

Université Paris 7 — IUP informatique 1^{ère} année
 Examen de compilation — 8 septembre 2000
 Durée : 2h

La notation tiendra compte de la clarté des explications et des justifications fournies.

Nom :	Prénom :
--------------	-----------------

Exercice 1 Écrire un source lex qui supprime les commentaires d'un programme C++. Les lignes contenant seulement des blancs et un commentaire seront supprimées (y compris le caractère de fin de ligne). Pour les autres lignes contenant un commentaire, on supprimera seulement le commentaire. Enfin les lignes ne contenant pas de commentaire seront laissées telles quelles.

Le programme lira sur l'entrée standard et écrira sur la sortie standard. On rappelle que les commentaires C++ commencent par // et se terminent en fin de ligne. Le programme devra de plus traiter correctement les chaînes de caractères (qui peuvent contenir la séquence //).

Exercice 2 Répondre par vrai ou faux aux questions suivantes. Barème : 0,5 pour une réponse correcte, -0,5 pour une réponse incorrecte et 0 en cas d'absence de réponse. Il n'est pas demandé de justification.

- | | Vrai | Faux |
|---|--------------------------|--------------------------|
| 1. Toute grammaire LR(0) est LL(1) | <input type="checkbox"/> | <input type="checkbox"/> |
| 2. Si une grammaire est ambiguë, elle ne peut pas être SLR(1). | <input type="checkbox"/> | <input type="checkbox"/> |
| 3. Le premier état de l'automate LR(0) contient toutes les règles de la grammaire, le marqueur étant au début de chaque règle. | <input type="checkbox"/> | <input type="checkbox"/> |
| 4. On ne peut pas travailler avec une grammaire ambiguë lorsqu'on utilise le logiciel yacc. ... | <input type="checkbox"/> | <input type="checkbox"/> |

Problème On considère la grammaire G suivante :

- | | |
|-----------------------------------|-----|
| $P \rightarrow B\$$ | (1) |
| $B \rightarrow \{ LI \}$ | (2) |
| $LI \rightarrow LI ; I$ | (3) |
| $LI \rightarrow I$ | (4) |
| $I \rightarrow B$ | (5) |
| $I \rightarrow \mathbf{id} = E$ | (6) |
| $E \rightarrow \mathbf{id} (E)$ | (7) |
| $E \rightarrow \mathbf{nb}$ | (8) |
| $E \rightarrow \mathbf{id}$ | (9) |

Dans cette grammaire, $\$, \}, \{, ;, =,), (, \mathbf{nb}$ et \mathbf{id} sont des terminaux, et où P, B, LI, I, E sont des non-terminaux.

1. Décrire de façon informelle le langage engendré par G .
2. La grammaire G est-elle ambiguë? Justifier précisément la réponse. Un exemple ne suffit pas.
3. Donner un arbre de dérivation de la chaîne $\{ \mathbf{id} = \mathbf{id} (\mathbf{nb}) \} \$$.
4. Montrer que G n'est pas LL(1).
5. Calculer pour chaque symbole non-terminal de G les ensembles Premier et Suivant correspondants.
6. Trouver une grammaire G_1 qui engendre le même langage que G mais qui est LL(1). On demande une justification précise du fait que G_1 est LL(1).
7. Préciser pour chaque non-terminal de G_1 quelle règle est appliquée par l'analyseur LL(1) en fonction du symbole d'avance.
8. Calculer l'automate des items LR(0) de G .
9. La grammaire G est-elle LR(0)?

10. La grammaire G est-elle SLR(1)?
11. Simuler le comportement d'un analyseur syntaxique montant (en précisant à chaque étape l'entrée, la pile des symboles, l'action choisie) sur l'entrée **{ id = id (nb) } \$**.
12. Écrire un source yacc permettant de
 - compter le nombre total d'instructions dans le programme (les instructions étant séparées par le symbole ;).
 - compter et afficher la profondeur de chaque bloc entre accolades (le bloc le plus externe est de profondeur 0, et les blocs à l'intérieur d'un bloc de profondeur i sont de profondeur $i + 1$);

Les attributs utilisés sont-ils synthétisés? hérités?