Fiche TP 02

Partie 01 : Les procédures récursives

1.1 Soit G une grammaire définie par : **G=({S } , {a, b} ,P ,S)** avec S : axiome
P : S →Sab / b / ε (avec : ε : le mot vide)
- Programmez les procédures récursives correspondantes à la grammaire G

1.2 Soit G une grammaire définie par : **G=({E, B, A} , {id ,+ , := } ,P ,E)** avec E : axiome

P : E → A := B
B → A+B / id
A → id

- Programmez les procédures récursives correspondantes à la grammaire G
- Compléter ces procédures par la production en sortie de l'analyse gauche ?

1.3 Soit G une grammaire définie par : **G=({S, A, B} , {a ,b} ,P ,S)** avec S : axiome

P S → A / B
A → aAb / ε (avec : ε : le mot vide)
B → bB / b

- Programmez les procédures récursives correspondantes à la grammaire G

Partie 2 : Les Méthodes descendantes

2.1 Soit G une grammaire définie par : **G=({S } , {a } ,P ,S)** avec S : axiome
P : S →SaS / aa
Programmez G en LL(2)

2.2 On considère les grammaires suivantes :

G_1 =({S_1, A, B} , {a, b, 0, 1 } ,P_1 ,S_1)

P_1 : **S_1** → aA / bB
A → 0A1 / 01
B → 0B11 / 011

G_2 =({S_2, A} , {a, b } ,P_2 ,S_2)

P_2 : **S_2** → abA / ε
A → **S_2**aa / b (avec : ε : le mot vide)

G_3 =({S_3, A, B} , {a, b} ,P_3 ,S_3)

P_3 : **S_3** → aAaB / bAbB
A → a / ab
B → Ba / a

- Faites un programme qui vérifier si G est en LL**(3)**?

2.3 Soit G une grammaire définie par : **G=({S, A, B, C} , {x , y, a, b, e } ,P ,S)** avec S : axiome

P : S → xAB / yAbC
B → eB / e
A → ab / a
C → e / a

- Montrer que G est **LL(3)** mais non pas **LL(3)** forte

Partie 03 : Automates a état fini

Sur l'alphabet A= *{a, b,c},* Programmez un automate fini déterministe reconnaissant les langages suivants :

- Tous les mots contenant un nombre pair de "a" et un nombre pair de "b" ;
- Tous les mots contenant un nombre pair de "a" ou un nombre pair de "c" et que le "b" est au milieu;
- Tous les mots qui n'ont pas plus (>=) de quatre "a" consécutifs ;
- Le langage *L == {$a^n b^p$"*, n = P mod 3, *n, p>=O}.*

Partie 04 :

Voici un programme qui vérifié les expressions régulières en langage Java

```
try{
   Pattern p = Pattern .compile("a*b|c");
   String entree = "aabbbcab";
   Matcher m = p.matcher(entree);
   while (m.find())
      System.out.println(entree.substring(m.start(), m.end()));
}catch(PatternSyntaxException pse){
}
```

A l'exécution de ce programme, qu'est-ce nous donne ? Essayer le rendre un peu dynamique.

"Money you can buy a clock but not time, with money you can buy a book but not knowledge"