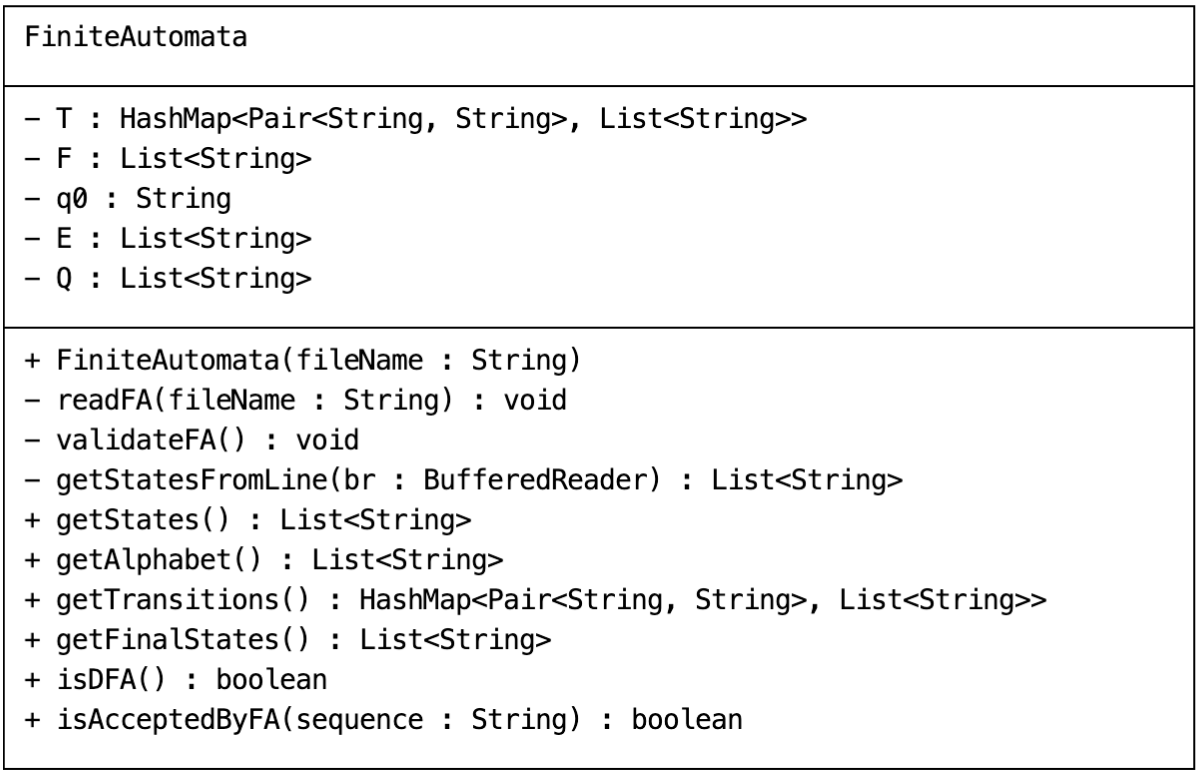
https://github.com/donpaul999/Assignments_UBB/tree/master/3rd%20Year/FLCD/Lab_4

The Finite Automaton is structured as a class with 5 fields: Q, E , q0, F, T.

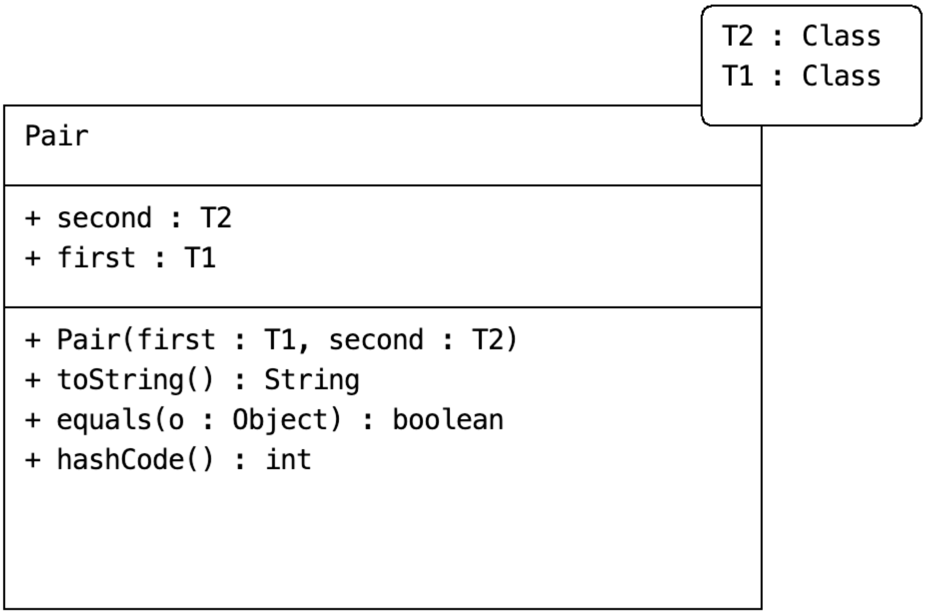
The transitions T are kept in a HashMap, where each pair (q, a) is mapped to a list of destination states, for example: (q, 1)->p, meaning q goes to p with value 1.



```
FiniteAutomata

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

+ FiniteAutomata(fileName : String)
- readFA(fileName : String) : void
- validateFA() : void
- getStatesFromLine(br : BufferedReader) : List<String>
+ getStates() : List<String>
+ getAlphabet() : List<String>
+ getTransitions() : HashMap<Pair<String, String>, List<String>>
+ getFinalStates() : List<String>
+ isDFA() : boolean
+ isAcceptedByFA(sequence : String) : boolean
```



```
T2 : Class
T1 : Class

Pair

+ second : T2
+ first : T1

+ Pair(first : T1, second : T2)
+ toString() : String
+ equals(o : Object) : boolean
+ hashCode() : int
```

Checking whether the FA is a DFA is done by looking through all the dictionary keys and looking for lists longer than 1.

In order for the FA to accept a sequence, it goes through each symbol and checks that the respective point can be reached by following the FA transitions.

EBNF

```
states = word { word }
initialState = word
finalStates = word { word }
alphabet = word {word}
transitions = word word word
word = character {character}
character = “a” | “b” | ... | “z” | “A” | “B” | … | “Z” | “0” | “1” | “2” | “3” | “4” | “5” | “6” | “7” | “8” | “9”
```