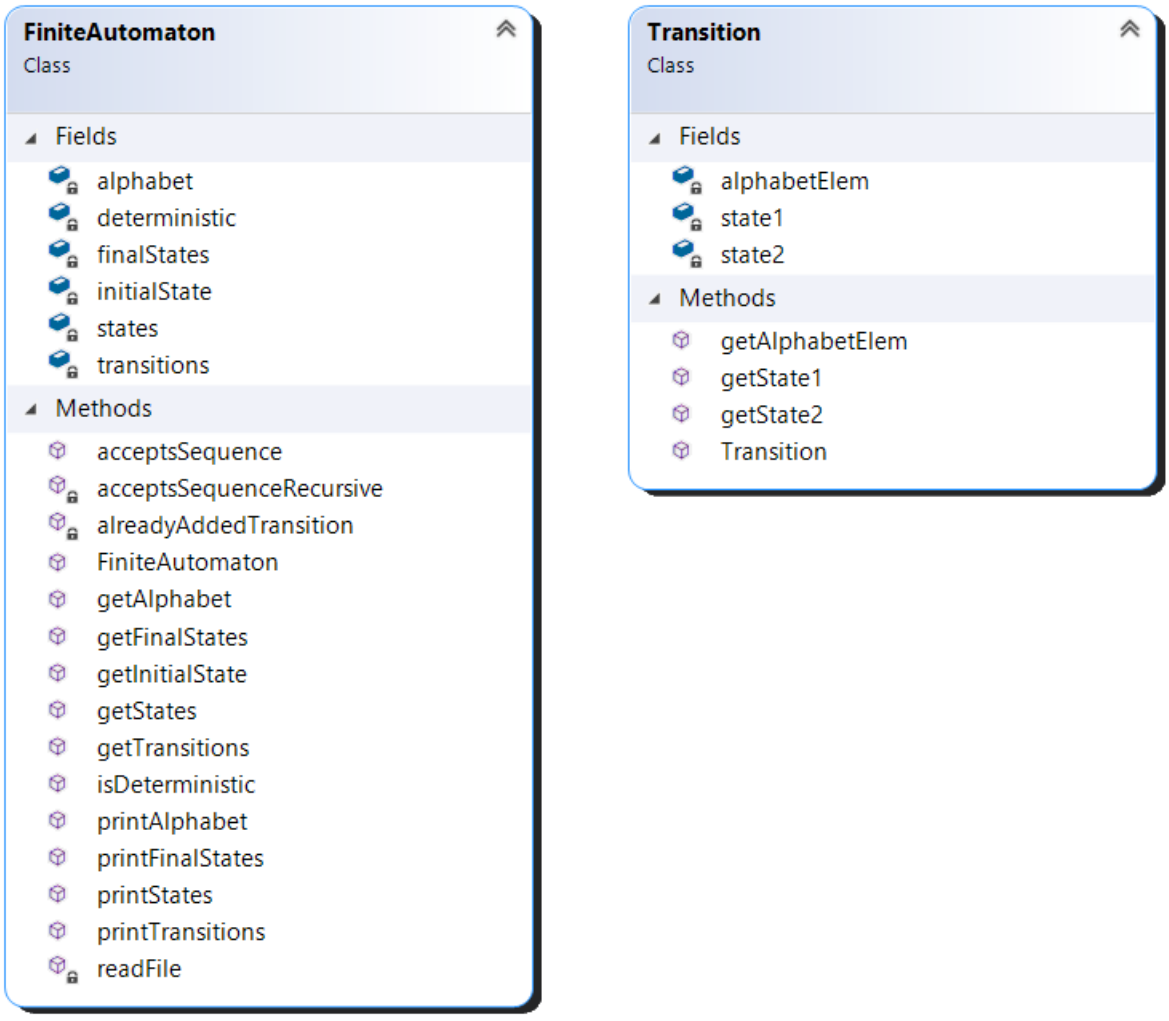
# Finite Automaton

## *Requirement:*

Write a program that:

1. Reads the elements of a FA (from file)
2. Displays the elements of a finite automata, using a menu: the set of states, the alphabet, all the transitions, the set of final states.
3. For a DFA, verify if a sequence is accepted by the FA.

## *Class Diagram:*



## *Implementation details:*

### *Github:*

https://github.com/adabirtocian/Scanner_FLCD/tree/finite-automaton-lab4/FiniteAutomaton

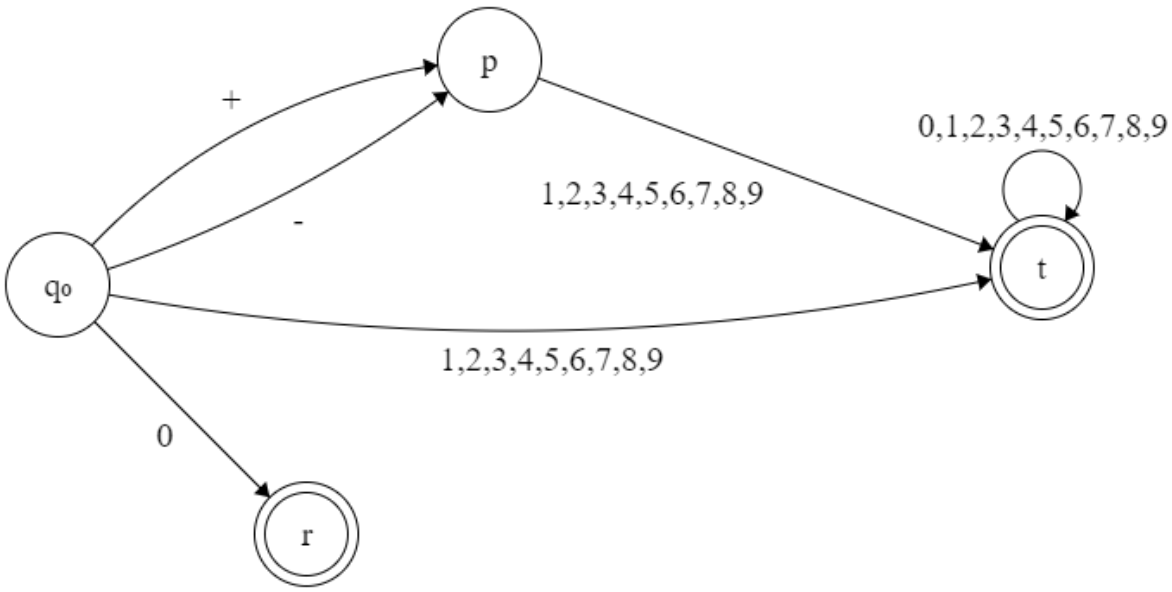
Methods:

- Constructor `FiniteAutomaton(std::string fileName);`
  - Parameters:
    - *fileName*: the file that inputs the initial state, the final states and the transitions
- `bool isDeterministic();`
  - returns true or false whether the automaton is deterministic or not
- `bool acceptsSequence(std::string sequence);`
  - Parameters:
    - *sequence*: the sequence of alphabet elements to check for acceptance
  - returns true or false whether the automaton accepts the sequence or not
- `bool alreadyAddedTransition(Transition transition);`
  - Parameters:
    - *transition*: the transition with 2 states and 1 alphabet element
  - returns true or false whether the given transition is in the set of transitions

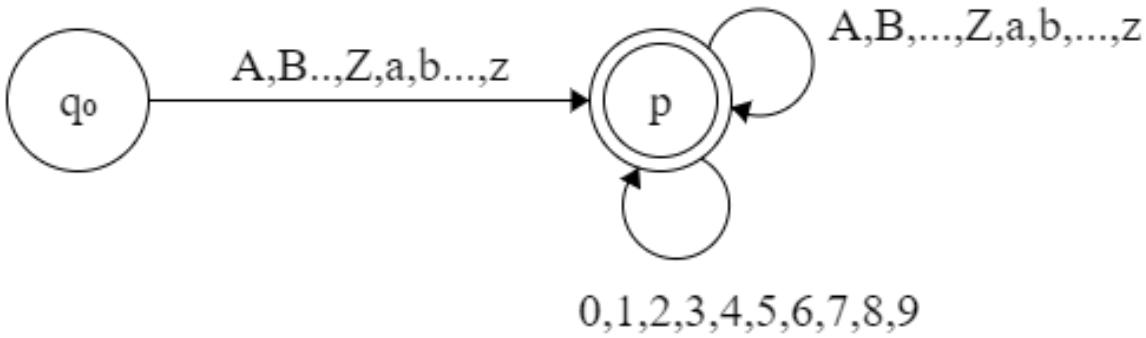
Integration with Scanner:

The finite automaton is also integrated in the Scanner for checking the integer constants and the identifiers. The corresponding files are *intergers.in* and *identifiers.in.*

The integer constants automaton looks like this:



The identifiers automaton looks like this:




## *Input file:*

The input file must follow the following structure (EBNF) and the content is interpreted as strings.

```
file ::= initialState number finalStates number transitions
transitions ::= {state alphabetElem state}
finalStates ::= state newline {state newline}
state ::= "A" | "B" | … | "Z" | "a" | "b" | … | "z"
alphabetElem ::= "A" | "B" | … | "Z" | "a" | "b" | … | "z"
newline ::= '\n'
digit ::= "0" | "1" |...| "9"
number ::= nonZeroDigit{digit}
nonZeroDigit ::= "1" |...| "9"
```

## *Testing:*

- Using the finite automaton for integer constants, sequence *12* is accepted, as well as *-9, 0, -12, +34, 5655, 2*, but sequences like *-0, +0, 012* are not accepted
- Using the finite automaton for identifiers, sequences ab, ABZ, aB12, A1 are accepted, but the sequences *_a, 12f, 12* are not accepted