NFP 108: feuille d'exercices numéro 3

F. Barthélemy et O. Pons

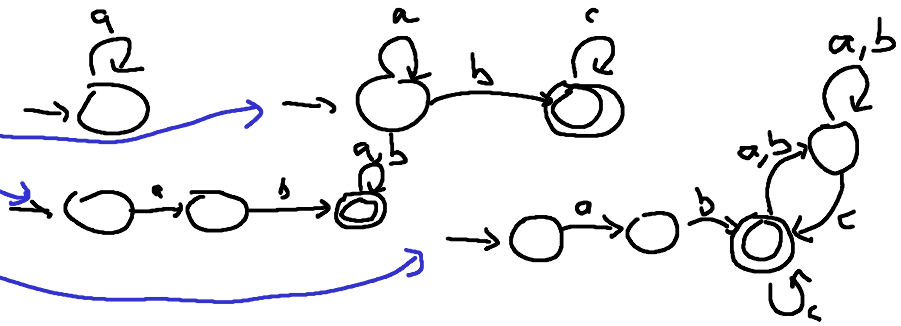
14 octobre 2016

(a*|c*) n'est pas égale à (a|c)*

Exercice 1

Donnez sous forme graphique les automates finis correspondant aux expressions régulières suivantes :

- a^*
- a^*bc^*
- $ab(a|b)^*$
- $ab((a|b)^*c)^*$



Exercice 2

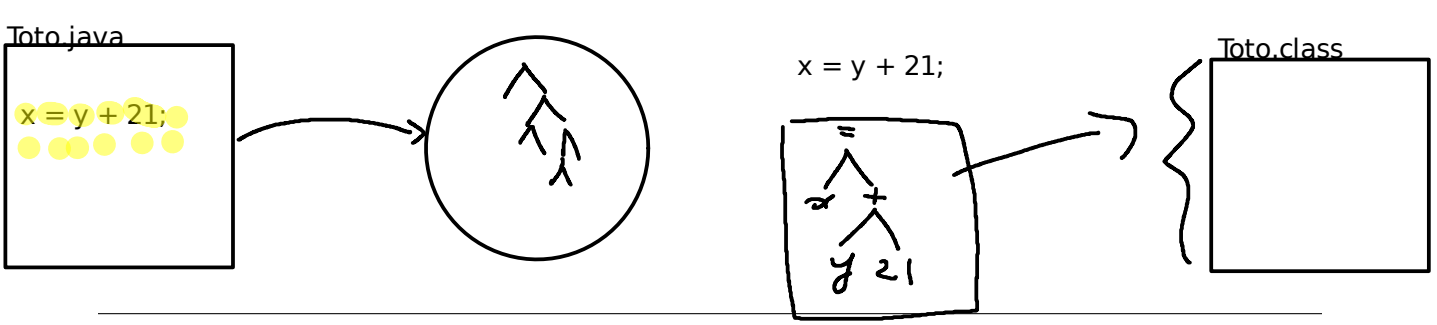
1. donnez une expression régulière très simple permettant de décrire les nombres entiers positifs ou nul, en représentation décimale (base 10). (0|1|2|...|9)(0|1|2|...|9)* (0|1|2|...|9)+
2. donnez une expression régulière représentant ces mêmes nombres en interdisant que le premier chiffre soit 0 sauf dans le cas où il n'y a qu'un seul chiffre, pour représenter le nombre nul. ((1|2|...|9)(0|1|2|...|9)*)|0 e+ = e e*
3. modifiez l'expression précédente pour représenter les nombres entiers positifs, négatifs ou nul. ((-|+|€)(1|2|...|9)(0|1|2|...|9)*) | 0 ici: +0 non accepté (-|+)? e? = (e|€)
4. donnez une expression régulière représentant les nombres décimaux (avec une virgule).
5. donnez une expression des nombres représentant les montants en euros et centimes (nombres avec exactement deux chiffres après la virgule).

4. (((-|+|€)(1|2|...|9)(0|1|2|...|9)*) | 0),(0|1|2|...|9)+

5.(((1|2|...|9)(0|1|2|...|9)*) | 0),(0|1|2|...|9)(0|1|2|...|9)

Exercice 3

Dans cet exercice, on va chercher à décrire les lexèmes (en anglais, token) d'un langage de programmation au moyen d'expressions régulières. Le langage en question sera un petit sous-ensemble de java, illustré par le programme suivant.




```
public class Test{
    public static void main(String[] args){
        int max, nieme;
        boolean exaequo = false;
        max = Integer.parseInt(args[0]);
        for (int i = 1; i<args.length; i++){
            nieme = Integer.parseInt(args[i]);
            if (nieme > max){
                max = nieme;
                exaequo = false;
            } else if (nieme == max){
                exaequo = true;
            }
        }
        System.out.print("Le nombre le plus grand est ");
        System.out.print(max);
        if (exaequo){
            System.out.print(" (plusieurs occurrences)");
        }
        System.out.println();
    }
}
```

Notion d'échappement de caractère
print(" atten\\tion \"danger\" ")

On va se limiter aux constructions et instructions apparaissant dans cet exemple.

Question 1

(public|class|static|void|int|boolean|for|false|true|if|else|return)

- dressez la liste des mots-clés apparaissant dans ce programme (en gras).
- dressez la liste des signes de ponctuations (c'est à dire des signes utilisant un ou plusieurs symboles qui ne sont ni des chiffres ni des lettres) utilisés dans ce programme. (|)|[|]|,|;|=|++|<|>|==|{|}
- donnez les expressions régulières correspondant à ces deux listes.

Question 2

e* = €|e|ee|eee|eeee|...
(ab)* = € | ab |abab| ababab |abababab | ...

- Un identificateur (nom de classe, de variable, de méthode) commence par une lettre et ne comporte que des lettres, des chiffres ou le symbole _. Donnez une expression régulière décrivant les identificateurs. (a-z|A-Z)(0-9|a-z|A-Z|_)* accepte a a1 a_1b abb_12_87 mais pas _a ni 1a
- Donnez une expression régulière pour les entiers positifs, négatifs ou nul. Voir exercice précédent
- Donnez une expression régulière décrivant les chaîne de caractère java (caractères entre doubles quotes). "[^"]*" " ([^\"] | \"| \\ )* "

Question 3

- Donnez une expression régulière donnant l'ensemble des léxèmes (ensemble des constitutants de base) du langage de programmation. Vous pourrez utiliser les expressions définies auparavant en les désignant par un nom.
- Donnez une expression régulière décrivant une déclaration de variable (on se limite aux types présents dans notre exemple).