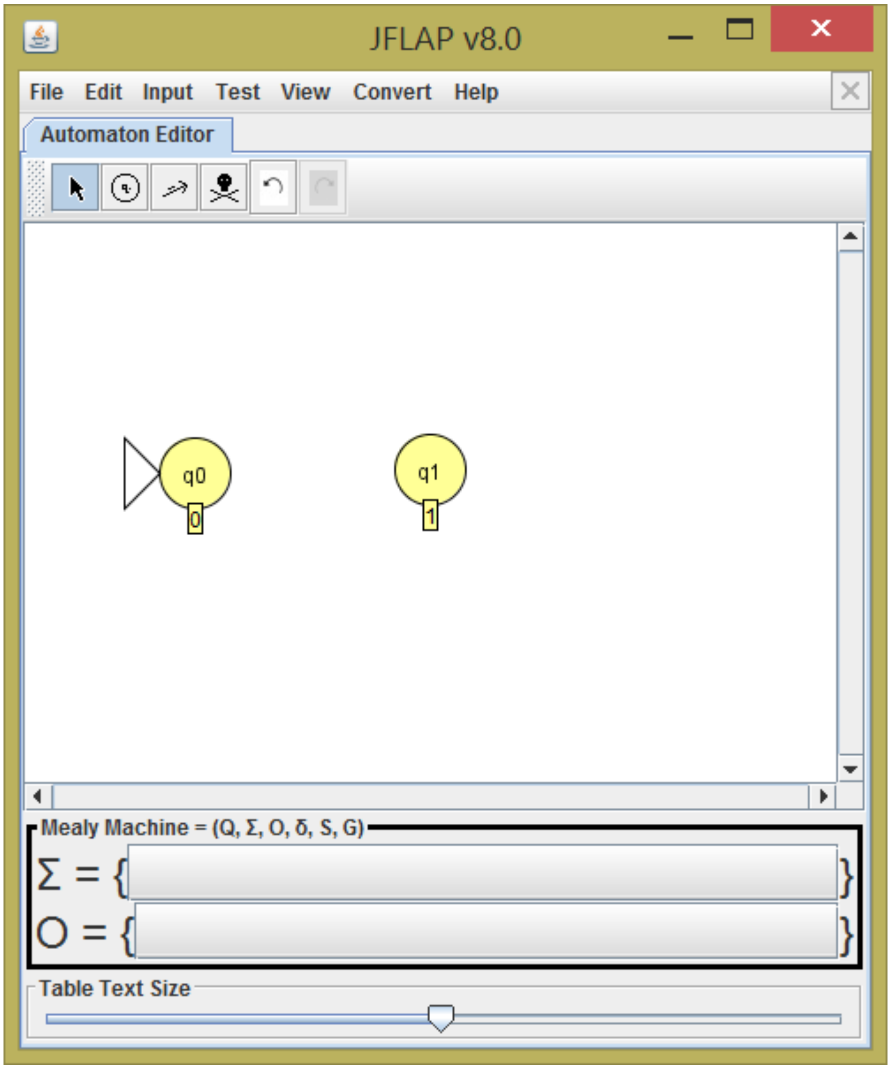
Mealy Machine – Exercise

Problem:

> Construct a Mealy machine which takes a binary number and replaces the first 1 with a 0 from every substring starting with 1. For example, 0001001110 becomes 0000000110. This type of "bit stuffing" may be used in data transmission and telecommunications for run-length coding to limit the number of consecutive bits of the same value. A bit of the opposite value is inserted after the maximum allowed number of consecutive bits.

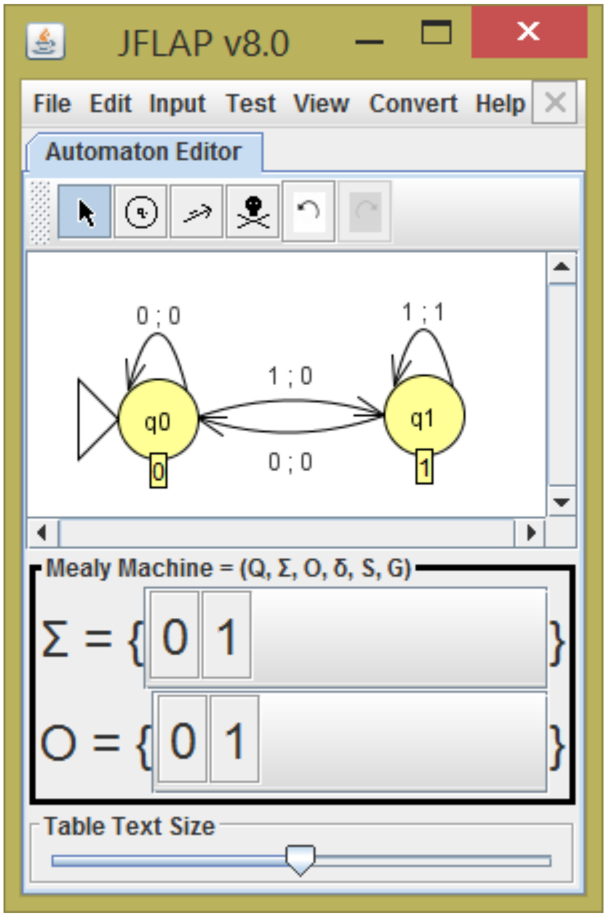
Solution:

Open JFLAP and create a Mealy machine with an initial state. Set its label to a "0" to remember that a zero is read. Next, add a second state to remember that a "1" was read. Label this with a "1".



Four transitions will be needed.

1. At q_0, a 0 is read so a loopback to q_0 is needed. Output a 0.
2. At q_0, a 1 is read so a transition to q_1 is needed. Output a 0 since this is the first 1 in a substring starting with a 1.
3. At q_1, a 0 is read so a transition to q_0 is needed. Output a 0.
4. At q_1, a 1 read so a loopback to q_1 is needed. Output a 1.

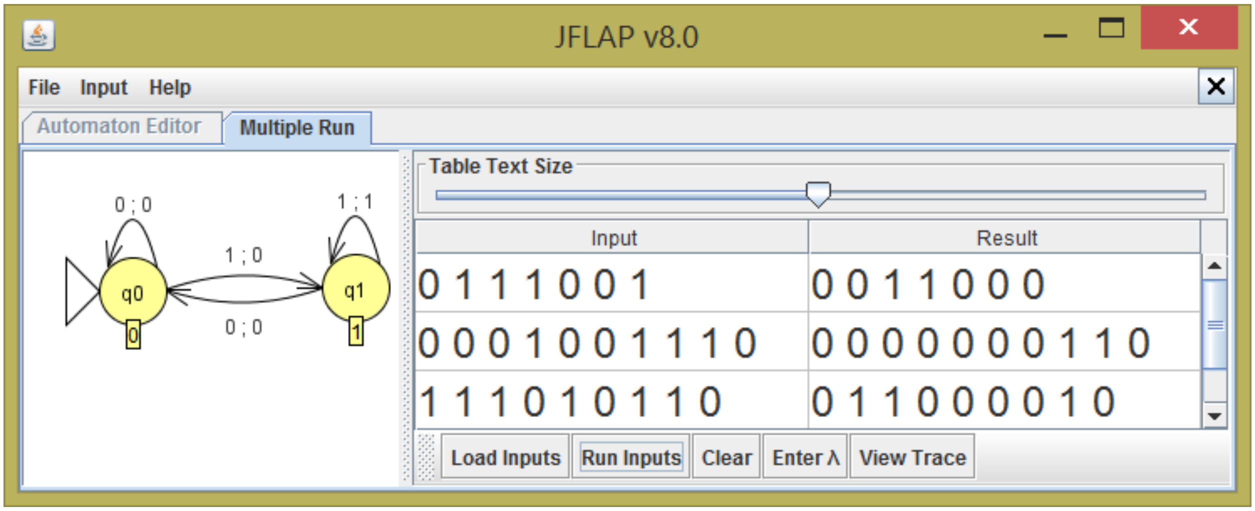


Mealy Machine = (Q, Σ, O, δ, S, G)

Σ = { 0 1 }

O = { 0 1 }

Run some test strings using *Input* > *Multiple Runs*.



Input	Result
0 1 1 1 0 0 1	0 0 1 1 0 0 0
0 0 0 1 0 0 1 1 0	0 0 0 0 0 0 1 1 0
1 1 1 0 1 0 1 1 0	0 1 1 0 0 0 0 1 0

Recall that a Moore machine is state machine whose output is determined solely by its current state while a Mealy machine is a state machine whose output is determined **both** by its current state and its input. In this example, we implement a Mealy machine that uses fewer states than a Moore machine for this same problem because we are able to check both the input and the current state at the same time. A machine which needs to “remember” both would require more states.