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

CSEN1002 Compilers Lab, Spring Term 2022
Task 4: Fallback DFA

Due: Week starting 26.03.2022

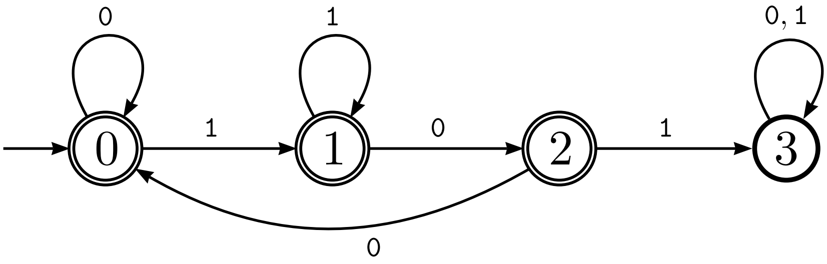
1 Objective

For this task you need to implement a fallback deterministic finite automaton with actions (FDFA) abstract data type. Recall that an FDFA is a sextuple $(Q, \Sigma, \delta, q_0, F, \mathcal{A})$: 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; $F \subseteq Q$ is the set of accept states; and $\mathcal{A}$ is function that maps every state in Q to an action. Refer to the slides of Lecture 2 of CSEN1003 for more details about the operation of FDFA.

2 Requirements

- We make the following assumptions for simplicity.
  a) The alphabet Σ is always the binary alphabet $\{\mathtt{0}, \mathtt{1}\}$.
  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.
  d) $\mathcal{A}$ maps each state to a string s; the action is to print "*lex*,*s*;", where *lex* is as indicated in the lecture.
- You should implement a class constructor `FDFA` and a method `run`.
- `FDFA`, a class constructor, takes one parameter which is a string description of an FDFA and constructs an FDFA instance as per the description.
- A string describing an FDFA is of the form $P\#S$, where P is a prefix representing both the transition function δ and the action function $\mathcal{A}$ and S is a suffix representing the set F of accept state.
- P is a semicolon-separated sequence of quadruples. Each quadruple is a comma-separated sequence of items; the first three items are states and the fourth is an alphanumeric string. A quadruple i, j, k, s means that $\delta(i, \mathtt{0}) = j$, $\delta(i, \mathtt{1}) = k$, and $\mathcal{A}(i) = s$.
- S is a comma-separated sequence of states.
- For example, consider the FDFA for which the state diagram appears below. Suppose that, for state i, $\mathcal{A}(i)$ is the string representation of i. Thus, such an FDFA may have the following string representation.

```
0,0,1,A;1,2,1,B;2,0,3,C;3,3,3,N#0,1,2
```



- `run` simulates the operation of the constructed FDFA on a given binary string. For example, running the above FDFA on the string `1011100` produces the output `10`, `C`; `11100`, `A`;.
- Important Details:
  - Your implementation should be done within the template file "`FDFA.java`" (uploaded to the CMS).
  - You are not allowed to change package, file, constructor, 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 FDFA and running each on five strings.
- You get one point for each correct output of `run`; hence, a maximum of ten points.

4 Online Submission

- You should submit your code at the following link.

https://forms.gle/zkacR6LoynfV3woz7

- Submit one Java file (`FDFA.java`) containing executable code.
- Online submission is due on Thursday, March 31, by 23:59.