Examen final d'ILP
2ème session

Christian Queinnec

23 janvier 2009

Conditions générales

Cet examen est formé d'un unique problème en plusieurs questions auxquelles vous pouvez répondre dans l'ordre qui vous plait.

Le barème est fixé à 20 ; la durée de l'épreuve est de 3 heures. Tous les documents sont autorisés et notamment ceux du cours.

Votre copie sera formée de fichiers textuels que vous laisserez aux endroits spécifiés dans votre espace de travail pour Eclipse. L'espace de travail pour Eclipse sera obligatoirement nommé `workspace` et devra être un sous-répertoire direct de votre répertoire personnel.

À l'exception des clés USB en lecture seule, tous les appareils électroniques sont prohibés (y compris les téléphones portables, les assistants numériques personnels et les agendas électroniques).

L'examen sera corrigé à la main, il est donc absolument inutile de s'acharner sur un problème de compilation ou sur des méthodes à contenu informatif faible. Il est beaucoup plus important de rendre aisé, voire plaisant, le travail du correcteur et de lui indiquer, par tout moyen à votre convenance, de manière claire, compréhensible et terminologiquement précise, comment vous surmontez cette épreuve. À ce sujet, vos fichiers n'auront que des lignes de moins de 80 caractères, n'utiliseront que le codage ASCII ou UTF-8 enfin, s'abstiendront de tout caractère de tabulation.

Le langage à étendre est ILP4. Le paquetage Java correspondant à cet examen sera donc nommé `fr.upmc.ilp.ilp4dynarray`. Sera ramassé, à partir de votre *workspace* (situé sous ce nom directement dans votre HOME), tout répertoire ou fichier ayant le fragment `4dynarr` dans son nom.

1 Introduction de tableaux

On souhaite étendre ILP4 avec des tableaux unidimensionnels (des vecteurs) dynamiquement alloués. Le programme (inepte) suivant illustre toutes les opérations possibles sur les tableaux :

```
function foo (i, t) {
  2.78 * t[2*i]                 // lecture
}
let k = 0
  let tab = array(4*(k+1))      // creation
    tab[3] = "hello World!"     // écriture
    while ( k < tab.length ) {  // taille
      let v = foo(k-1, tab)     // transmission
      tab[k+1] = v              // écriture
      k = k+1
    }
  }
```

Les positions dans le tableau peuvent être lues ou écrites, elles sont indexées à partir de zéro (comme en C). La taille du tableau peut être obtenue (ici avec une notation à la Java). Un tableau peut être passé en argument. Cette extension doit être sûre : toute lecture ou écriture en dehors du tableau signalera une exception.

Question 1 – Grammaire (4 points)

Étendre la grammaire d'ILP4 pour inclure les nouvelles expressions autour des tableaux. Vous pouvez abuser de commentaires pour y insérer les points les plus pertinents de vos réflexions.

Livraison

– un fichier `grammar4dynarray.rnc` placé dans le répertoire `Grammars`.

Question 2 – Programme (3 points)

Écrire un programme en syntaxe XML conforme à la grammaire précédente et utilisant toutes ces nouvelles extensions. Vous placerez un commentaire en tête pour indiquer ce que ce programme effectue.

Livraison

– un fichier `a-4dynarray.xml` placé dans le répertoire `Grammars/Samples`.

Question 3 – Interprétation (5 points)

Écrire les méthodes d'évaluation des expressions ajoutées. Vous préciserez en commentaire dans ces classes comment vous représentez les tableaux, où vous les créez et où vous les stockez.

Livraison

– les fichiers correspondant à ces classes placés dans le paquetage `fr.upmc.ilp.ilp4dynarray`.

Question 4 – Compilation (8 points)

Écrire les schémas de compilation (tels qu'utilisés dans le cours, pour ILP4) correspondant aux expressions ajoutées et au nouveau patron de génération du fichier C. Vous les préciserez dans les commentaires des classes concernées. Vous détaillerez également les enrichissements concernant la bibliothèque d'exécution.

Livraison

– les fichiers correspondant à ces classes placés dans le paquetage `fr.upmc.ilp.ilp4dynarray`.
– les éventuels fichiers C (avec un nom contenant `4dynarr`) correspondant.