Finite automata documentation

https://github.com/MirceaDragosVlad917/FLCD/tree/main/Lab4

Write a program that:

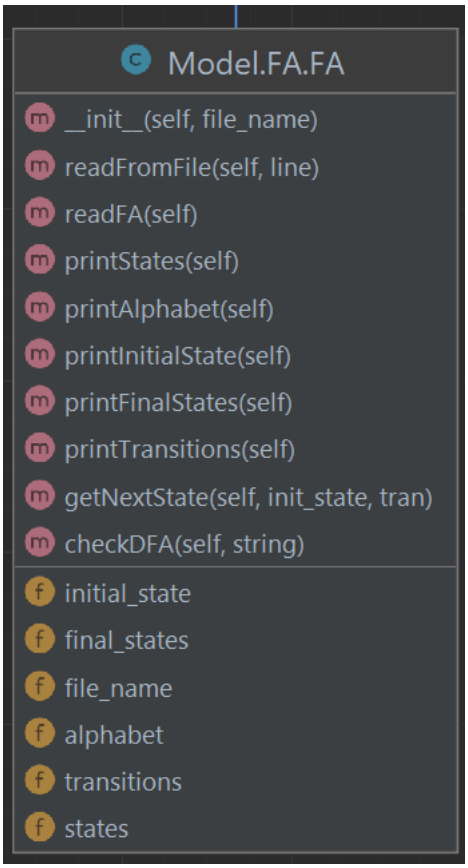
1. Reads the elements of a FA(from file)
2. Displays its elements, using a menu: the set of states, the alphabet, all the transitions, the initial state and the set of final states
3. For a DFA, verify if a sequence is accepted by the FA

Deliverables:

1.FA.in - input file (on Github)
2.Source code (on Github)
3.Documentation. It should also include in BNF or EBNF format the form in which the FA.in file should be written (on Moodle and Github)

Max grade = 9
Max grade = 10: Use FA to detect tokens <identifier> and <integer constant> in the scanner program

Class diagram



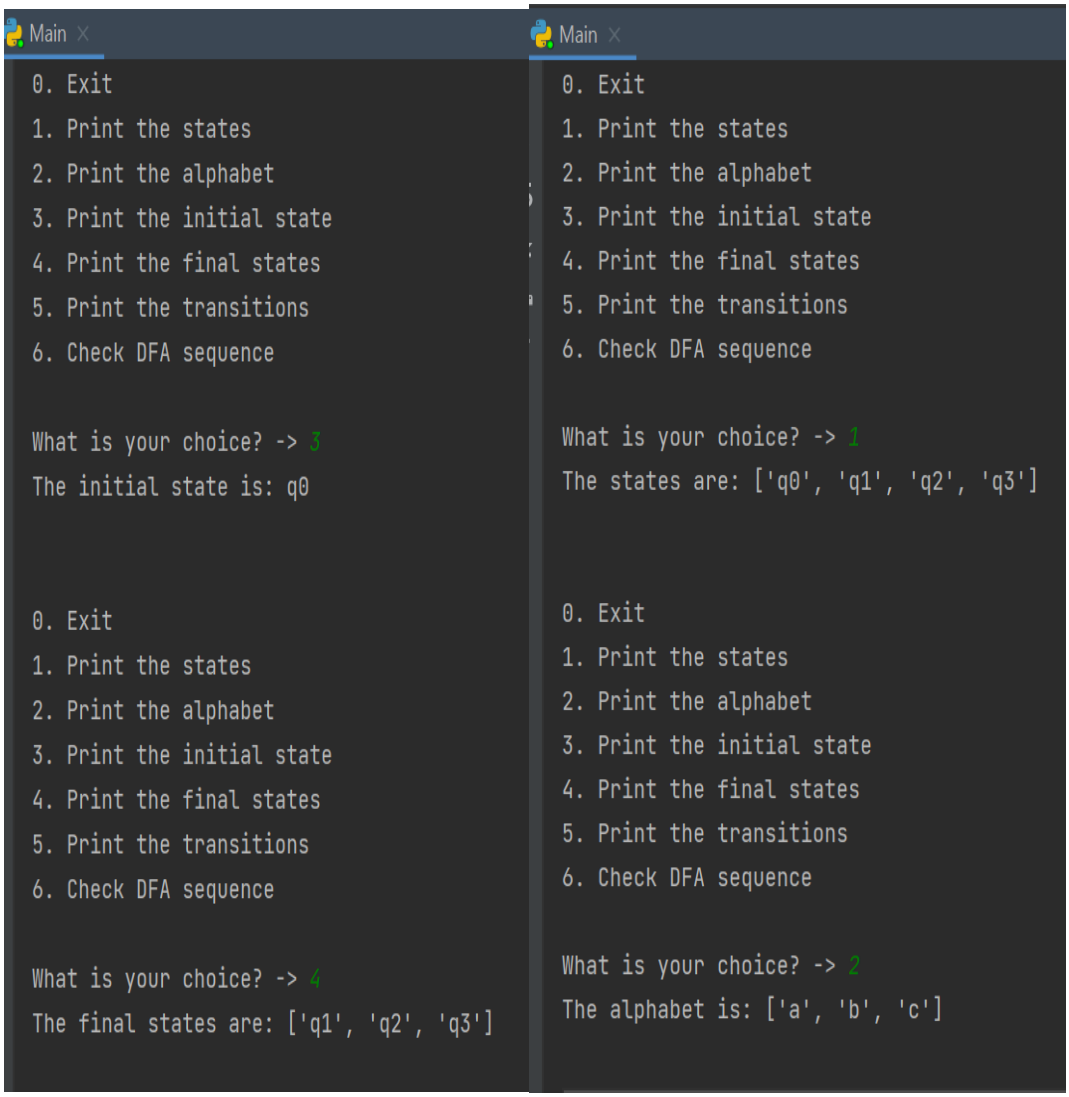
The program reads a finite automata form a file and interprets its set of states, alphabet, initial state, final states and transitions, which are all represented as lists of strings. The checkDFA function receives as parameter a sequence, starts in the initial state, and for every letter in the sequence, it checks if a transition from the current state to another with the letter as label exists. If the final letter is not in the list of final states, or there is not such transition, the word ins not accepted by the finite automata. Otherwise, it is accepted.

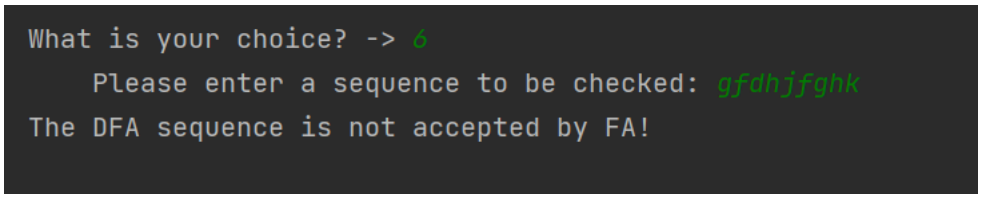
Tests:

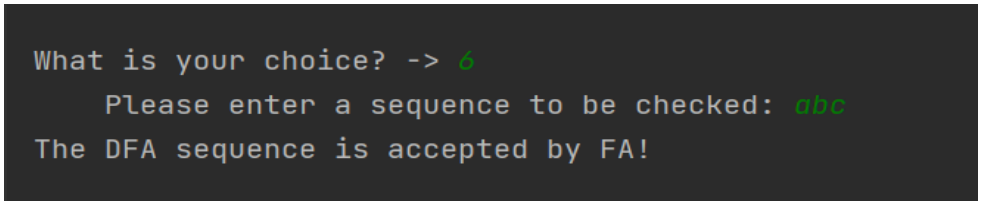
FA.in:

```
States = {q0,q1,q2,q3}
Alphabet = {a,b,c}
InitialState = q0
FinalStates = {q1,q2,q3}
sigma:
    (q0,a) -> {q1}
    (q1,b) -> {q2}
    (q2,c) -> {q3}
```

This is the result of a run of the program on the file:







nzdigit:= 1 | 2 | … | 9
digit:= 0 | 1 | 2 | … | 9
sign:= + | -

integer:= [sign]nzdigit{digit} | 0
letter:= a | b | … | z | A | B | … | C
identifier:= letter{(letter | digit | _)}
character:= letter | digit
characterList:= {character} {“,” character}
states:= “States” “=” “{” characterList “}”
alphabet:= “Alphabet” “=” “{” characterList “}”
initialState:= “InitialState” “=” characterList
finalStates:= “FinalStates” “=” “{” characterList “}”
transition:= “(”characterList“,”characterList“)“->” “{”characterList“}”
transitions:= “Transitions” “=” {transition} “\n” {transition}
FA.in:= states “\n” alphabet “\n” initialState “\n” finalStates “\n” “sigma:” “\n” transitions