GitHub: https://github.com/Raiu-Madalin-Augustin/FLCD

The Finite Automaton (FA) is represented as 5 components, each one being a list or dictionary, that contain these sp
ecific
elements: the set of states (Q), the alphabet (E), the initial state (q0), the set of final states (F) and the transitions
(T). Compared to the others, the set of transitions is actually a dictionary of lists, each key showing
a pair of a state and element from the alphabet and the list tied to it has the destination states
(e.g. key (a,1): list[b,c] = a goes to b or c using the value 1)

is_dfa():
    -verifies if the current FA is actually a DFA (if each list in the set of transitions has their length 1, then it is a DFA
)

def is_accepted_by_fa(self, sequence):
    -checks if the given sequence is accepted by the current FA (use the transitions to form a path to the destination an
d check if it is in the set of final states)
    -param sequence: is of type string

def read_input_file(self, filename):
    -reads the components of the FA (the way they are structured is found in "SyntaxFa.in") which will be used to che
ck elements from the program (identifiers, constant integers)
    -filename is the name of the file which the data about the FA components will be taken from
    -param filename: is of type string

def is_fa_valid(self):
    -verifies if the current FA is a valid one by checking the contents of its components and see if some of them (e.g. i
nitial state, final state)
    are found in the others (e.g. set of states)

The EBNF format that is the form in which the FA.in is written it looks as follows (also found in file “SyntaxFa.in”):

    big_letter = "A" | "B" | . ..| "Z"
    small_letter = "a" | "b" | . ..| "z"
    letter = big_letter | small_letter
    non-zero = "1" |...| "9"
    digit = "0" | non-zero
    state = letter [digit] | letter | digit
    alphabet_element = letter | digit
    states_int = state | state "," states_int
    states = "Q" "=" "{" states_int "}"
    alphabet_int = alphabet_element | alphabet_element "," alphabet_int
    alphabet = "E" "=" "{" alphabet_int "}"
    initial_state = "q0" "=" {" state "}"
    final_states = "F" "=" "{" states_int "}"
    transition = "(" state "," alphabet_element ")" "-" ">" state
    transitions_int = transition | transition NEWLINE transitions_int
    transitions = "T" "=" "{" transitions_int "}