Nom : ..... Prénom : ..... Matricule : .....

Introduction à la théorie de l'informatique

Examen écrit du vendredi 23 août 2013

Durée : 3 heures 1/2 maximum.

Lisez les consignes avant de commencer l'examen :

- Commencez par inscrire vos nom, prénom et matricule en haut de chaque page. Vous pouvez détacher les feuilles d'examen si vous le souhaitez.
- Exposez votre raisonnement de manière claire et complète si vous voulez avoir le maximum de points. Il n'est pas suffisant de donner simplement la solution finale.
- Pour les questions demandant une réponse finale numérique, entourez ou identifiez clairement la réponse finale.
- Soyez bref et concis, mais précis. Les propositions incorrectes ou hors-sujet vous feront perdre des points.
- Vous avez le droit d'utiliser **uniquement** les transparents du cours théorique.
- Vous ne pouvez communiquer avec personne excepté les examinateurs pendant l'examen. Toute personne surprise à tricher recevra une note de 0/20 pour l'examen et s'expose à des sanctions.
- Cet examen comporte 5 questions. Vérifiez que vous avez bien les 5 questions sur vos feuilles d'énoncé !

Bon travail !

1. L'algorithme suivant implémente le calcul de la racine carrée entière d'un nombre entier N, c'est-à-dire le calcul de $a \in \mathbb{N}$ tel que $a^2 \leq N$ et $(a+1)^2 > N$

```
SQRTINT(N)
1  a = 0 ;b = 1
2  while b^2 ≤ N
3        b = b · 2
4  while b ≠ 1
5        b = ⌊b/2⌋
6        if (a + b)^2 ≤ N
7              a = a + b
8  return a
```

   (a) En considérant les lignes 1 à 3 comme une initialisation, modéliser cet algorithme par une machine d'état. Définissez précisément l'ensemble d'états, l'état initial et les transitions de la machine.

   (b) En utilisant le théorème d'invariant, montrez que l'algorithme est partiellement correct pour les précondition et postcondition suivantes :
   Pre = "$N \in \mathbb{N}$"
   Post = "$a^2 \leq N$ et $(a+1)^2 > N$".
   Suggestion : utilisez comme invariant le prédicat : "$a^2 \leq N$ et $(a+b)^2 > N$ et $\exists i : i \geq 0 : b = 2^i$"

   (c) Montrez que l'algorithme se termine toujours après au plus $\log_4(N)+1$ exécutions du corps de la deuxième boucle.

2. Soit l'ensemble $BinNum$ de séquences binaires défini récursivement comme suit :
   – Cas de base : $\mathtt{0} \in BinNum$, $\mathtt{1} \in BinNum$.
   – Cas inductif : si $b \in BinNum$, alors $b\mathtt{0} \in BinNum$ et $b\mathtt{1} \in BinNum$.

   (a) Définissez une fonction `number` : $BinNum \to \mathbb{N}$ qui renvoie le nombre entier codé par la séquence binaire et une fonction `sum` : $BinNum \to \mathbb{N}$ qui renvoie la somme des chiffres de la séquence binaire. Par exemple,

$$\mathtt{number}(\mathtt{101}) = 5,\ \mathtt{number}(\mathtt{001111}) = 15$$

$$\mathtt{sum}(\mathtt{101}) = 2,\ \mathtt{sum}(\mathtt{001111}) = 4.$$

   (b) Montrez par induction structurelle que pour tout $b \in BinNum$, on a $\mathtt{sum}(b) \leq \mathtt{number}(b)$.

3. Soit le langage de programmation simple vu au cours dont la syntaxe et la sémantique opérationnelle sont rappelées aux figures 1 et 2 :
   (a) Complétez la sémantique opérationnelle pour pouvoir évaluer le programme suivant et développer les 11 premières étapes de son évaluation :
   $$b := 1;\ \texttt{while}\ b * b \leq N\ \texttt{do}\ b := b * 2;$$
   (b) On désire ajouter au langage la possibilité d'effectuer une boucle `for` du type :
   $$\texttt{for}\ x := E_1\ \texttt{to}\ E_2\ \texttt{do}\ C$$
   où $E_1, E_2 \in Exp$ et $C \in Com$.
      i. Décrivez informellement une sémantique d'évaluation possible de cette instruction.
      ii. Ajoutez les règles correspondant à cette sémantique à la sémantique opérationnelle du langage. Vous ne pouvez pas supposer l'existence de l'instruction `while` dans le langage.

4. (a) Montrez qu'un graphe connexe avec n sommets et $n-1$ arêtes est un arbre.
   (b) Montrez *par induction sur le nombre d'arêtes* que tout graphe connexe $G = (V, E)$ contient un arbre couvrant.

5. On cherche à déterminer a_n le nombre de séquences binaires de longueur n codant pour un entier multiple de 3. On supposera que la longueur d'une séquence binaire ne prend pas en compte les `0` les plus à gauche. Par exemple, `00101` et `101` représentent la même séquence de longueur 3 et `0` est une séquence de longueur 0. On a ainsi par exemple $a_0 = 1$ et $a_1 = 0$ (la seule séquence de longueur 1 est `1` qui n'est pas un multiple de 3).
   (a) Montrez que $a_2 = 1$ et $a_3 = 1$.
   (b) En vous basant sur le fait que les séquences multiples de 3 peuvent être représentées par l'expression suivante :
   $$(1(01^*0)^*10^*)^*,$$
   montrez que la fonction génératrice $A(z)$ correspondant à a_n est :
   $$A(z) = \sum_{n=0}^{+\infty} a_n z^n = 1 + \frac{z^2}{(1-2z)(1+z)}$$
   (c) En vous servant de $A(z)$, montrez que $a_n = a_{n-1} + 2a_{n-2}$ pour $n \geq 3$.
   (d) Trouvez une formulation analytique pour a_n (soit en calculant $[z^n]A(z)$, soit en résolvant la récurrence).

$$
\begin{array}{rcl}
B \in \mathit{Bool} & ::= & \texttt{true} \mid \texttt{false} \mid E = E \mid E < E \mid \ldots \\
 & & \mid B \& B \mid \neg B \mid \ldots \\
E \in \mathit{Exp} & ::= & x \mid \mathtt{n} \mid (E + E) \mid \ldots \\
C \in \mathit{Com} & ::= & x := E \mid \texttt{if } B \texttt{ then } C \texttt{ else } C \\
 & & \mid C; C \mid \texttt{skip} \mid \texttt{while } B \texttt{ do } C
\end{array}
$$

FIGURE 1 – Définition de la syntaxe du langage (Transparent 174)

$$\frac{}{\langle x, s\rangle \to \langle \mathtt{n}, s\rangle}\ \texttt{(W-EXP.NUM)} \qquad \frac{\langle E, s\rangle \to \langle E', s'\rangle}{\langle x := E, s\rangle \to \langle x := E', s'\rangle}\ \texttt{(W-ASS.EXP)}$$

$$\frac{}{\langle (\mathtt{n}_1 + \mathtt{n}_2), s\rangle \to \langle \mathtt{n}_3, s\rangle}\ \texttt{(W-EXP.ADD)} \qquad \frac{}{\langle x := \mathtt{n}, s\rangle \to \langle \texttt{skip}, s[x \mapsto \mathtt{n}]\rangle}\ \texttt{(W-ASS.NUM)}$$

$$\frac{\langle E_1, s\rangle \to \langle E_1', s'\rangle}{\langle (E_1 + E_2), s\rangle \to \langle (E_1' + E_2, s'\rangle}\ \texttt{(W-EXP.LEFT)} \qquad \frac{\langle C_1, s\rangle \to \langle C_1', s'\rangle}{\langle C_1; C_2, s\rangle \to \langle C_1'; C_2, s'\rangle}\ \texttt{(W-SEQ.LEFT)}$$

$$\frac{}{\langle \texttt{skip}; C_2, s\rangle \to \langle C_2, s\rangle}\ \texttt{(W-SEQ.SKIP)}$$

$$\frac{\langle B, s\rangle \to \langle B', s'\rangle}{\langle \texttt{if } B \texttt{ then } C_1 \texttt{ else } C_2, s\rangle \to \langle \texttt{if } B' \texttt{ then } C_1 \texttt{ else } C_2, s'\rangle}\ \texttt{(W-COND.B)}$$

$$\frac{}{\langle \texttt{if true then } C_1 \texttt{ else } C_2, s\rangle \to \langle C_1, s\rangle}\ \texttt{(W-COND.TRUE)}$$

$$\frac{}{\langle \texttt{if false then } C_1 \texttt{ else } C_2, s\rangle \to \langle C_2, s\rangle}\ \texttt{(W-COND.FALSE)}$$

$$\frac{}{\langle \texttt{while } B \texttt{ do } C, s\rangle \to \langle \texttt{if } B \texttt{ then}(C; \texttt{while } B \texttt{ do } C) \texttt{ else skip}, s\rangle}\ \texttt{(W-WHILE)}$$

FIGURE 2 – Sémantique opérationnelle du langage (Transparent 183)