NFA – Exercise

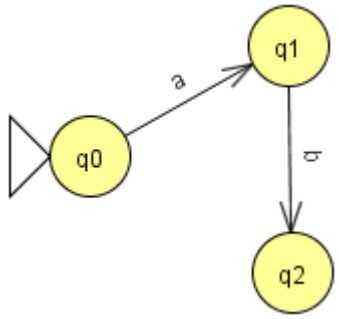
Problem:

> Construct an NFA that accepts the language `{ab, abc}*`. This is the set of strings where *ab* and *abc* may be repeated. Example strings include *abcab*, *ababcab*, *abcabcabc*, and the empty string.

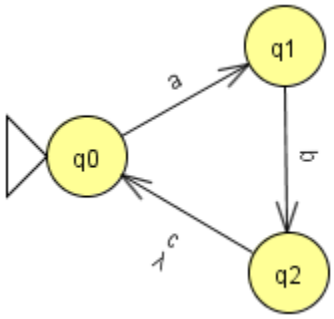
Solution:

We start by analyzing the type of strings accepted by this language. Each substring may be either an *ab* or an *abc*. The substring may be repeated zero or more times. Valid strings include *abab*, *abcababc*, and *ababcabcabc* (respectively, *ab ab*, *abc ab abc*, *ab abc abc abc*, the spaces showing the substrings).

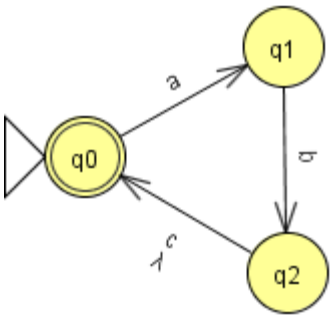
Since the beginning of each substring is an *ab*, we start an NFA with states accepting the ab from either substring.



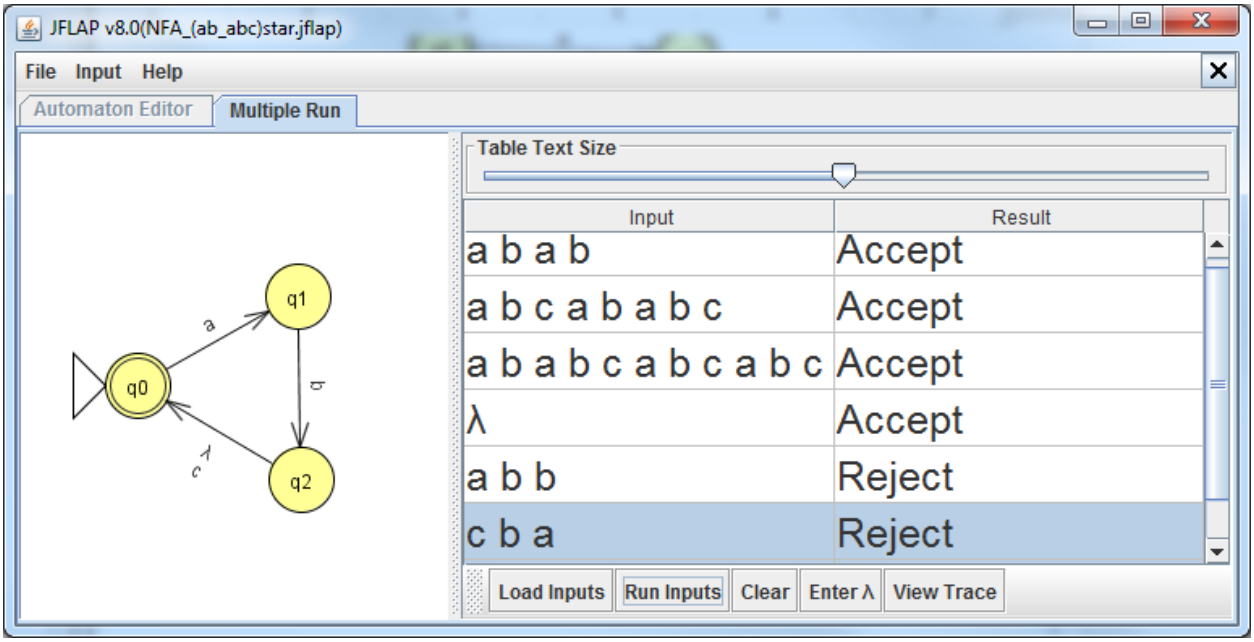
To this partial NFA, we add a transition to accept the *c* in the *abc* substring, or an empty string. We also want a loop back to the initial state so that the star-closure may iterate.

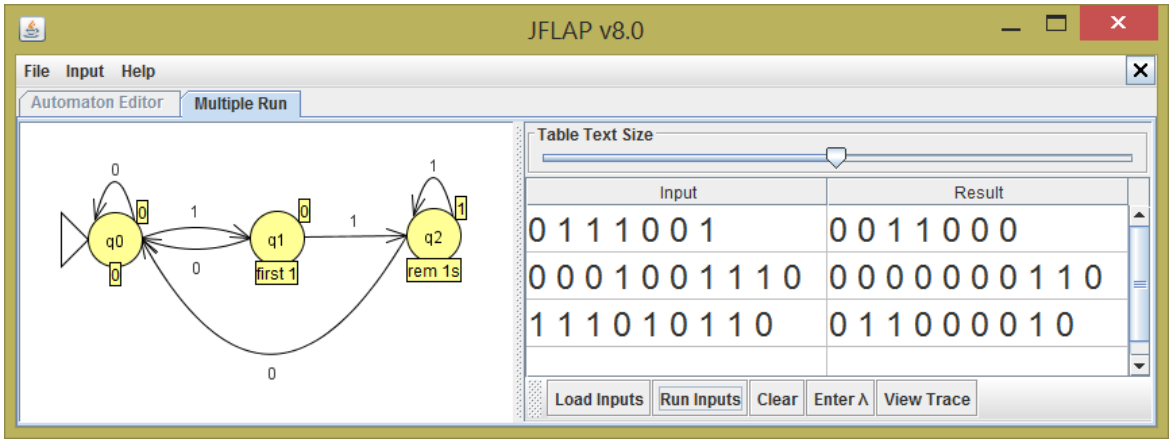


Lastly, we make the initial state a final state to allow it to accept λ and to end anytime a substring is parsed. Note that an NFA does not have to have trap states nor an outgoing edge for each alphabet symbol out of every state.



Test the NFA with sample strings.





Lastly, we explore the non-determinism attribute of the finite automaton further. With the NFA loaded on JFLAP, click *Input* > *Step by State*. Enter ab as the input string. Click *Step* for each of the

input symbols and watch JFLAP simulate the moves. Note that after it reads the “b”, it may stay at q1 and reject the string or transition to q0 to accept.