Documentation Lab 4

Link to GitHub repository:
https://github.com/GeorgeDanicico/Formal-Languages-and-Compiler-Design

The finite automaton has the following fields: initial state that is represented as a string, alphabet and final states which are represented as a set of strings each, and transitions, which is represented as a map, where the key is a pair between the initial state of the transition and an element from the alphabet, and the value is a set of strings representing the destination states.

The finite automaton is read from a file.
In order to check if a sequence is accepted by the FA defined in the file(assuming that the FA in the file is also a DFA), we start from the initial state and the first character of the sequence, and check if there exists an entry where the key is a pair of the initial state and the first character in the transition map. If it does not exists, means that the sequence is not accepted by the FA and the process stop here. If there is such an entry, we update the state with the value of the entry and move on to the second character and repeat the aforementioned process until the end of the sequence. If we reached the end of the sequence and the state we obtained is not present in the final states set or if the process was stopped earlier, we return false, otherwise we return true.

letter = “a” | … | “z” | “A” | … | “Z”
digit = “1” | … | “9”
sign = “+” | “-“
alphabet_character = letter | digit | sign
type = “initial” | “final”
state = letter digit

declared_state = state “,” type “\n”
transition = state “,” alphabet_character “,” state “\n”

File = declared_state (declared_state)+ (transition)+