

Example: One Tape Turing Machine_{JP}

Define a Turing Machine M that decides the language $L = \{ w cw \mid w \in \{a,b\}^* \}$

Recall that JFLAP defines a Turing Machine M as the septuple $M = (Q, \Sigma, \Gamma, \delta, q_s, \square, F)$ where

- Q is the set of internal states $\{q_i \mid i \text{ is a nonnegative integer}\}$
- Σ is the input alphabet
- Γ is the finite set of symbols in the tape alphabet
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$ is the transition function
- \square is the blank symbol.
- q_s (is member of Q) is the initial state
- F (is a subset of Q) is the set of final states

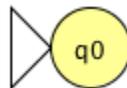
Sample Solution (see: TM_wcw.jff)

One approach to defining this Turing Machine (TM) takes advantage of the existence of a single symbol, c , marking the end of the first occurrence of w and the beginning of the second occurrence of w . The TM can determine if the initial symbol and the first symbol following c are identical. If so, they can be removed from further consideration and each subsequent character checked for a match. If the symbols are ever different, the string is not in language L . If the symbol c is reached before the end of the input string, then the string is not in language L . If the end of string is reached before c , then the string is not in language L . Otherwise, because all symbols have been matched and the c symbol marks the exact center of the string, the string is in language L .

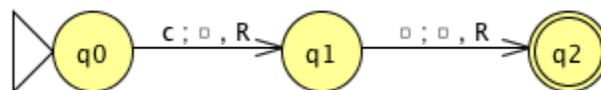
1. Identify the input alphabet: $\Sigma = \{a, b, c\}$

2. Identify the tape alphabet, which should include the input alphabet, the blank symbol, and another symbol to be used as a marker for symbols already processed: $\Gamma = \{a, b, c, 0, \square\}$

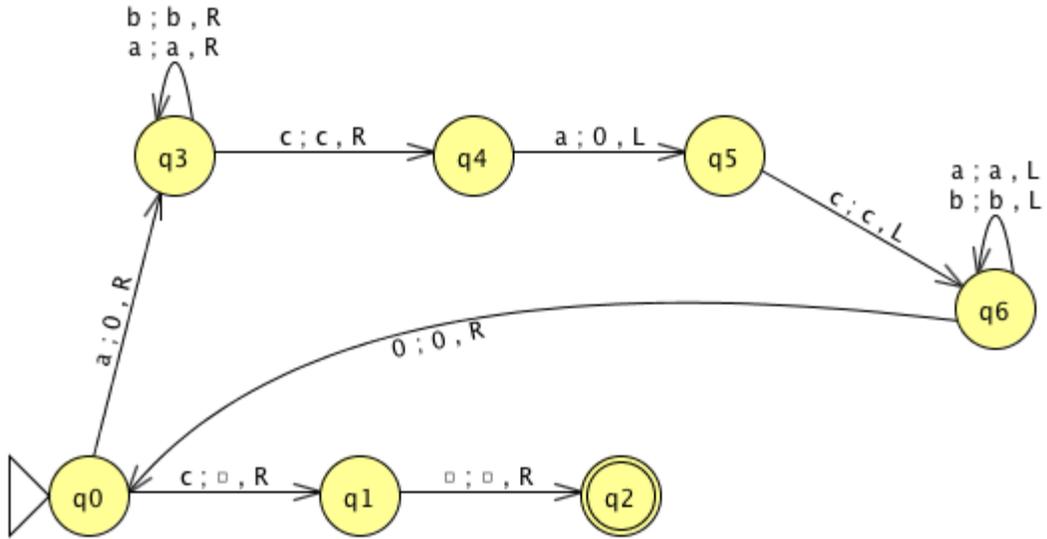
3. Create an initial state for the TM.



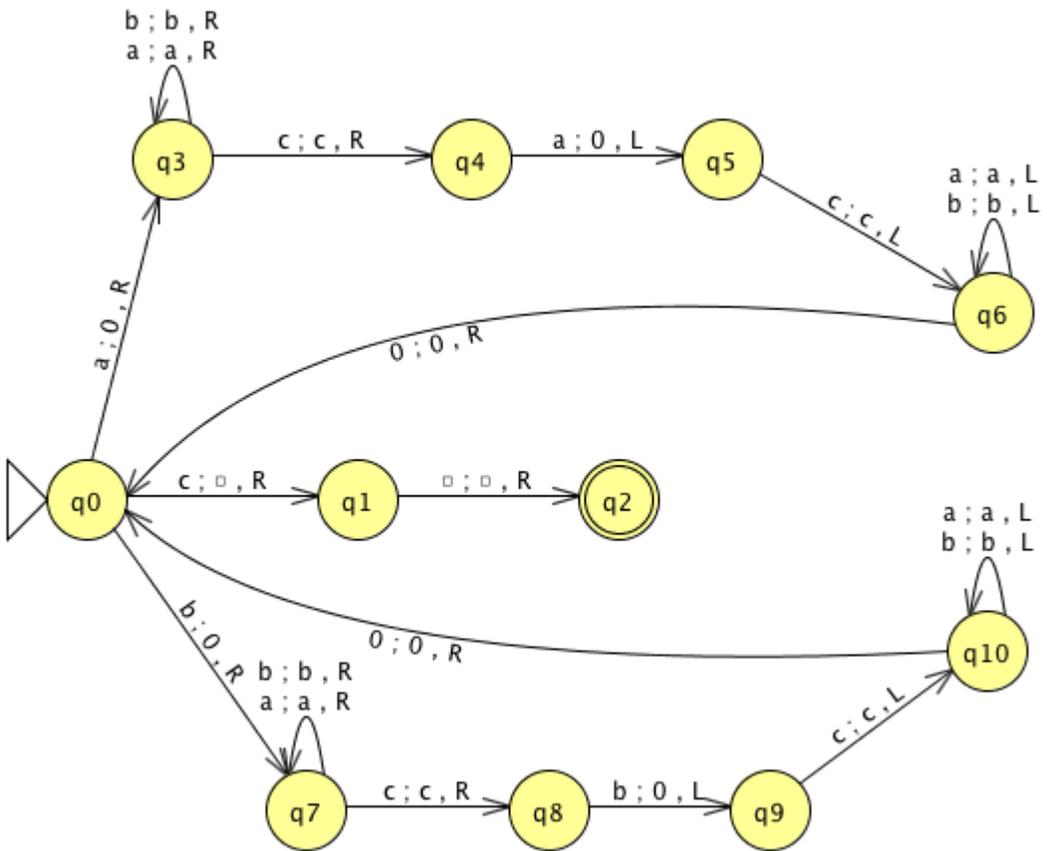
4. Address and accept the string “c”.



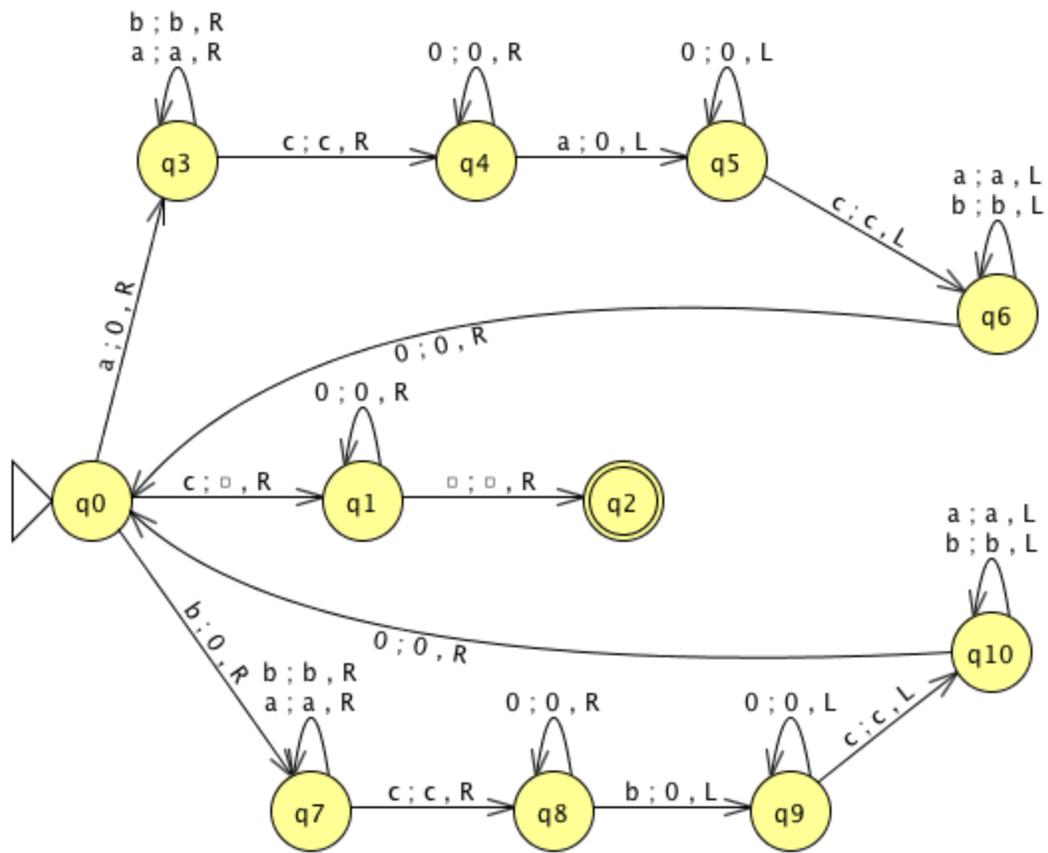
5. Address the case in which the current symbol is a by overwriting it with 0 , scanning right until the c symbol is found, verifying that the next symbol is also a , overwriting that with 0 , then scanning left to process the next symbol of the string.



6. Address the analogous case in which the current symbol is **b**.



7. Handle subsequent symbols by skipping over markers in the second instance of w and unprocessed symbols in the first instance of w .



8. Check your TM by running multiple inputs and comparing with expected results.

The screenshot shows the JFLAP software interface. The main window displays a state transition diagram for a Turing Machine with 11 states (q0 to q10). The start state is q0, and the final state is q2. Transitions are defined by the following rules:

- q0: a:0,R → q3; b:0,L → q7; c:0,R → q1; 0:0,R → q10
- q1: 0:0,R → q2
- q2: 0:0,R → q1
- q3: b:b,R → q3; a:a,R → q3; c:c,R → q4
- q4: 0:0,R → q4; a:0,L → q5
- q5: 0:0,L → q5; c:c,L → q6
- q6: a:a,L → q6; b:b,L → q6
- q7: b:b,R → q7; a:a,R → q7; 0:0,R → q10
- q8: c:c,R → q8; b:0,L → q9
- q9: 0:0,L → q9; c:c,L → q10
- q10: a:a,L → q10; b:b,L → q10

On the right side, the 'Multiple Run' tab is active, showing a table of test results:

Input	Result
	Reject
c	Accept
aca	Accept
bcb	Accept
abcab	Accept
aabbacaabba	Accept
ababbabcababbab	Accept
a	Reject
b	Reject
aa	Reject
ab	Reject
acb	Reject
aaacaaaa	Reject
aaaacaaa	Reject

At the bottom of the table, there are buttons for 'Load Inputs', 'Run Inputs', 'Clear', 'Enter Lambda', and 'View Trace'.

Here is a formal definition of this Turing Machine that decides language L:

$$(Q, \Sigma, \Gamma, \delta, q_s, \square, F) = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}\}, \{a, b, c\}, \{a, b, c, 0, \square\}, \delta \text{ as defined in the preceding state diagram, } q_0, \square, \{q_2\})$$

Note that states q6 and q10 could be collapsed into a single state. Likewise, states q5 and q9 could be collapsed into a single state. The result would be a TM with nine states instead of eleven.