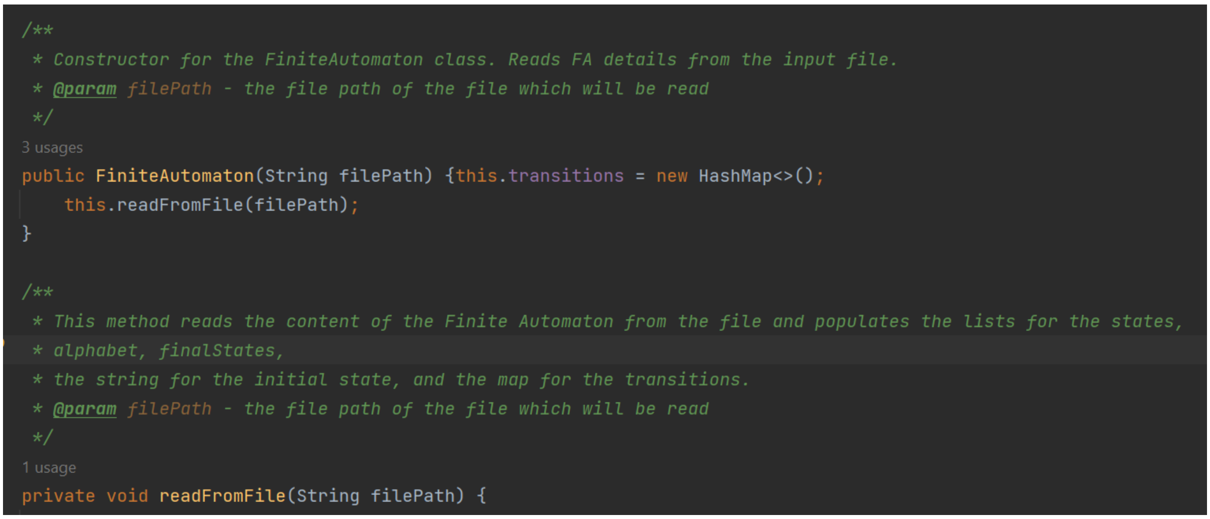
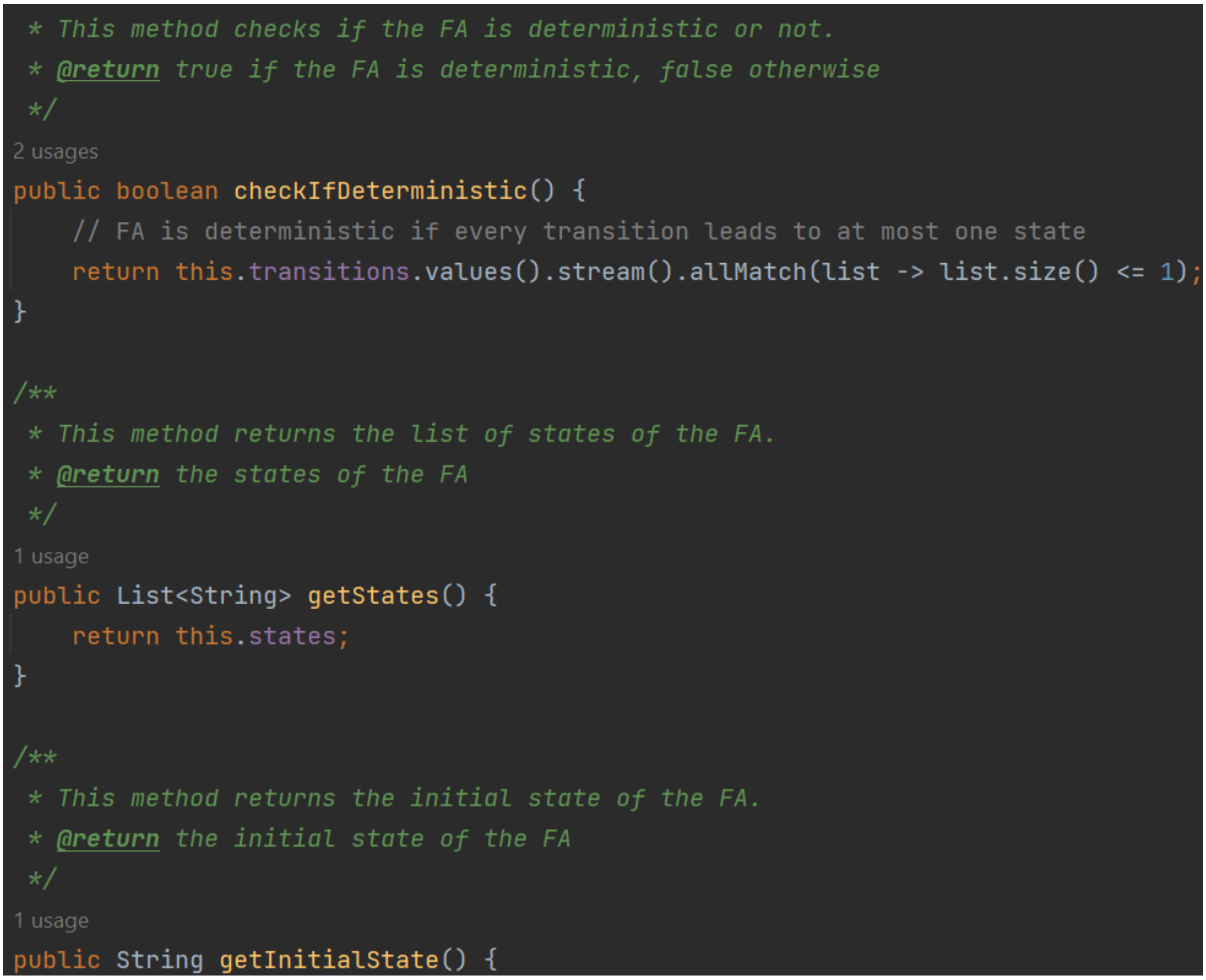
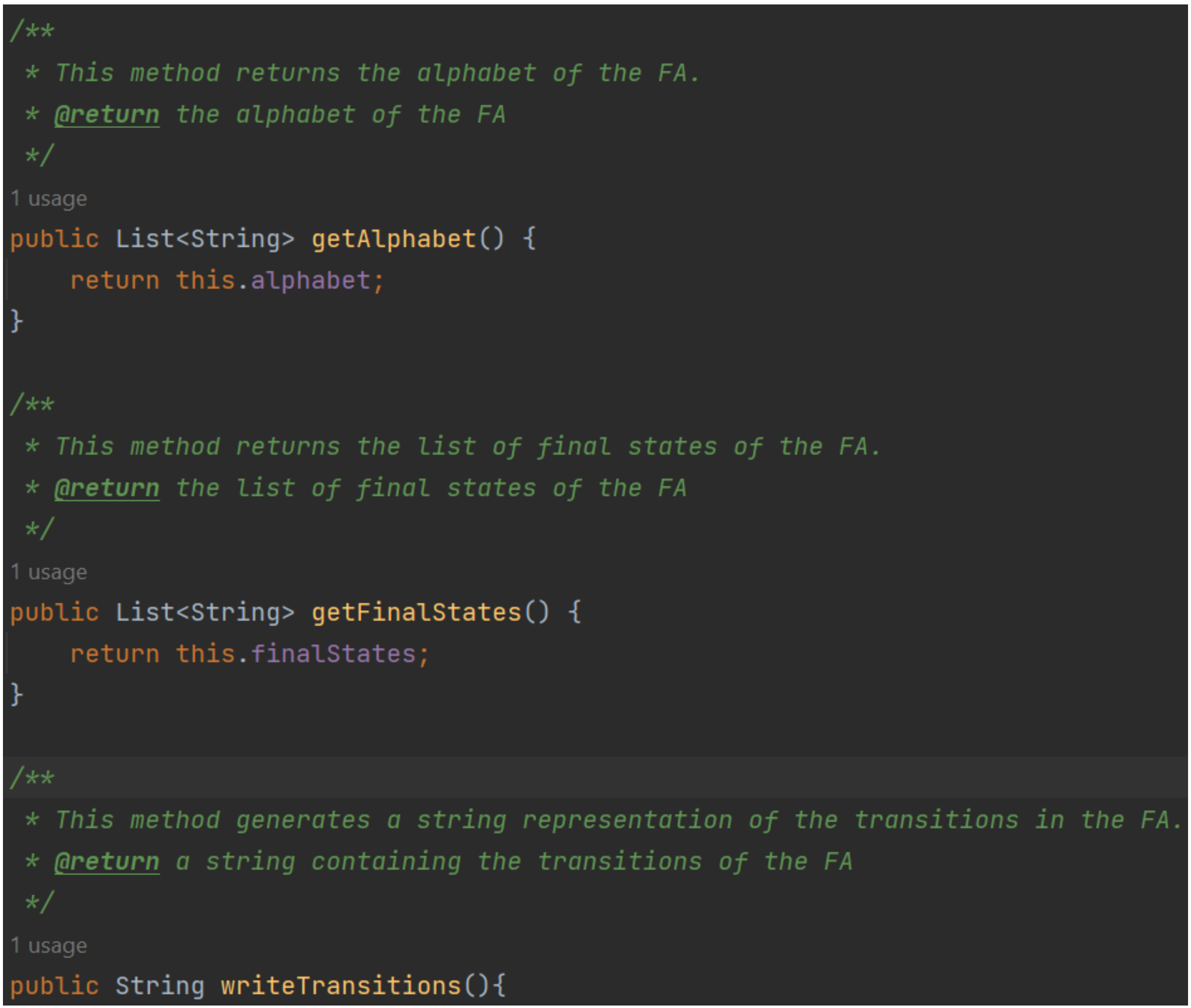
https://github.com/RazvanAndreiMoga/LFTC



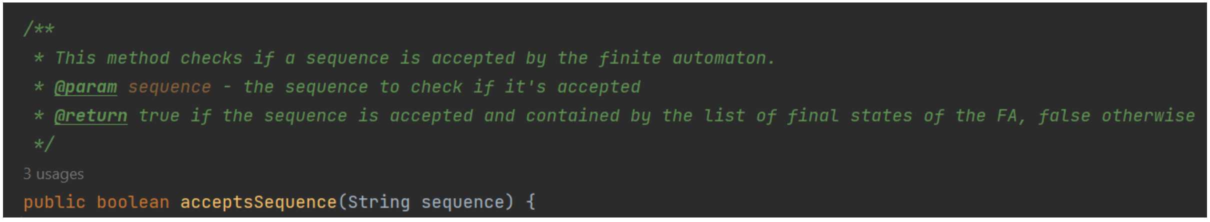
```
/**
 * Constructor for the FiniteAutomaton class. Reads FA details from the input file.
 * @param filePath - the file path of the file which will be read
 */
3 usages
public FiniteAutomaton(String filePath) {this.transitions = new HashMap<>();
    this.readFromFile(filePath);
}

/**
 * This method reads the content of the Finite Automaton from the file and populates the lists for the states,
 * alphabet, finalStates,
 * the string for the initial state, and the map for the transitions.
 * @param filePath - the file path of the file which will be read
 */
1 usage
private void readFromFile(String filePath) {
```



```
 * This method checks if the FA is deterministic or not.
 * @return true if the FA is deterministic, false otherwise
 */
2 usages
public boolean checkIfDeterministic() {
    // FA is deterministic if every transition leads to at most one state
    return this.transitions.values().stream().allMatch(list -> list.size() <= 1);
}

/**
 * This method returns the list of states of the FA.
 * @return the states of the FA
 */
1 usage
public List<String> getStates() {
    return this.states;
}

/**
 * This method returns the initial state of the FA.
 * @return the initial state of the FA
 */
1 usage
public String getInitialState() {
```



```
/**
 * This method returns the alphabet of the FA.
 * @return the alphabet of the FA
 */
1 usage
public List<String> getAlphabet() {
    return this.alphabet;
}

/**
 * This method returns the list of final states of the FA.
 * @return the list of final states of the FA
 */
1 usage
public List<String> getFinalStates() {
    return this.finalStates;
}

/**
 * This method generates a string representation of the transitions in the FA.
 * @return a string containing the transitions of the FA
 */
1 usage
public String writeTransitions(){
```



```
/**
 * This method checks if a sequence is accepted by the finite automaton.
 * @param sequence - the sequence to check if it's accepted
 * @return true if the sequence is accepted and contained by the list of final states of the FA, false otherwise
 */
3 usages
public boolean acceptsSequence(String sequence) {
```

Lexic + Syntax for how we write the FA file:

letter ::= 'a'|'b'|...|'z'|'A'|'B'|...|'Z'

specialSymbol ::= '-'|'+'

digit ::= '0'|'1'|...|'9'

alphabetCharacter ::= letter | digit | specialSymbol

alphabet ::= alphabetCharacter;{alphabetCharacter;}

state ::= letter

listOfStates ::= state;{state;}

transition ::= state alphabet state

listOfTransitions ::= transition'\n'{transition'\n'}

initialState ::= state

finalStates ::= state;{state;}

FAfile ::= listOfStates '\n' alphabet '\n' initialState '\n'finalStates '\' transition