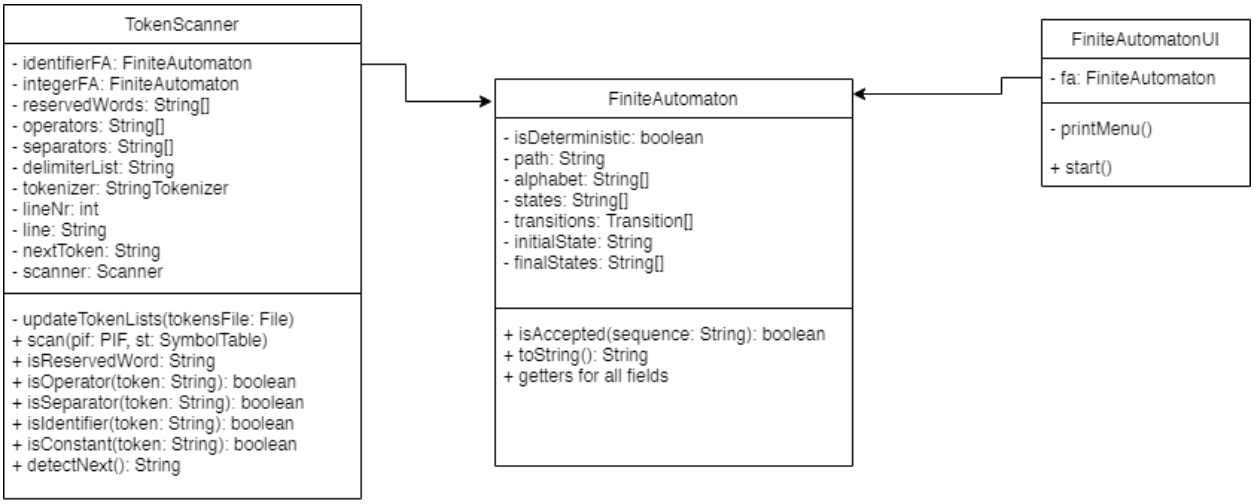
Lab 4 – Trombitas Richard-Alexandru, Group 933

Source code: https://github.com/RichardTrombitas/flcd-projects/tree/lab4



FiniteAutomaton Methods

- isAccepted: verifies whether a given sequence is accepted by the FA or not
  - input: String sequence, output: true / false
- toString: returns a string representation of the FA containing the alphabet, set of states, transitions, the initial state and the set of final states
  - input: -, output: String
- getters for all fields

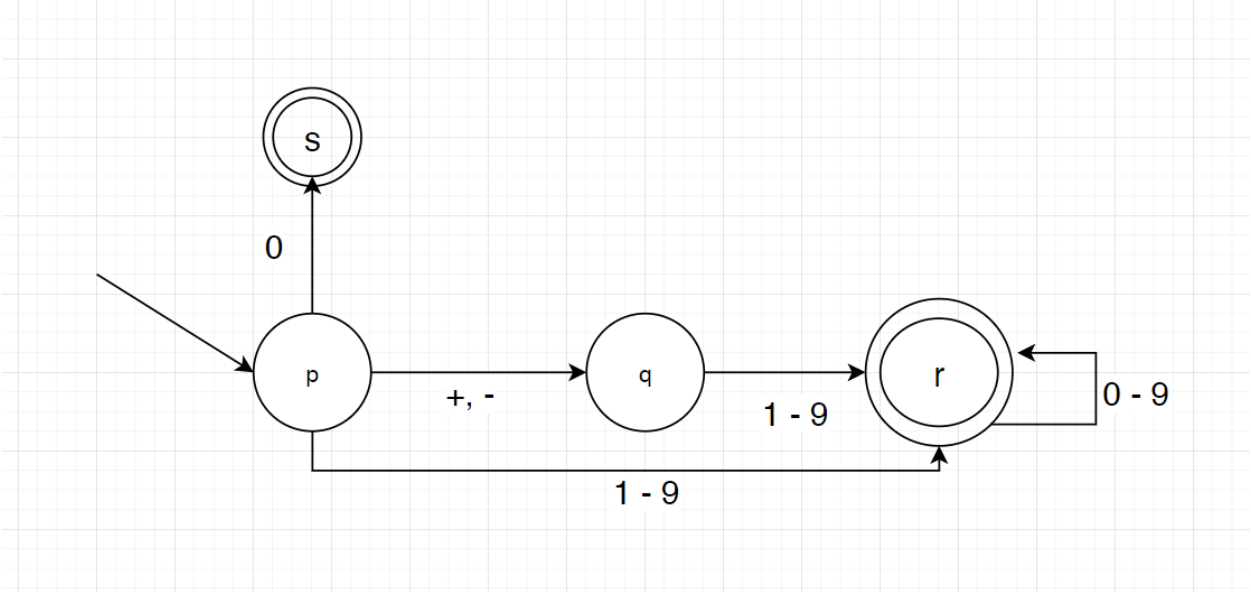
FiniteAutomatonUI Methods

- printMenu: prints a menu of options for the user to choose from (see the set of states, see the alphabet, see transitions, see the set of final states, check if a sequence is accepted, exit)
  - input: -, output: String
- start: starts the execution of the UI, stops when the user chooses exit
  - input: -, output: -

Input file format for a FA

The FA will be represented in a table format. The first line of the input file should contain the elements of the alphabet. The following lines should contain states (or sets of states, put between {} characters and separated by commas). The empty state should be represented as an empty set: {}. Before the first state from a line, there may also be a * symbol representing that the first state from that line is a final state. One or more lines may contain such a symbol. Additionally, there may be a -> symbol in the input file, at the very beginning of a line of states, representing that the first state from that line is the initial state of the FA. All elements will be separated by one or more spaces.

Example of representing integer constants:



fa-integer.in:

	+	-	0	1	2	3	4	5	6	7	8	9
-> p	q	q	s	r	r	r	r	r	r	r	r	r
q	{}	{}	{}	r	r	r	r	r	r	r	r	r
* r	{}	{}	{}	r	r	r	r	r	r	r	r	r
* s	{}	{}	{}	{}	{}	{}	{}	{}	{}	{}	{}	{}

EBNF notation

input = alphabet ”\n” {“ “} stateLines

alphabet = character | (character ” “ {“ “} alphabet)

character = any UTF-8 character

stateLines = stateLine | stateLine ”\n” {“ “} stateLines

stateLine = [“-> ”] {“ “} [“* ”] {“ “} stateList

stateList = state | stateSet | (state | stateSet) “ “ {“ “} stateList

state = (letter | digit | “_”) | (letter | digit | “_”) state

stateSet = “{“[statesWithCommas]”}”

statesWithCommas = state | state “,” {“ “} statesWithCommas