - 1



''start '' state checks whether given string is empty or not. If string is not empty and 0 appears, it writes ‘ ’, then the tape moves right and the next state is ''q1''.

''q0'' state when 0 appears it writes ‘ ’ and moves right. When ‘ ’ appears then tape moves left and goes the ''accept'' state.

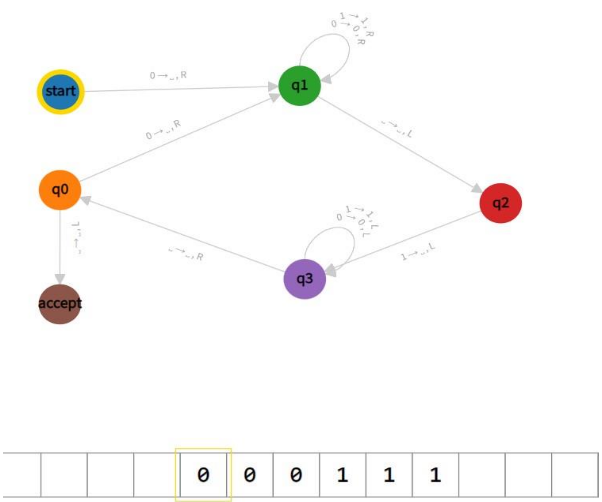
''q1'' state moves right whenever it sees 0 or 1 and state does not change. When ‘ ’ appears it just moves left and the next state is ''q2''.

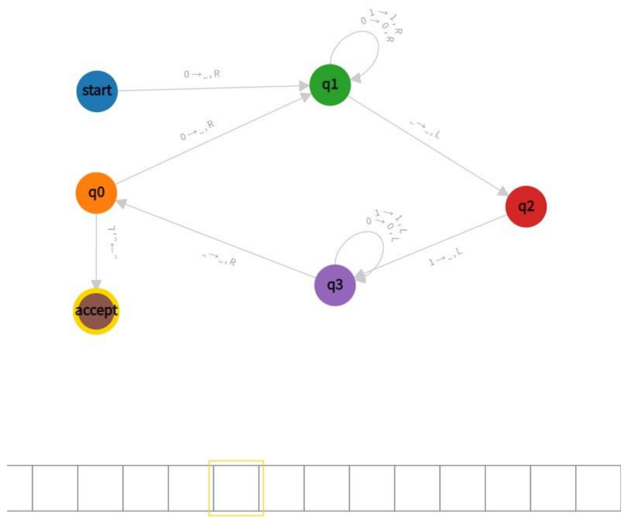
''q2'' when 1 appears it writes ‘ ’, moves left and the next state is ''q3''.

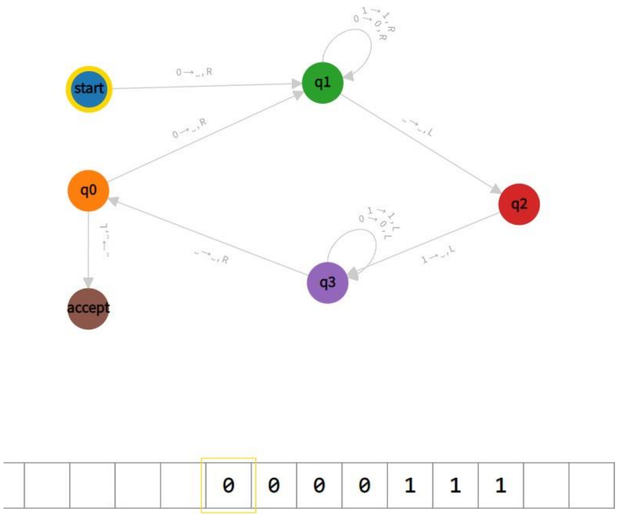
''q3'' state moves left right whenever it sees 0 or 1 and state does not change. When ‘ ’ appears it just moves right and next state is ''q0''.

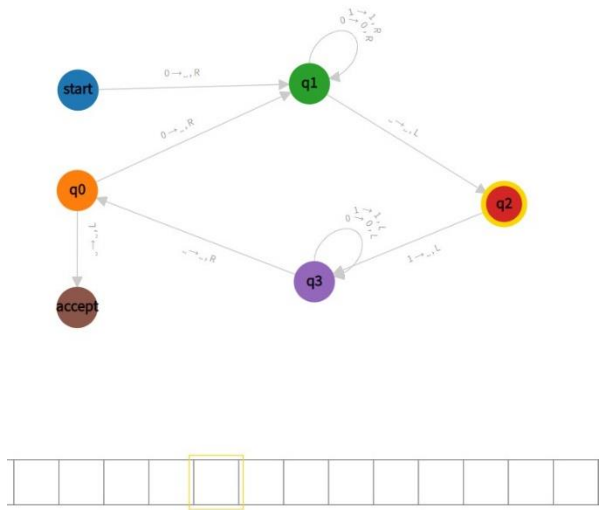
INPUT

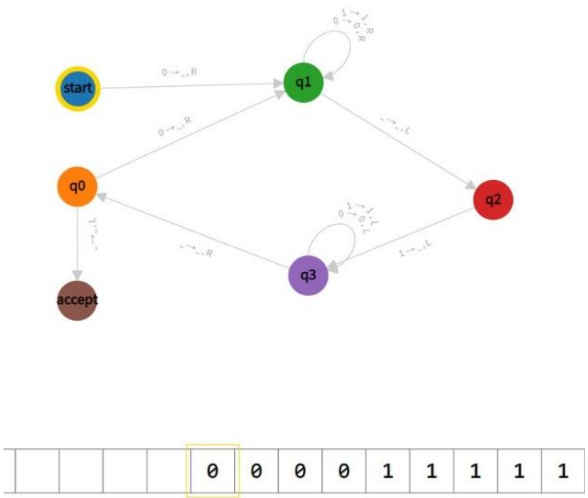
OUTPUT

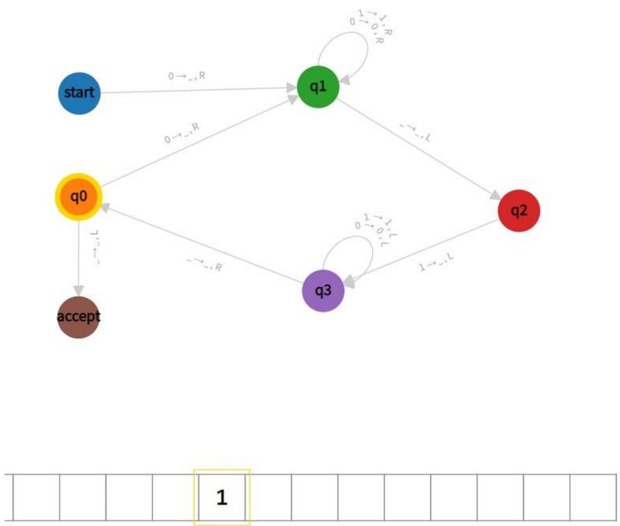


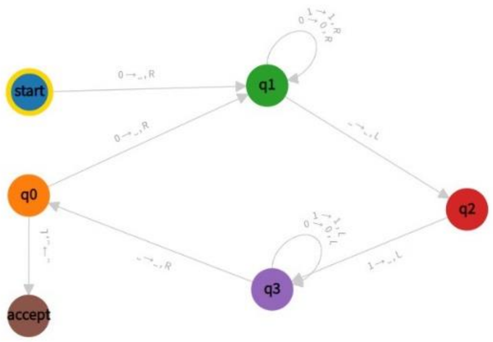




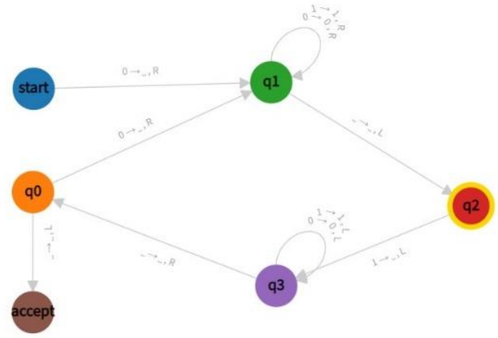




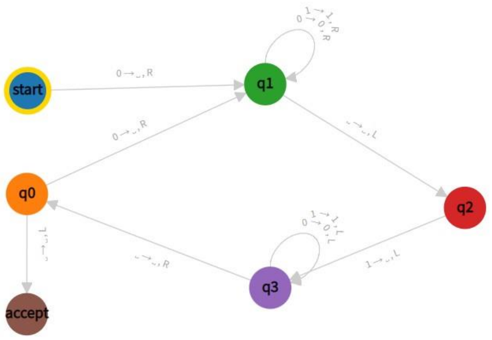




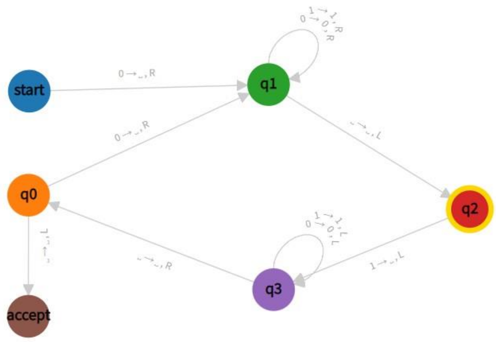
				0	0	0	1	1	1	0		



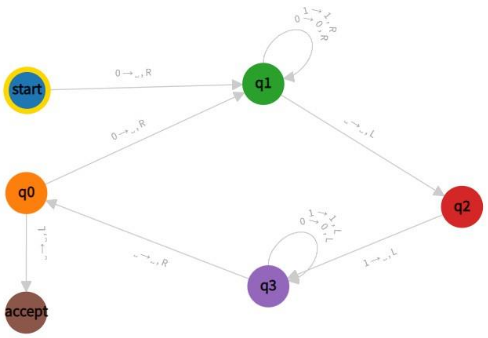
0	0	1	1	1	0							



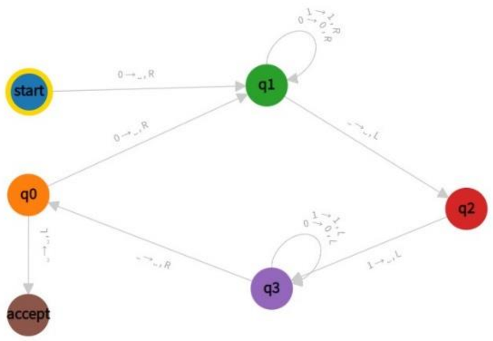
				0	0	1	1	0	1		



		1	1	0							



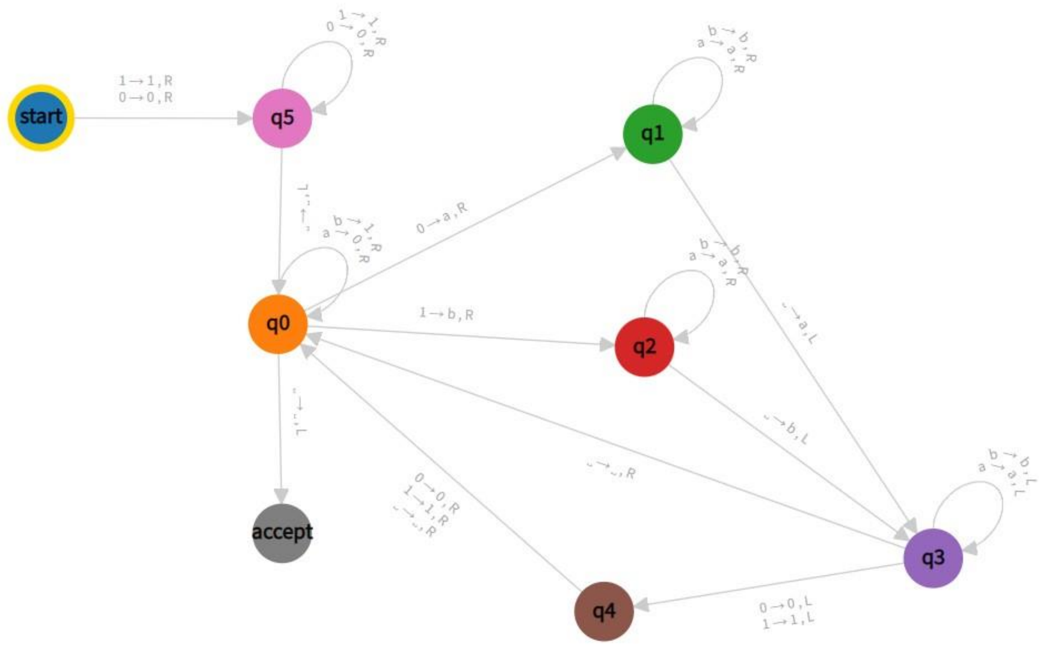
				1	0	0	0	1	1		



				1	0	0	0	1	1		

Answer 2

Turing Machine - 2



''start '' state checks whether given string is empty or not. If string is not empty and tape sees 0 or 1, it writes '   ' and the tape moves right and the next state is ''q5''.

''q0'' state when 'a' appears it writes 1 and moves right, when 'b' appears it writes 0 and moves right and state does not change. When 0 appears writes 'a', moves right and the next state is ''q1''. When 1 appears writes ' b', moves right and the next state is ''q2''. When ' ' appears moves left and tape goes the ''accept'' state.

''q1'' state when 'a' or 'b' appears, moves right and state does not change. When ' ' appears then write 'a',moves left and next state is ''q3''.

''q2'' state moves right whenever it sees 'a' or 'b' and state does not change.When ' ' appears it writes 'b' moves left and the next state is ''q3''.

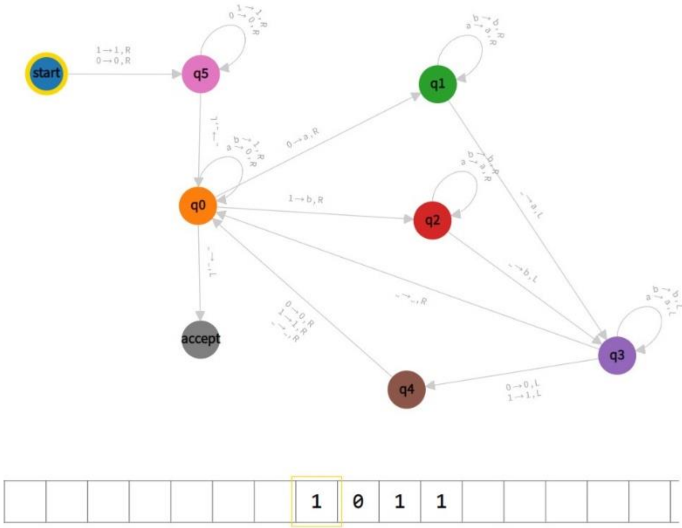
''q3'' state moves left whenever it sees 'a' or 'b' and state does not change. When 0 or 1 appears it moves left and the next state is ''q4''.

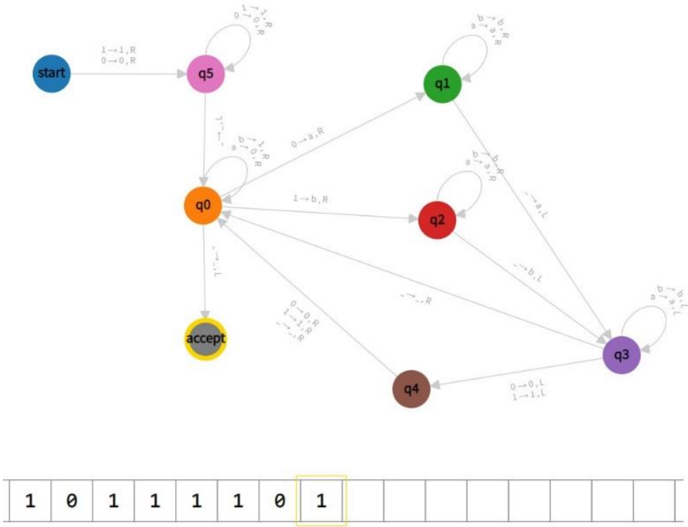
''q4'' state when it sees 0, 1 or '   ' it moves right and the next state is ''q0''.

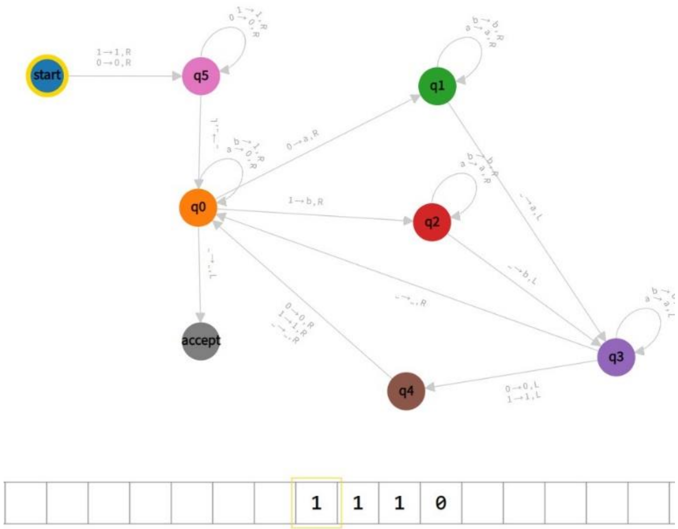
''q5'' state when 0 and 1 appears it moves right and state does not change. When '   ' appears it moves left and the next state is ''q0''.

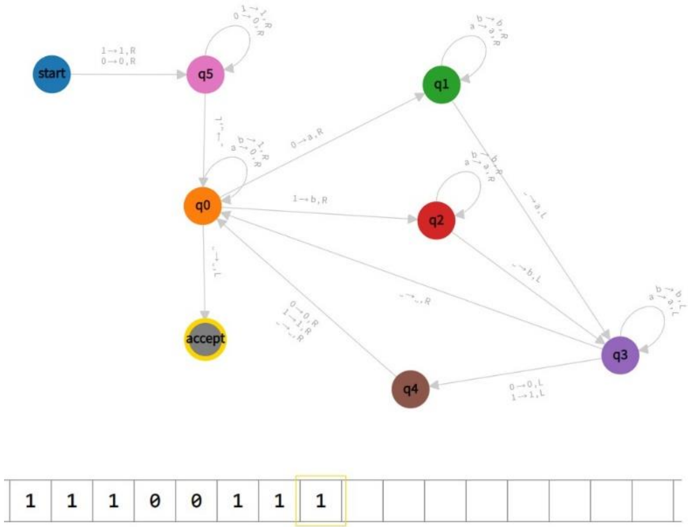
INPUT

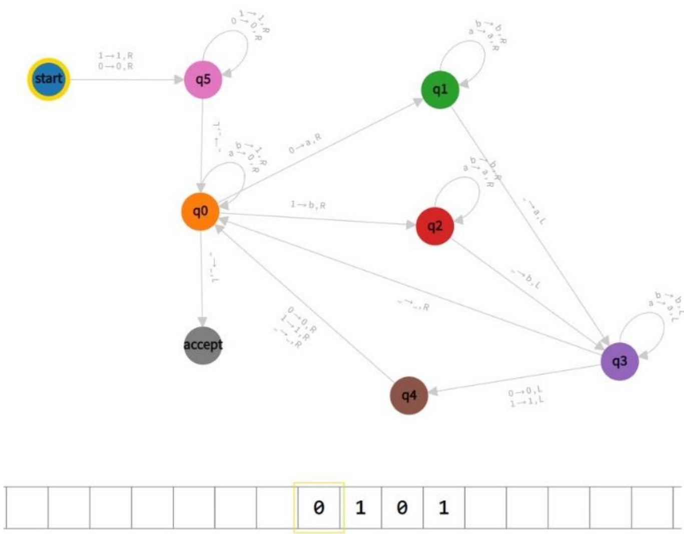
OUTPUT

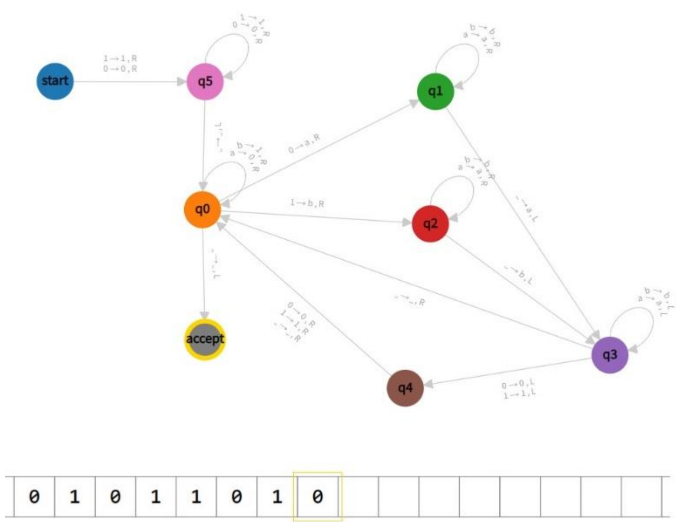


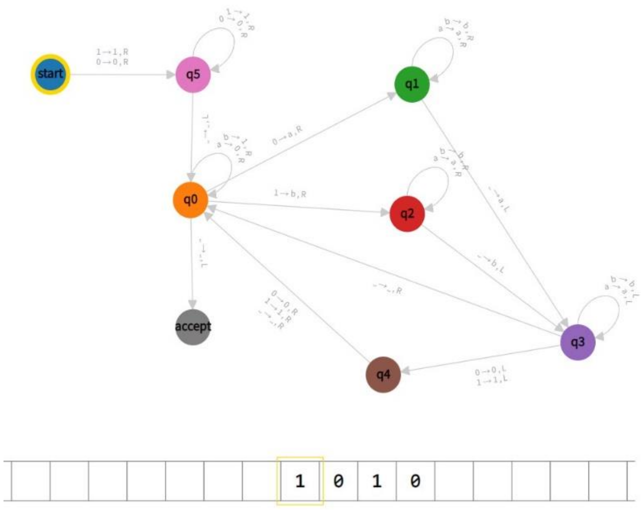


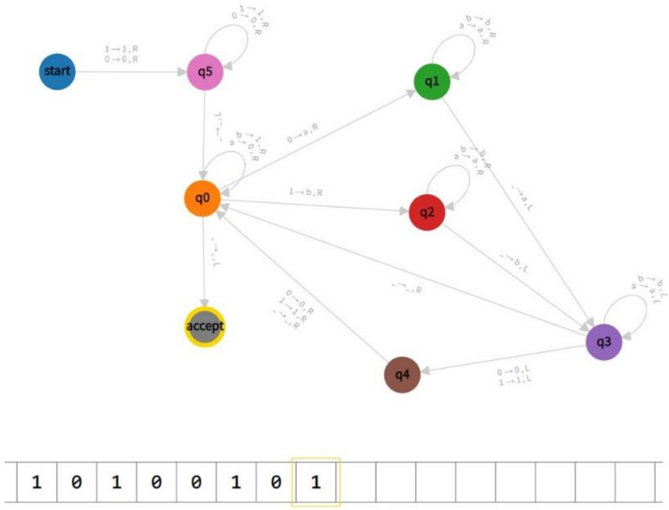


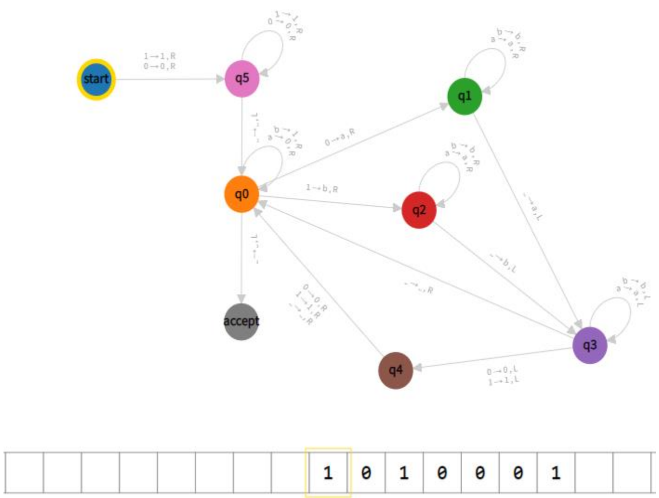


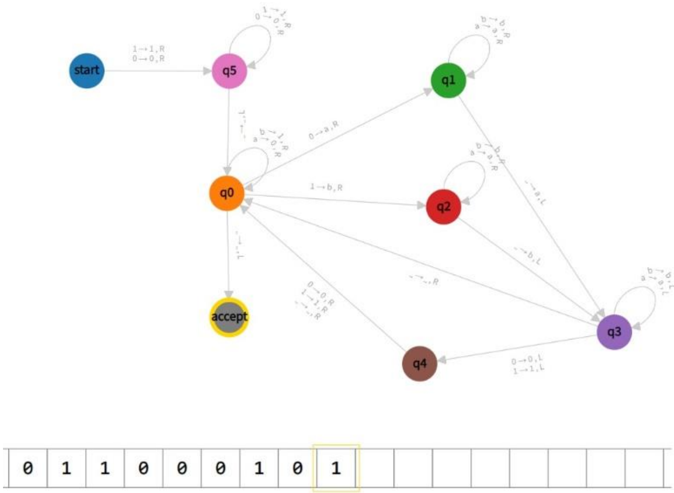




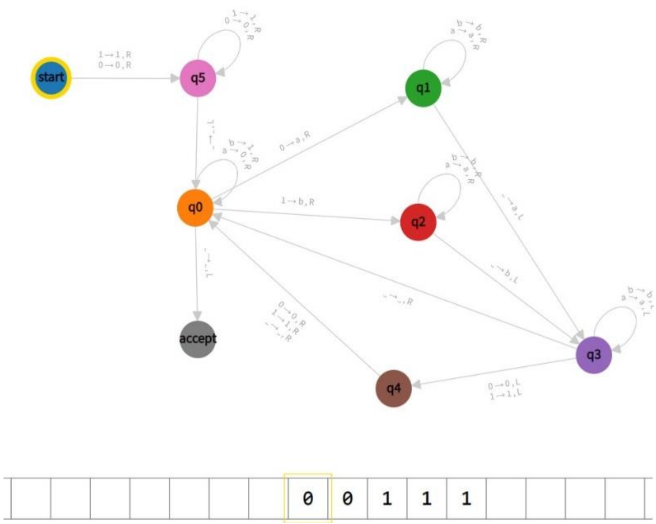


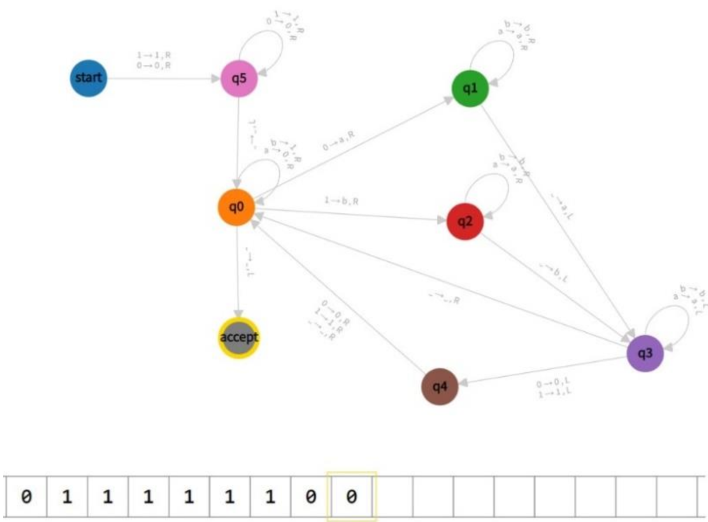






*output is not totally visible because of website





*output is not totally visible because of website

Answer 3

A two-dimensional Turing machine is a pentuple M = (K, Σ, δ, s, H), where K, s and H are as given for a normal Turing machine, where Σ is defined as for a Turing machine but also includes the tape-bottom marker Δ, and where δ is a function from K x Σ to K x ($\Sigma \cup \{\leftarrow, \uparrow, \rightarrow, \downarrow\}$), such that δ (q_1, $\rhd$) = (q_2, $\rightarrow$) and δ (q_1, Δ) = (q_2, $\uparrow$) for all q_1.

A configuration of a two-dimensional Turing machine is K x N x N x T,

where S is the set of functions from N x N to Σ that have t(0, y) = $\rhd$ for all y $\in$ N, t(x, 0) = Δ for all x > 0 and have t(x,y) = 0 for all but a finite number of (x,y) pairs. We thus implicitly represent a configuration by the current state, current head position, and list of all non-blank squares on the tape. We say that (q_1, x_1, y_1, t_1) $\vdash_M$ (q_2, x_2, y_2, t_2) if δ (q_1, $t_1(x_1,y_1)$) = (q_2, σ) and one of following.

$x_1 = x_2$, $y_1 - 1 = y_2$, $t_1 = t_2$, and $\sigma = \leftarrow$

$x_1 + 1 = x_2$, $y_1 + 1 = y_2$, $t_1 = t_2$, and $\sigma = \uparrow$

$x_1 = x_2$, $y_1 + 1 = y_2$, $t_1 = t_2$, and $\sigma = \rightarrow$

$x_1 - 1 = x_2$, $y_1 + 1 = y_2$, $t_1 = t_2$, and $\sigma = \downarrow$

$x_1 = x_2$, $y_1 + 1 = y_2$, $t_2(x_1,y_1) = \sigma$, $t_2(x,y) = t_1(x,y)$ for all other pairs (x,y), and $\sigma \notin \{\leftarrow, \uparrow, \rightarrow, \downarrow\}$

Given a string w, let $t_w \in T$ be the function that has t(i+1, 1) = w(i) for 0 < i $\leq$ |w|, t(0,y) = $\rhd$ for y $\in$ N, t(x,0)=Δ for all x > 0, and t(x,y) = ‘ ’ otherwise. Then if we have a two-dimensional tape Turing machine M with with two distinguished halting states y and n such that for any string w either (s,1,1, t_w) $\vdash_M^*$ (y, i, j, t’) or (s,1,1, t_w) $\vdash_M^*$ (n, i, j, t’) for some i, j $\in N$ and t’ $\in T$, we let the language decided by M be the set of strings for which M thus halts in the y state.

M' is a three-tape Turing Machine that we use to simulate M. M' uses one tape for calculation, one for holding the encoding of M's tape, and the third for taking its input. M is required to execute the following types of calculations: It will need to increment and decrement two'registers' I and j for M''s tape location, and then do simple multiplications and additions to transform this number into binary, which will tell M' where the symbol in that cell is stored. M' begins by scanning ahead through its real input and labeling each symbol in the encoded spot M would receive that symbol as input, copying out the tape of M into its internal representation.

M' checks the current symbol at each stage of the computation and then moves on. If the instructions say to write a symbol, M' does so and then moves on. M' increments and decrements I and j as needed, then calculates the value of the dovetail function if the instructions are to move. The encoding-tape head is then moved all the way to the left, then out to the appropriate square by M'.

A language L is said to be decided by a 2-dimensional Turing machine if, for every string w in L, the machine halts in a final state after a finite number of steps, and for every string w not in L, the machine never halts.