UNIVERSITÉ DE SHERBROOKE

DÉPARTEMENT D'INFORMATIQUE

IFT 313

Introduction aux langages formels

EXAMEN PÉRIODIQUE [1]

Le jeudi 8 octobre 2015 de 10 h 30 à 12 h 20 Professeur : Richard St-Denis

— Toute documentation est permise.

— **Tout appareil électronique, incluant le téléphone cellulaire et l'ordinateur, est interdit.**

— Ne dégrafez pas ce questionnaire.

— Répondez dans les espaces prévus à cet effet.

— **Personne ne peut quitter la salle d'examen avant 10 h 45.**

— **Personne ne peut quitter son siège entre 12 h 10 et 12 h 20.**

— La correction est, entre autres, basée sur le fait que chacune de vos réponses soit :

  — claire, c'est-à-dire lisible et compréhensible pour le correcteur ;

  — précise, c'est-à-dire exacte ou sans erreur ;

  — complète, c'est-à-dire que toutes les étapes de résolution du problème sont présentes ;

  — concise, c'est-à-dire que la méthode de résolution est la plus courte possible.

Nom : _____ Prénom : _____

Signature : _____ Matricule : _____

Question	Barème
1	/6
2	/6
3	/6
4	/6
5	/6
total :	/30

1. Ce questionnaire comporte sept (7) pages.

1. (6 points) Le langage de programmation *PL/1* a été développé par la compagnie *IBM*, bien avant votre naissance, dans le début des années 1960.

   Par exemple, dans ce langage, l'énoncé suivant déclare les variables `I`, `J` et `K` comme étant des valeurs numériques à point fixe :

   ```
   DECLARE (I,J,K) FIXED
   ```

   Mais l'énoncé suivant affecte la valeur 0 à la composante `I`, `J`, `K` du tableau de dimension 3 ayant le nom `DECLARE` (dans ce langage les indices d'un tableau ne sont pas entre crochets, mais entre parenthèses) :

   ```
   DECLARE (I,J,K) = 0
   ```

   a) (3 points) Que peut-on dire du lexème `DECLARE` rencontré dans un programme *PL/1* ?

   b) (3 points) Quel problème ce cas pathologique soulève-t-il au niveau de l'analyse lexicale du compilateur *PL/1* ?

2. (6 points) Le langage de programmation *Rust*, lancé en 2010 par la compagnie *Mozilla*, inclut un littéral particulier, appelé *raw string literal* (traduit librement en *chaîne de caractères brute* en français). Voici la règle lexicale en anglais d'un tel littéral tirée du manuel de référence du langage de programmation *Rust* (la version française est également donnée) :

   > Raw string literals do not process any escapes. They start with the character `U+0072` (`r`), followed by zero or more of the character `U+0023` (`#`) and a `U+0022` (double-quote) character. The raw string

> body can contain any sequence of *Unicode* characters and is terminated only by another `U+0022` (double-quote) character, followed by the same number of `U+0023` (`#`) characters that preceded the opening `U+0022` (double-quote) character.
>
> All *Unicode* characters contained in the raw string body represent themselves, the characters `U+0022` (double-quote) (except when followed by at least as many `U+0023` (`#`) characters as were used to start the raw string literal) or `U+005C` (\) do not have any special meaning.

Voici une traduction libre en français :

> Les chaînes de caractères brutes commencent par le caractère `U+0072` (la lettre `r`), suivi de zéro ou plusieurs caractères `U+0023` (le caractère `#`) et du caractère `U+0022` (le guillemet). Le corps d'une chaîne de caractères brute peut contenir toute suite de caractères *Unicode* qui se termine par un autre caractère `U+0022` (le guillemet), suivi par le même nombre de caractères `U+0023` (le caractère `#`) qui précèdent le guillemet ouvrant de la chaîne.
>
> Tous les caractères *Unicode* contenus dans le corps d'une chaîne de caractères brute sont considérés comme tels, en particulier le caractère `U+0022` (le guillemet), sauf celui suivi d'au moins d'autant de caractères `U+0023` (le caractère #) utilisés en ouverture de la chaîne, et le caractère `U+005C` (le caractère \), considéré usuellement comme le caractère d'échappement, n'ont pas de signification particulière.

Par exemple, les chaînes de caractères brutes suivantes

```
r"bit"     r#""bit""#     r##"bit #"# qubit"##  r"\x52"
```

représentent respectivement les chaînes de caractères `bit`, `"bit"`, `bit #"# qubit` et `\x52`.

a) (3 points) Donnez une expression régulière pour les chaînes de caractères brutes, la plus courte possible, qui se rapproche au mieux de la description précédente. Vous pouvez utiliser *c* pour désigner un caractère quelconque de l'ensemble *Unicode*. Expliquez votre solution.

b) (2 points) Quel problème la règle lexicale des chaînes de caractères brutes soulève-t-elle dans l'écriture d'une expression régulière ?

c) (1 point) Comment ce problème peut-il être résolu par un analyseur lexical ?

3. (6 points) Répondez aux deux questions suivantes.

   a) (4 points) Donnez une grammaire hors contexte pour les chaînes de caractères brutes telles que définies au numéro précédent. Vous devez clairement indiquer l'ensemble des variables V, l'ensemble des symboles terminaux T, l'ensemble des règles de production P et l'axiome S.

b) (2 points) À partir de votre réponse en a), donnez une dérivation pour la chaîne `r##""a""##`.

4. (6 points) Voici une expression régulière pour les chaînes de $\{a, b\}^*$ qui ne se terminent pas la sous-chaîne aaa :

$$r = (a + b)^*(b + ba + baa) + \lambda + a + aa$$

Vous devez construire l'automate fini déterministe qui accepte le même langage que celui défini par l'expression régulière r en utilisant la méthode basée sur la construction d'un arbre d'analyse.

(3 points) Arbre d'analyse (incluant les décorations *firstpos*, *nullable* et *lastpos*) :

(2 points) La fonction *followpos* :

(1 point) L'automate fini déterministe sous la forme d'un graphe de transition d'états :

5. (6 points) Voici l'expression régulière comme solution à la dernière question du devoir #1. Elle représente le comportement contrôlé du chat et de la souris dans un labyrinthe qui comporte les portes c_1 à c_7 pour le chat et les portes s_1 à s_6 pour la souris :

$$(s_5 s_6 s_4 + c_3(c_1 + c_4 c_7)(c_7 c_7)^* c_2)^*$$

a) (3 points) Calculez la λ-*fermeture* de l'item suivant en applicant successivement les règles de déplacement instantané du point telles que présentées dans le deuxième algorithme pour le passage d'une expression régulière à un automate fini déterministe. Indiquez toutes les étapes.

$$\bullet\Big((s_5 s_6 s_4 + c_3(c_1 + c_4 c_7)(c_7 c_7)^* c_2)^*\Big)$$

b) (3 points) Calculez les prochains ensembles d'items à partir de l'ensemble d'items suivant et de la lecture de symboles. Vous devez, pour chaque ensemble d'items obtenu, indiquer sur quel symbole la lecture est effectuée. Indiquez toutes les étapes.

$$\{(s_5 s_6 s_4 + c_3(\bullet c_1 + c_4 c_7)(c_7 c_7)^* c_2)^*, (s_5 s_6 s_4 + c_3(c_1 + \bullet c_4 c_7)(c_7 c_7)^* c_2)^*\}$$

FIN DE L'EXAMEN