Exercice 1 : LR(0), SLR(1) (pour commencer)

On considère la grammaire G_1 suivante d'axiome L et de terminaux {`a`, `b`, `;`} : $L \rightarrow b \mid a;L$

Q 1.1 : Quel est le langage engendré par G_1 ? □

Q 1.2 : Donner l'automate LR-AFD de G_1. □

Q 1.3 : G est-elle LR(0) ? SLR(1) ? □

Q 1.4 : Donner la suite des piles résultant de la reconnaissance par l'analyse LR(0) des mots b et $a;b$. □

Q 1.5 : Donner la table des actions LR(0) de G_1. □

Soit maintenant la grammaire G_2 suivante, de terminaux {`a`, `;`} et d'axiome L : $L \rightarrow \mathtt{a} \mid \mathtt{a};L$

Q 1.6 : Donner l'automate LR-AFD de G_2. □

Q 1.7 : G est-elle LR(0) ? SLR(1) ? □

Q 1.8 : Donner la table des actions SLR(1) de G_2. □

Q 1.9 : Donner la suite des piles résultantes de la reconnaissance par l'analyse SLR(1) du mot $a;a$. □

Soit enfin la grammaire G_3 suivante, de terminaux {`a`, `;`} et d'axiome L : $L \rightarrow \mathtt{a}S \qquad S \rightarrow \epsilon \mid ;L$

Q 1.10 : Donner l'automate LR-AFD de G_3. □

Q 1.11 : G est-elle LR(0) ? SLR(1) ? □

Q 1.12 : Illustrer le fonctionnement de l'analyseur SLR(1) sur le mot `a;a`. □

Exercice 2 : LR(0), SLR(1), et expressions

On considère la grammaire G_1 suivante d'axiome E :

$$
\begin{array}{lcll}
E & \rightarrow & E + T & \mid \; T \\
T & \rightarrow & T * F & \mid \; F \\
F & \rightarrow & *F & \mid \; \mathtt{i}
\end{array}
$$

Q 2.1 : Construire l'automate LR-AFD correspondant. □

Q 2.2 : Cette grammaire est-elle $LR(0)$, $SLR(1)$? □

Q 2.3 : Cette grammaire est-elle ambiguë ? □

Q 2.4 : Illustrer le fonctionnement de l'analyseur SLR(1) sur le mot `i+*i`. □

On remplace la règle $F \rightarrow *F$ de la grammaire G_1 par la règle $F \rightarrow F*$, on obtient la grammaire G_2.

Q 2.5 : Montrer que la grammaire G_2 n'est pas $SLR(1)$ (sans recalculer tout l'automate LR-AFD). □

Exercice 3 : LR(0), SLR(1) (encore et toujours)

Soit la grammaire G des expressions, variables et déréférencement de pointeurs à la C, de terminaux { `x`, `*`, `=` } et de productions : $P = \{S' \to S, S \to V = E \mid E, E \to V, V \to \mathtt{x} \mid *E\}$

Q 3.1 : Construire l'automate LR-AFD et la table des actions. La grammaire est-elle LR(0) ? Si ce n'est pas le cas identifier les conflits. □

Q 3.2 : La grammaire est-elle SLR(1) ? □

Q 3.3 : Montrer sur l'analyse SLR(1) du mot `x=*x` à quel moment l'analyseur ne sait pas choisir entre 2 possibilités, et laquelle il devrait choisir pour accepter ce mot. Construire dans le cas d'acceptation la suite des piles d'exécution, l'arbre syntaxique (vérifier qu'il est construit en ordre postfixe) et la dérivation (vérifier que c'est une dérivation droite). □

Exercice 4 : Récursivité droite / gauche et analyse ascendante

On considère le langage b^+, engendré par l'axiome A, soit par une récursivité gauche (grammaire G_g), soit par une récursivité droite (grammaire G_d) :

- G_g : $A \to Ab \mid b$;
- G_d : $A \to bA \mid b$.

Q 4.1 : Calculer dans les deux cas l'automate LR-AFD et constater que G_g et G_d sont SLR(1). □

Q 4.2 : Exécuter dans les deux cas l'analyse SLR(1) sur le mot *bbb*. En déduire s'il est plus rentable d'utiliser une récursivité gauche ou une récursivité droite pour l'analyse ascendante. □

Exercice 5 : Grammaire LR(1)

Q 5.1 : Montrer que la grammaire de l'exercice 3 est LR(1) en construisant l'état initial de l'automate LR(1) et le chemin qui mène à l'état qui rend la grammaire non SLR(1). □

Exercice 6 : Grammaires non LR(1)

On suppose donné un langage d'impression qui permet d'afficher entre autres des entiers et des booléens. On donne une grammaire incomplète (en réalité il y a d'autres affichages et les expressions ne sont pas réduites à des identificateurs) :

affichage → `write ( ` *exprEnt* ` , %i )`   *exprEnt* → `id`
affichage → `write ( ` *exprBool* ` , %b )`   *exprBool* → `id`

Q 6.1 : Expliquer pourquoi la grammaire G_2 n'est pas LR(1). Est-elle LR(k) ? □

On reprend l'exemple de l'exercice 2 avec la production $F \to F* \mid \mathtt{i}$.

Q 6.2 : Expliquer pourquoi cette grammaire n'est pas LR(1). Est-elle LR(k) ? □