Licence d'informatique - Traduction

Examen du 15 janvier 2007

Tous documents sont autorisés. Durée de l'épreuve : 2 h

1 Analyse syntaxique

Soit la grammaire algébrique suivante :

(0)	S	$\longrightarrow$	$E\ \$$
(1)	E	$\longrightarrow$	$ST\ C\ ET$
(2)	E	$\longrightarrow$	MT
(3)	ST	$\longrightarrow$	$<\ \mathtt{id}\ >$
(4)	ET	$\longrightarrow$	$<\ /\ \mathtt{id}\ >$
(5)	MT	$\longrightarrow$	$<\ \mathtt{id}\ /\ >$
(6)	C	$\longrightarrow$	$C\ E$
(7)	C	$\longrightarrow$	$C\ \mathtt{text}$
(8)	C	$\longrightarrow$	E
(9)	C	$\longrightarrow$	$\mathtt{text}$

Les non-terminaux sont S, E, ST, C, MT et ET et les terminaux sont $<$, `id`, $>$, $/$, `text`, et $\$$. Le terminal `id` désigne des identificateurs qui ne peuvent contenir que des lettres et des chiffres. Le terminal `text` désigne du texte qui peut contenir n'importe quels caractères sauf $<$ et $\$$.

1. Dessiner l'arbre de dérivation de :

$$< html >< head >< title > \texttt{ Ma page } < /title >< /head >< body > \texttt{ Ma page } < hr/ >< /body >< /html > \$$$

2. Construire l'automate d'analyse LR(0). Numéroter les états de l'automate en commençant par 0, OBLIGATOIREMENT dans l'ordre correspondant à un parcours de l'automate en largeur (quand on a fini d'étudier les transitions sortant d'un état e de l'automate, on passe au premier état qui a été créé après e), et en respectant l'ordre suivant parmi les symboles : $<$ `id` $>$ $/$ `text` $\$$ S E ST C MT ET

3. La grammaire est-elle LR(0)? Construire une table d'analyse ascendante. Chaque case de la table sera de la forme ei (empiler l'état i et avancer), rj (réduire selon la règle j), `Acc` (accepter) ou sera laissée vide (erreur de syntaxe).

4. Utiliser la table d'analyse précédente pour analyser l'expression

$$< body > \texttt{ Ma page } < hr/ >< /body >< /html > \$$$

5. Déterminer SUIVANT(ET).

2 Analyse lexicale

Réaliser une spécification Flex pour un analyseur lexical couplé avec l'analyseur syntaxique de la question précédente. Le texte qui apparait à l'extérieur des balises $<$ ... $>$, par exemple `Ma page` dans les exemples précédents, doit être analysé comme `text` et non comme `id`.

3 Génération de code

Réaliser des spécifications Flex et Bison pour une calculatrice booléenne à 40 variables. Les identificateurs des variables (`id` dans la grammaire ci-dessous) sont constitués d'une des lettres a, b, c, d seule ou suivie d'un chiffre de 1 à 9. Les constantes sont 0 (faux) et 1 (vrai). La syntaxe de la calculatrice est conforme à la grammaire suivante (en notation Bison) :

```
P   :   I ';' P
    |   I '.'
    ;
I   :   id ':' '=' E
    ;
E   :   E ou T
    |   T
    ;
T   :   T et F
    |   F
    ;
F   :   id
    |   0
    |   1
    |   non F
    |   ( E )
    ;
```

Pour chaque instruction, la calculatrice doit afficher la valeur de la variable. Par exemple, pour le source suivant :

```
a := 0 ; b := 1 ; c := 1 ; a := non (a ou b) et c .
```

la réponse doit être 0 1 1 0.