Informatique **DEUG A1** **Examen** **Juin 1991**

Durée 2 heures. Aucun document. Les questions sont indépendantes.

A la fin de l'examen, déposer votre copie dans le paquet de votre groupe de td.

Question 1
Evaluer les expressions suivantes:
a- (cdr (quote (1 2)))
b- (car '(car (1 2 3 4)))

Question 2
Représenter au choix, par boîtes et pointeurs ou sous forme d'arbre, la liste suivante:
((a b) (() c))

Question 3
On représente un *angle* par une liste à deux éléments contenant son cosinus et son sinus. Faire le constructeur creer-angle, et les sélecteurs cos-angle et sin-angle correspondants. Faire une fonction somme-angles qui additionne deux angles (On rappelle que cos(a+b)=cosa*cosb-sina*sinb, et sin(a+b)=sina*cosb+sinb*cosa).

Question 4
Faire une fonction qui recherche, au moyen d'un processus *itératif*, le plus grand élément d'une liste d'entiers supposée sans répétition et à un seul niveau. On utilisera la fonction max à deux arguments. Exemple: (max 4 7) ---> 7.

Question 5
Faire une fonction (circule n l) qui réalise une permutation circulaire de la liste l en plaçant n fois de suite le premier élément à la fin. Exemple:
(circule 1 '(a b c)) ---> (b c a)
(circule 5 '(a b c)) ---> (c a b)

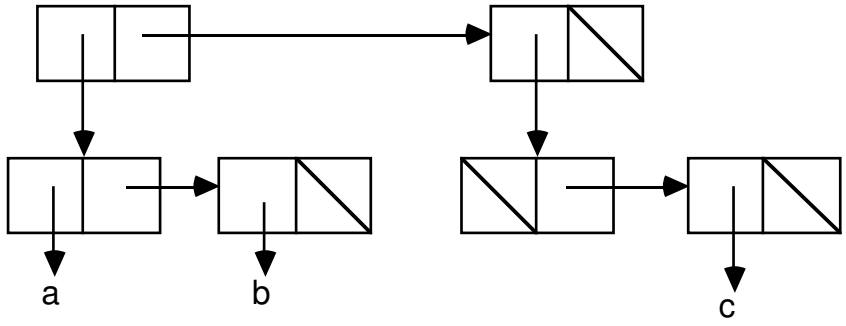
Question 6
On représente un *polynôme* par une liste de couples (coefficient degre) ordonnés à degrés décroissants. On suppose que les coefficients sont strictement positifs. Par exemple, $x^2 + 3$ est représenté par ((1 2) (3 0)). Faire une fonction somme-poly qui calcule la somme de deux polynômes.

EXAMEN INFORMATIQUE DEUG A1 **FORMULAIRE DE REPONSE** **Juin 1991**

NOM :
Prénom :
Numéro de groupe de td : **Numéro d'appel :**

1a (2) **1b** car

2



3 Constructeur: (define (creer-angle a b) (list a b))

Sélecteurs: (define (cos-angle teta) (car teta))
(define (sin-angle teta) (cadr teta))

Somme:

```
(define (somme-angles teta1 teta2)
.  (creer-angle (- (* (cos-angle teta1) (cos-angle teta2)) (* (sin-angle teta1) (sin-angle teta2)))
.               (+ (* (sin-angle teta1) (cos-angle teta2)) (* (cos-angle teta1) (sin-angle teta2)))))
```

4 Itérative:

```
(define (max-iter l)
.       (define (aux l emax)
.               (cond  ((null? l) emax)
.                      (else (aux (cdr l)
.                                 (max emax (car l))))))
.       (if (null? l) () (aux (cdr l) (car l))))
.
```

5

```
(define (circule n l)
.    (define (circule-aux k l)
.       (cond ((= k 0) l)
.             (else (circule-aux (- k 1) (append (cdr l) (list (car l)))))))
.    (if (null? l) () (circule-aux (remainder n (length l)) l)))
.  ;j'emprunte l'utilisation de remainder à une bonne copie d'étudiant...
```

6

```
(define (somme-poly p1 p2)
.   (cond ((null? p1) p2)
.         ((null? p2) p1)
.         ((> (cadar p1) (cadar p2)) (cons (car p1) (somme-poly (cdr p1) p2)))
.         ((< (cadar p1) (cadar p2)) (somme-poly p2 p1))
.         (else (cons (list (+ (caar p1) (caar p2)) (cadar p1)) (somme-poly (cdr p1) (cdr p2))))))
```