Université BORDEAUX I **Année 1995-96**
Session de janvier

Maîtrise d'informatique

Modèles et Techniques de Programmation (J.P.Braquelaire)

Durée: 2 heures
Tous documents autorisés

Première partie

On considère l'extrait suivant de session Lisp :

```
> (setq alpha 100)
alpha
>
> (defun f (x y)
    (/ (+ (* alpha x)
          (* (- 100 alpha) y))
       100))
f
> (+ (f 1 2)
   (let ((alpha 20))
     (f  15 5)))
8
>
```

1) – 1 point – S'agit-il d'une évaluation avec liaison statique ou dynamique (justifier sommairement) ?

2) – 1 point – Quel resultat obtient-on avec l'autre mode d'évaluation (justifier sommairement) ?

3) – 1 point – Expliquer au moyen d'une exemple de quelques lignes en quoi le mécanisme de liaison dynamique permet de reproduire celui du passage de paramètres lors d'un appel de fonction.

4) – 2 points – Que permet-il de faire que le passage de paramètres ne permet pas ?

5) – 1 point – Quel avantage ce mécanisme peut-il apporter du point de vue de la réutilisabilité ?

Deuxième partie

1) – 1 point – Quelles sont pour un programme C les deux façons d'allouer dynamiquement de la mémoire ?

2) – 1 point – Expliquer dans quels cas il est préférable d'utiliser l'un ou l'autre de ces deux mécanismes.

3) – 6 points – Écrire en C ANSI le module `blocmem` d'allocation dynamique de blocs mémoire dont les fonctionnalités sont les suivantes :

- allocation dynamique de mémoire au moyen de la fonction par défaut `malloc` de la bibliothèque standard;
- installation d'une nouvelle fonction d'allocation et de la fonction de libération associée;
- restitution d'un bloc mémoire alloué par la fonction d'allocation (la restitution doit se faire avec la bonne fonction de libération).

On s'attachera à donner rigoureusement l'interface et l'implémentation de ce module.

4) – 3 points – S'il se produit un défaut de mémoire lors d'une opération d'allocation, le comportement du module est le suivant :

   - réinstaller les fonctions d'allocation par défaut;
   - lever une exception.

   On suppose que si l'erreur se produit dans une fonction d'allocation installée par l'utilisateur, la même exception est levée par cette fonction (on s'appuie sur le mécanisme d'exceptions présenté en cours). Modifier l'implémentation du module `blocmem` pour obtenir ce comportement.

5) – 2 points – Considérons une suite imbriquée $f_1, f_2, \ldots, f_k$ d'appels de fonctions et supposons qu'un défaut de mémoire se produise durant le déroulement de f_k. Supposons également que l'exception ne soit récupérée que dans la fonction f_1. Quel problème peut-il se produire si des allocations ont été effectuées par les fonctions intermédiaires ?

6) – 1 point – Existe-t-il une solution à ce problème (préciser votre réponse) ?