The finite automata class is implemented as follows:

* The states, final states, alphabet are implemented as a hashset of strings
* The initial state is a string
* The transitions are defined as a map with the key being a pair of initial state (of that transition) and alphabet char, and value being a set of strings representing the states a fa can go from the pair

To read the file which contains the FA, we read line by line (splitting by “/r/n”), get the line number, then we switch cases depending on the line we have, the content of each line being parsed and places in a vector of Strings.

To check if it is deterministic we check the set of states (value) to have size 1.

To check a sequence if it is accepted or not, we check the FA to be deterministic, check the transition between current state and current char to be in the map, and lastly we check for the final state to be in the list of final states.

FA file

States

Alphabet

initialState

finalStates

transitions

ex:

A B C  
0 1 2 3 4 5 6 7 8 9  
A  
B C  
A 0 B  
A 1 C  
A 2 C  
A 3 C  
A 4 C  
A 5 C  
A 6 C  
A 7 C  
A 8 C  
A 9 C  
C 0 C  
C 1 C

FA file EBNF

state ::= character

states ::= state{state} ”/n”

symbol ::= character

alphabet ::= symbol {symbol} ”/n”

initialState ::= state ”/n”

finalStates :: state{state} ”/n”

transition ::= state ” “ symbol ” “ state “/n”

inputFile ::= states alphabet initialState finalStates transition {transition}