German University in Cairo
Department of Computer Science
Assoc. Prof. Haythem O. Ismail

CSEN1002 Compilers Lab, Spring Term 2022
Task 1: DFA

Due: Week starting 05.03.2022

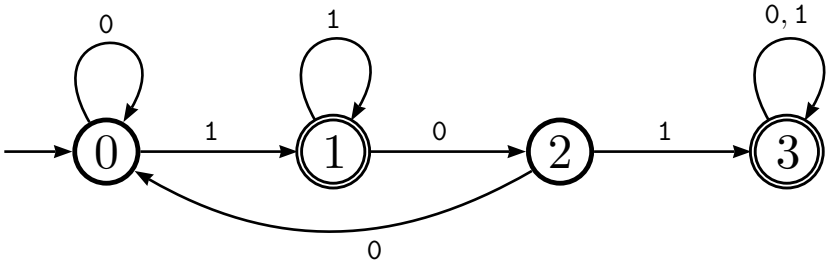
1 Objective

For this task you need to implement a deterministic finite automaton (DFA) abstract data type. Recall that a DFA is a quintuple $(Q, \Sigma, \delta, q_0, F)$: Q is a non-empty, finite set of states; Σ is non-empty, finite set of symbols (an alphabet); $\delta : Q \times \Sigma \longrightarrow Q$ is the transition function; $q_0 \in Q$ is the start state; and $F \subseteq Q$ is the set of accept states. A DFA *accepts* a string $w = w_1 w_2 \cdots w_n \in \Sigma^*$ if there is a sequence $r_0, r_1, \ldots, r_n$ of states such that (i) $r_0 = q_0$, (ii) $r_n \in F$, and (iii) $\delta(r_i, w_{i+1}) = r_{i+1}$, for every $0 \leq i < n$.

2 Requirements

- We make the following assumptions for simplicity.
  a) The alphabet Σ is always the binary alphabet $\{\texttt{0}, \texttt{1}\}$.
  b) The set of states Q is always of the form $\{0, \ldots, n\}$, for some $n \in \mathbb{N}$.
  c) The start state is always state 0.
- You should implement two functions: `dfa` and `run`.
- `dfa` (which could be a class constructor) takes one parameter which is a string description of a DFA and returns (or constructs) a DFA instance as per the description.
- A string describing a DFA is of the form $P\#S$, where P is a prefix representing the transition function δ and S is a suffix representing the set F of accept state.
- P is a semicolon-separated sequence of triples of states; each triple is a comma-separated sequence of states. A triple i, j, k means that $\delta(i, \texttt{0}) = j$ and $\delta(i, \texttt{1}) = k$.
- S is a comma-separated sequence of states.
- For example, the DFA for which the state diagram appears below may have the following string representation.

```
0,0,1;1,2,1;2,0,3;3,3,3#1,3
```



- `run` simulates the operation of the constructed DFA on a given binary string. It returns `true` if the string is accepted by the DFA and `false` otherwise.
- Important Details:
  - Your implementation should be done within the template file "`DFA.java`" (uploaded to the CMS).
  - You are not allowed to change package, file, contractor, or method names/signatures.
  - You are allowed to implement as many helper classes/methods within the same file (if needed).
  - Public test cases have been provided on the CMS for you to test your implementation.
  - Please ensure that the public test cases run correctly without modification before coming to the lab to maintain a smooth evaluation process.
  - Private test cases will be uploaded before your session and will have the same structure as the public test cases.

3 Evaluation

- Your implementation will be tested by constructing two DFA and running each on five strings.
- You get one point for each correct output of `run`; hence, a maximum of ten points.
- The evaluation will take place during your lab session of the week starting Saturday March 5.

4 Online Submission

- You should submit your code at the following link.

  `https://forms.gle/LESqNzQvJU6e9jKw8`

- Submit one Java file (`DFA.java`) containing executable code.
- Online submission is due on Thursday, March 10, by 23:59.