https://github.com/taniasasaran/UBB-Computer-Science/tree/main/Semester-5/FLCD/Lab4

Transition – class to represent the transitions in the finite automata. It has the following fields:

- String fromState - the state from which the transition is made
- String toState - the state to which the transition is made
- String symbol - the symbol that triggers the transition

FiniteAutomata – class to represent a finite automata. It has the following attributes:
- String filename - the name of the file from which the finite automata is read
- ArrayList<String> states – the set of states of the finite automata
- ArrayList<String> alphabet – the set of symbols that can be used in the transitions
- String initialState – the initial state of the finite automata
- ArrayList<String> finalStates – the set of final states of the finite automata
- ArrayList<Transition> transitions – the set of transitions of the finite automata

It has the following methods:
- FiniteAutomata(String filename) – constructor
- private void readFromFile() – reads the finite automata from the file
- public boolean isDeterministic() – checks if the finite automata is deterministic
- private ArrayList<Transition> getTransitionsFromStateAndSymbol(String state, String symbol) – returns the transitions that start from the given state and have the given symbol
- private void printElements(List<String> elements) – prints the elements of a List<String>
- private void printStates() – prints the states of the finite automata
- private void printAlphabet() – prints the alphabet of the finite automata
- private void printFinalStates() – prints the final states of the finite automata
- private void printTransitions() – prints the transitions of the finite automata
- private void printInitialState() – prints the initial state of the finite automata
- private void printMenu() – prints the menu for displaying the elements of the finite automata
- public void displayThings() – displays a menu and reads the input from the user, then it displays the elements of the finite automata based on the input
- public boolean checkAccepted(String sequence) – checks if a sequence is accepted by the finite automata

FA.in format(BNF):

```
<file> ::= <states> "\n" <alphabet> "\n" <initial_state> "\n" <final_states> "\n" <transitions>
<states> ::= <state> | <state> " " <states>
<initial_state> ::= <state>
<final_states> ::= <state> | <state> " " <final_states>
<alphabet> ::= <symbol> | <symbol> " " <alphabet>
<transitions> ::= <transition> | <transition> "\n" <transitions>
<transition> ::= <from_state> " " <symbols> " " <to_state>
<from_state> ::= <state>
<to_state> ::= <state>
<symbols> ::= <symbol> | <symbol> "," <symbols>
```

(integer.in)

<state> ::= "p" | "q" | "s" | "t"

<symbol> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "-"

(identifier.in)

<state> ::= "p" | "q"

<symbol> ::= "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" | "_" | "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"