Finite Automata

Dacian Florea, 933
Github: https://github.com/dacianf/FLDC/tree/main/lab_fa

Implementation:

Finite Automata class has the following fields:

```
    - String[] states          -- the set of all states
    - String[] alphabet        -- the alphabet
    - String initialState      -- the initial state
    - Dictionary<Tuple<String, String>, String[]>   --
all transition pairs
    - String[] finalStates     -- the set of final states
```

Transitions are stored inside a dictionary with:
- KEY: tuple from a state and an element from the alphabet
- VALUE: list of strings where a string is a state that can be reach from the key

Methods:

- public parseLine(String line)
  - -PRE:
    - fileName - a valid sequence of characters
  - -POST:
    - given line is split in a list of tokens
  - -RETURN:
    - String[] tokens — list with the tokens from the given line

- public fromFile(String fileName)
  - -PRE:
    - fileName - a valid sequence of characters which is a valid file name
  - -POST:
    - a finite automata is created from the given file
  - -RETURN:
    - FiniteAutomata — a finite automata object

- public parseTransition(String[] parts)
  - -PRE:
    - parts - a list of strings which are all transitions from the input fa
  - -POST:
    - a dictionary with the given transitions is created
  - -RETURN:
    - Dictionary<Tuple<String, String>, String[]> transitions — dictionary with all transitions

- public isState(String state)
  -PRE:
  state — a valid sequence of characters
  -POST:
  -
  -RETURN:
  true - if given state exists in the states list
  false - otherwise

- public getTransitionsFor(String state)
  -PRE:
  state — a valid sequence of characters
  -POST:
  -
  -RETURN:
  String[] - list with all transitions for a given state
  throws error - if the given state does not exists

- public showTransitionsFor(String state)
  -PRE:
  state — a valid sequence of characters
  -POST:
  prints all the transitions for a given state
  -RETURN:
  -

- public showAllTransitions()
  -PRE:
  -
  -POST:
  prints all the transitions for the current FA
  -RETURN:
  -

- public isDFA()
  -PRE:
  -
  -POST:
  -
  -RETURN:
  true - if given current FA is DFA
  false - otherwise

- public isAccepted(String sequence)
  -PRE:

sequence — a valid sequence of characters
-POST:
-
-RETURN:
true - if given sequence is accepted by FA
false - otherwise

- public nextState(String currentState, String currentChar)
  PRE:
  currentState — a valid state
  currentChar — a valid character
  -POST:
  -
  -RETURN:
  String state - returns the state from the given pair of state and char
  None - if there is no state from the given pair of state and char

The finite automata is read from a file an has the following EBNF form:

```
   states = 'Q' '=' '{' stateIdentifier {',' stateIdentifier } '}'
   alphabet = 'E' '=' '{' charIdentifier {',' charIdentifier } '}'
   initialState = 'q0' '=' stateIdentifier
   finalStates = 'Q' '=' '{' stateIdentifier {',' stateIdentifier } '}'
   transitions = 'S' '=' '{' '\n'  { transitionStateIdentifier } '}'
   transitionStateIdentifier = '(' stateIdentifier ',' charIdentifier ')' '->' stateIdentifier ','
'\n'
   stateIdentifier = 'q' digit {digit}
   digit = '0-9'
   charIdentifier = digit | 'a-z' | 'A-Z' | { digit | 'a-z' | 'A-Z' }
```