Source code:

https://github.com/915-Motoc-Simona/FLCD/tree/main/Lab4

The Finite Automata is a class that has 5 fields:

- Q -> set of states - List of Strings
- E -> alphabet - List of Strings
- q0 -> initial state - String
- F -> set of final states – List of Strings
- T -> transitions – HashMap with the key being a Pair of <String, String> and a List of Strings for the value

For the DFA checking, we look through all the transitions and see if there is a key that has associated a list with more than 1 element.

A sequence is accepted by the Finite Automata if starting from the initial state it can reach one of the final states following the transitions and when a final state is reached there are no other transitions left (the sequence is empty). This is done by going character by character in the sequence and checking if there exists a key in the HashMap of the form (currentState, sequence.charAt(i)), where i is the current character in the sequence, i.e. checking if there exists a transition.

EBNF:

character = “a” | “b” | ... | “z” | “0” | “1” | ... | “9”

word = character {character}

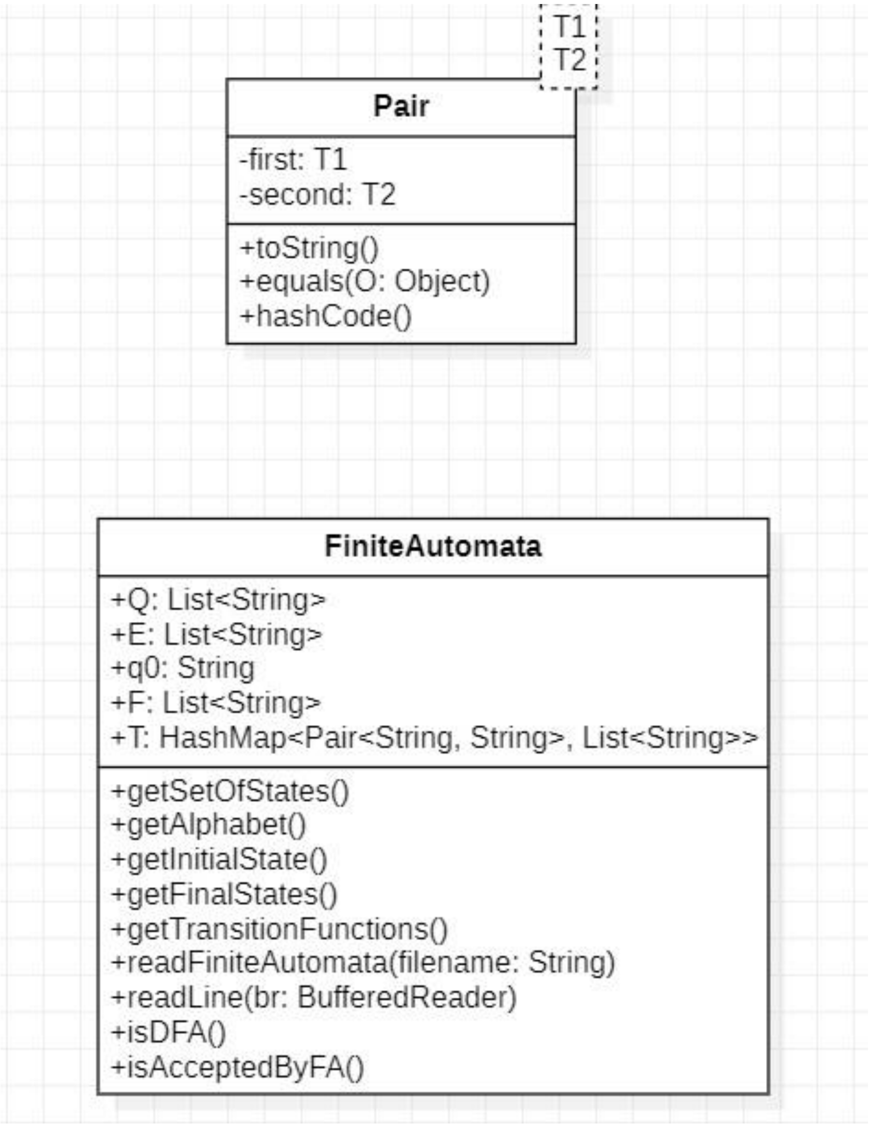
states = word {word}

initialState = word

finalStates = word {word}

alphabet = word {word}

transitions = word word word



T1
T2

Pair

-first: T1
-second: T2

+toString()
+equals(O: Object)
+hashCode()

FiniteAutomata

+Q: List<String>
+E: List<String>
+q0: String
+F: List<String>
+T: HashMap<Pair<String, String>, List<String>>

+getSetOfStates()
+getAlphabet()
+getInitialState()
+getFinalStates()
+getTransitionFunctions()
+readFiniteAutomata(filename: String)
+readLine(br: BufferedReader)
+isDFA()
+isAcceptedByFA()