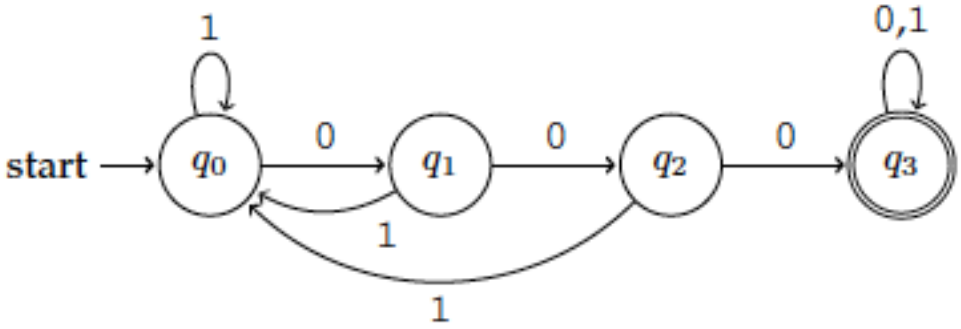
1. Implementazione di un DFA in Java (seconda parte)



Esercizio 1.6

- Progettare e implementare un DFA che riconosca il linguaggio di stringhe che contengono un numero di matricola seguito (subito) da un cognome, dove la combinazione di matricola e cognome corrisponde a studenti del turno T2 o del turno T3 del laboratorio di Linguaggi Formali e Traduttori. Si ricorda le regole per suddivisione di studenti in turni:
  - Turno T1: cognomi la cui iniziale è compresa tra A e K, e la penultima cifra del numero di matricola è dispari;
  - Turno T2: cognomi la cui iniziale è compresa tra A e K, e la penultima cifra del numero di matricola è pari;
  - Turno T3: cognomi la cui iniziale è compresa tra L e Z, e la penultima cifra del numero di matricola è dispari;
  - Turno T4: cognomi la cui iniziale è compresa tra L e Z, e la penultima cifra del numero di matricola è pari.
- Un numero di matricola deve essere composto di almeno due cifre, ma (come in Esercizio 1.3) non ha un numero massimo prestabilito di cifre.
- Ad esempio, il DFA deve accettare le stringhe:
  - "654321Bianchi", "123456Rossi", "221B"

ma non

  - "123456Bianchi", "654321Rossi", "5", "654322", "Rossi", "2Bianchi"

- Assicurarsi che il DFA sia minimo.

Esercizio 1.7

- Progettare e implementare un DFA che riconosca il linguaggio di stringhe che contengono il tuo nome e tutte le stringhe ottenute dopo la sostituzione di un carattere del nome con un altro qualsiasi.
- Ad esempio, nel caso di uno studente che si chiama Paolo, il DFA
  - deve accettare la stringa "Paolo" (cioè il nome scritto correttamente), ma anche le stringhe "Pjolo", "caolo", "Pa%lo", "Paola" e "Parlo"
    (il nome dopo la sostituzione di un carattere)

  - Non deve accettare
    "Eva", "Perro", "Pietro" oppure "P*o*o".

Esercizio 1.8

- Progettare e implementare un DFA che riconosca il linguaggio delle costanti numeriche in virgola mobile utilizzando la notazione scientifica dove il simbolo **e** indica la funzione esponenziale con base 10. L'alfabeto del DFA contiene i seguenti elementi:
  - le cifre numeriche **0, 1, . . . , 9**
  - il segno **.** (punto) che precede una eventuale parte decimale, i segni **+** (più) e **-** (meno) per indicare positività o negatività,
  - il simbolo **e** .
- Le stringhe accettate devono seguire le solite regole per la scrittura delle costanti numeriche.
- In particolare, una costante numerica consiste di due segmenti (il secondo è opzionale):
  - il primo segmento è una sequenza di cifre numeriche che (1) può essere preceduta da un segno + o meno - , (2) può essere seguita da un segno punto . , che a sua volta deve essere seguito da una sequenza non vuota di cifre numeriche;
  - il secondo segmento inizia con il simbolo e, che a sua volta è seguito da una sequenza di cifre numeriche che soddisfa i punti (1) e (2) scritti per il primo segmento. Si nota che, sia nel primo segmento, sia in un eventuale secondo segmento, un segno punto . non deve essere preceduto per forza da una cifra numerica.
- Ad esempio il DFA
  - deve accettare le stringhe "123 ", "123.5 ", ".567 ", "+7.5 ", "-.7 ", "67e10 ", "1e-2 ", "-.7e2 ", "1e2.3 "

  - Non deve accettare
    ".", "e3", "123.", "+e6", "1.2.3", "4e5e6", "++3"

Esercizio 1.9

- Progettare e implementare un DFA definito sull'alfabeto {/ ,*,a} che riconosca il linguaggio di "commenti" delimitati da /* (all'inizio) e */ (alla fine)
- L'automa deve accettare stringhe sull'alfabeto che
  - contengono almeno 4 caratteri
  - che iniziano con /*, che finiscono con */
  - e che contengono una sola occorrenza della stringa */, quella finale (dove l'asterisco della stringa */ non deve essere in comune con quello della stringa /* all'inizio,)
- Esempio: l'automa
  - deve accettare le stringhe:
    "/****/", "/*a*a*/", "/*a/**/", "/**a///a/a**/", "/**/" e "/*/*/"

  - non deve accettare le stringhe:
    "/*/", oppure "/**/***/".

Esercizio 1.10

- Modificare l'automa dell'esercizio precedente in modo che riconosca il linguaggio di stringhe (sull'alfabeto {/, *, a}) che contengono "commenti" delimitati da /* e */ , ma con la possibilità di avere stringhe prima e dopo come specificato qui di seguito.
- L'idea è che sia possibile avere eventualmente commenti (anche multipli) in una sequenza di simboli dell'alfabeto. Quindi l'unico vincolo è che l'automa deve accettare le stringhe in cui un'occorrenza della stringa /* deve essere seguita (anche non immediatamente) da un'occorrenza della stringa */ .
- Le stringhe del linguaggio possono non avere nessuna occorrenza della stringa /* (caso della sequenza di simboli senza commenti).
- Ad esempio, il DFA
  - deve accettare le stringhe:
    "aaa/****/aa", "aa/*a*a*/ ",
    "aaaa", "///"
    "/****/ ", "/*aa*/", "*/a ", "a/**/***a", "a/**/***/a" e "a/**/aa/***/a"

  - ma non deve accettare
    "aaa/*/aa" oppure "aa/*aa".