UNIVERSITE LIBRE DE BRUXELLES Le 29 mai 2006

Théorie des langages et de la compilation
Année académique 2005-2006
1ère licence en Informatique et
2ème année du grade de l'ingénieur civil informaticien
Examen de première session

Remarques préliminaires

- On vous demande de répondre à chaque question sur des feuilles séparées (les correcteurs pouvant être différents).
- N'oubliez pas d'inscrire vos nom et prénom sur chacune des feuilles.
- Vous ne pouvez pas utiliser de notes.
- L'examen comporte cinq questions pour un total de 16 points, les projets (4 points) complétant la note.
- Vous disposez de quatre heures. Pensez à gérer correctement votre temps.
- Si rien n'est spécifié pour les formalismes, les conventions utilisées habituellement dans le cours sont prises.

Question 1 (6pts)

Soit $\Sigma = \{[,]\}$ l'alphabet comprenant uniquement les 2 symboles '[' et ']'. On donne comme définition : un mot est "bien parenthésé" si à toute parenthèse ouvrante est associée une (unique) parenthèse fermante qui lui est postérieure. Ainsi :
- '[[][[][]]]' est un mot bien parenthésé
- '][' n'est pas bien parenthésé.

Notez que tout mot de L peut être construit par une suite d'insertions, à partir du mot vide ϵ, du sous mot '[]' dans le mot x obtenu à l'étape précédante, insertion après la position p avec $0 \leq p \leq |x|$.

Dans la suite de cette question, L est ce langage des mots bien parenthésés.

1. Donnez une définition mathématique formelle du langage L.
2. Parmi les formalismes suivants :
   (a) les automates finis déterministes
   (b) les automates à pile
   (c) les machines de Turing
   (d) les grammaires
   (e) les expressions régulières

   lesquels permettent de définir ce langage L (précisez si nécessaire, mais sans démonstration) ?
3. Choississez 3 formalismes parmi ceux-ci et donnez une définition formelle complète pour L dans ces formalismes.
4. Enoncez et démontrez formellement le lemme de pompage (*pumping lemma*) des langages réguliers.
5. Peut-on définir L avec une expression régulière ? (Dé)montrez.

Question 2 (3pts)

Soit la grammaire :

$1, 2$	S	$\rightarrow$	$aA \mid bB$
$3, 4, 5$	A	$\rightarrow$	$Ac \mid d \mid \varepsilon$
$6, 7$	B	$\rightarrow$	$De \mid DG$
$8, 9, 10$	G	$\rightarrow$	$fG \mid fHK \mid g$
$11, 12$	H	$\rightarrow$	$HeH \mid I$
13	I	$\rightarrow$	HI
14	K	$\rightarrow$	k
15	D	$\rightarrow$	d

Transformez-la en grammaire LL(1) "nettoyée". Construisez la table des actions de l'analyseur LL(1) associé à cette grammaire. Écrivez un analyseur en decente récursive (dans un langage impératif) pour cette grammaire.

Question 3 (3pts)

Soit la grammaire :

S	$\rightarrow$	$A\$$	
A	$\rightarrow$	$bcC \mid d \mid B$	
B	$\rightarrow$	dAe	
C	$\rightarrow$	aAa	

Construisez l'automate LALR(1) correspondant, en donnant les détails des différentes étapes de la construction. Expliquez comment fonctionne l'analyse du mot $bcada\$$.

Question 4 (2pts)

Expliquez la gestion de la mémoire lors de l'exécution d'un programme impératif (tel que C++) et comment le compilateur peut générer le code correspondant à un accès mémoire à une variable.

Question 5 (2pts)

1. Donnez la définition de langage Récursivement énumérable ($\in RE$) et récursif ($\in R$).
2. Démontrez que le langage universel $L_u = \{\langle M, w\rangle | M \text{ accepte } w\}$ est récursivement énumérable mais pas récursif (où M est la codification d'une machine de Turing (dans l'alphabet $\{0, 1\}$ et w un string dans ce même alphabet $\{0, 1\}$).