CS112 (LFA) - Projects Lab 2

March 2022

Exercise 1. *(1p) Extend the library/program you implemented in L1.Ex1 to load and validate a NFA input file of the format presented in the Appendix.*

```
nfa_parser_engine.py  nfa_config_file
```

Exercise 2. *(1p) Implement a library/program in a programming language of your choosing to test acceptance of a NFA - loaded from a NFA config file.*

```
nfa_acceptance_engine.py  nfa_config_file <word_to_test>
```

Exercise 3. *(1p) Implement a library/program in a programming language of your choosing to convert a NFA - loaded from a NFA config file, to a DFA.*

```
nfa_conversion_engine.py  nfa_config_file
```

The above command should print the resulted DFA in the format presented in L1.Appendix

Exercise 4. *(2p, Bonus) Implement a library/program in a programming language of your choosing to test acceptance of an* $\boldsymbol{\varepsilon - NFA}$.

```
e_nfa_acceptance_engine.py  e_nfa_config_file <word_to_test>
```

Appendix

NFA input file must be of the following format:

```
#
# comment lines (skip them)
#
Sigma:
    letter1
    letter2
    ...
End
#
# comment lines (skip them)
#
States:
    state1
    state2
    state3,F
    ...
    stateK,S
    ...
End
#
# comment lines (skip them)
#
Transitions:
    stateX,letterY,stateZ
    stateX,letterY,stateZ
    ...
End
```

Sections can be in any order. By validation we ask to check that transition section has valid states (first and third word) and valid letters (word two).

Note that states can be succeeded by ”F”, ”S”, both or nothing. ”S” symbol can succeed only one state.

For $\varepsilon - NFA$ we make the convention that ”*” is the epsilon character, and the alphabets will never use ”*” as a letter.