Finite Automaton

Github: https://github.com/andradatiutin/FLCD/tree/master

The Finite Automaton is a class which contains 5 fields: the set of states, the alphabet, the transition function, the initial state, the set of final states. The transition function is represented as a HashMap, where each pair (qi, a) is mapped to a list of destination states.

The methods in the class are: printQ(), printE(), printS(), printF(), isDFA(), isAccepted(String sequence), checkFinalState(String string).

The function isDFA() goes through the list of transitions and checks if there is at least one key in the list of transitions which is mapped to more than 1 states, case when it returns false, otherwise true.

The function checkFinalState(String string) checks if a given state belongs to the set of final states.

The function isAccepted(String sequence) checks if a given sequence is accepted by a DFA.

Scanner with FA: The regex matching for identifiers and integer constants are replaced with checking that the FA accepts a sequence(the one that represents an identifier/integer constant). In the files **identifier.in** and **constant.in** we keep the finite automatons.

EBNF:

```
program = line1 "\n" line2 "\n" line3 "\n" line4 "\n" line5 "\n"
line1 = "Q = {" state {state} "}"
line2 = "E = {" symbol {symbol} "}"
line3 = "S = {" {transition} "}"
line4 = "q0 = " initialState
line5 = "F = {" finalState {finalState} "}"

state = identifier
symbol = constant
transition = "(" identifier "," constant ")" "->" identifier
initialState = identifier
finalState = identifier
```