Licence 1ere année UE EDI (2020-2021)

TP N°2 : Recursivité

Vous devez retourner le fichier ".scm" contenant vos réponses à ces exercices.

Pour ce faire vous enverrez un mail avec votre fichier attaché, ayant comme **sujet** :
TP2_SCHEME_NOM1_PRENOM1_NOM2_PRENOM2

Adresse mail : santucci@univ-corse.fr

Veillez respecter ce format, sinon votre envoi risque de n'être pas pris en compte…

1. **Exercices sur les ensembles**

On représente les ensembles par des listes d'éléments.
a-Ecrire une fonction Union qui prend deux listes l1 et l2 en argument et qui retourne la liste représentant l'union des ensembles décrits par l1 et l2
Exemple:
>(union '(a b c d) '(b c d e))
=(a b c d e)

b-Ecrire la fonction inter qui prend deux listes l1 et l2 en argument et qui retourne la liste représentant l'intersection des ensembles décrits par l1 et l2
Exemple :
>(inter '(a b c d) '(b c d e))
=(b c d)

2. **Twice**

Ecrire la fonction twice qui prend en arguments deux listes l1 et l2 et qui retourne une nouvelle liste où chaque élément de la liste reçue en entrée est multiplié par deux.
Exemple :
>(twice '(1 2 3 4 5))
= (2 4 6 8 10)

3. **Merge**

Ecrire une fonction merge qui prend 2 listes de nombre de même longueur comme arguments. Elle additionne les nombres correspondant de chaque liste (ayant la même position dans la liste) et retourne le produit des nombres résultat.
Exemple :
>(merge '(1 2 3) '(2 2 2)) → 60 car (1+2)*(2+2)*(3+2)=60

4. **Compare-liste**

Ecrire la fonction compare-liste qui prend en arguments une liste d'éléments et qui retourne le nombre d'éléments identiques et à la même place.
Exemple :
>(compare-liste '(a b c d) '(a d e f))
= 1
>(compare-liste '(a b c d) (a c b d))
=2

5. Count

Ecrire la fonction count qui prend un atome s et une liste l d'atomes en argument et retourne le nombre d'occurrences de s dans l

Exemple :

> (count 'b '(a b c b d b e))

= 3

6. Compt

Ecrire une fonction permettant de compter le nombre d'entiers pairs dans une liste d'entiers naturels.

Exemple :

>(Compt '(1 3 4 5 7 9)) → 1

>(Compt '(2 3 4 6 8)) → 4

7. Max-sous-liste

Ecrire la fonction Max-sous-liste qui prend une liste L en arguments – les éléments de la liste L sont des listes - et qui retourne la longueur maximale des sous-listes ; on écrira la fonction de deux manières : a- recusive ; b- à l'aide d'une combinaison de MAP et APPLY.

Exemple

>(Max-sous-liste '((t y u) (1 2 3 4) (6 8) (y))

=4

8. NBDIST

Définir la fonction NBDIST permettant de calculer le nombre d'éléments distincts d'une liste d'atomes.

>(NBDIST '(a b c a d e b a a))

= 5

>(NBDIST '(1 3 4 5 5))

=4

9. a- Notre-append

Ecrire la fonction notre-append identique à la fonction NOTRE-APPEND sans utiliser append et des fonctions auxiliaires.

b- VALIS

Ecrire la fonction VALIS à 1 argument (une liste <l>) qui retourne une liste composée des valeurs des évaluations de tous les éléments de la liste <l>.

Exemple :

>(valis '((car '(1 2)) "oo" (* 4 5)))

=(1 "oo" 20)

10. Remove-first

Ecrire la fonction remove-first qui prend en arguments un atome s et une liste d'atomes latom et qui retourne une liste avec les mêmes éléments et dans le même ordre que la liste latom à l'exception de la première occurrence de l'atome s qui est enlevée. Si cet atome s n'appartient pas à latom, remove-first retourne latom inchangée.

Exemple :

> (remove-first 'b '(a b c b d e))

= (a c b d e)

11. Remove-all

Ecrire la fonction remove-all qui prend en arguments un atome s et une liste d'atomes latom et qui retourne une liste où toutes les occurrences de l'atome s ont été retirées. On ne considère que les atomes apparaissant au premier niveau de la liste comme dans le deuxième exemple.

Exemple :

> (remove-all 'b '(a b c b d e))

= (a c d e)

> (remove-all 'b '(a b c (b) d e))

= (a c (b) d e)

12. Remove-k-first

Ecrire la fonction remove-k-first qui enlève d'une liste l passée en paramétre les k premières occurrences d'un élément donné e (l k et e sont passés en arguments).

Exemple :

> (remove-k-first 'a 2 '(1 2 a 3 4 a 5 6 a 7))

= (1 2 3 4 5 6 a 7)

13. Notre-reverse

Ecrire la fonction notre-reverse identique à la fonction NOTRE-REVERSE sans utiliser append et….. reverse

Indication : utiliser une fonction auxiliaire reverse-bis qui aura deux listes comme arguments et qui empilera les éléments de l'une dans l'autre.

14. Alternate

Ecrire la fonction alternate qui prend en argument une liste d'atomes l et qui retourne une liste où les atomes de rang pair ont été permutés avec les atomes de rang impair ; c'est-à-dire que l'on permute le premier et le deuxième élément, les troisième et le quatrième et ainsi de suite.

Exemple :

> (alternate '(a b b c d e f))

= (b a d c f e)

>(alternate '(a b c d e f g)

=(b a d c f e g)

15. Longest

Ecrire la fonction longest qui retourne la plus longue liste d'une liste de listes donnée en arguments

Exemple :

> (longest '((1 3) (a b c d) () (a b)))

= (a b c d)

16. Stammer-n

Ecrire la fonction stammer-n qui prend une liste d'atomes en entrée et un nombre n et qui retourne une liste où tous les symboles sont répétés n fois.
Exemple :
> (stammer-n '(a b c d) 4)
= (a a a a b b b b c c c c d d d d)

17. Listref

Ecrire la fonction listref qui retourne l'élément de position n dans une liste l. (attention le premier élément d'une liste est en position 0).
Exemple :
> (listref '(a b c d e f) 0)
= a
> (listref '(a b c d e f) 3)
= d

18. List-Tail

Ecrire la fonction list-tail appliquant successivement n fois la fonction cdr à une liste donnée.
Exemple :
> (list-tail '(a b c d e) 0)
= (a b c d e)
> (list-tail '(a b c d e f) 3)
= (d e f)

19. recherche de doublons

Extraire d'une liste de symboles ceux qui y figurent au moins deux fois. La réponse peut comporter des répétitions ; son ordre n'est pas important.
Exemple : si poeme-pi vaut (Q U E J A I M E A F A I R E A P P R E N D R E U N N O M B R E U T I L E A U S A G E)
>(doublon poeme-pi) → (U E A I M E A A I R E A P R E N R E U N E U E A)

20. Odds

Ecrire la fonction odds qui prend une liste d'atomes en entrée et qui a pour valeur la liste de tous les atomes de rang impair.
Exemple :
> (odds '(a b c d e f))
= (a c e)
> (odds '(a b c d e f g))
= (a c e g)

21. Follow

Ecrire la fonction follow qui, pour un atome s et une liste d'atomes l, retourne l'ensemble de tous les atomes de l qui suivent directement les occurrences de s. Lorsque l'atome s apparait en dernière position de la liste il n'est suivi par aucun symbole. Conventionnellement on choisit de retourner la liste vide pour marquer ce fait.

Exemple :

> (follow 'b '(a b c b d e)
= (c d)
> (follow 'b '(a b c b d b)
= (c d ())
> (follow 'b '(a b b d b e)
= (b d e)
> (follow 'b '(a c f)
= ()

22. Precede

Ecrire la fonction precede qui, pour un atome s et une liste d'atomes l, retourne l'ensemble de tous les atomes de l qui apparaissent directement avant les occurrences de s. Lorsque l'atome s apparait en première position de la liste il n'est précédé par aucun symbole. Conventionnellement on choisit de retourner la liste vide pour marquer ce fait.

Exemple :

> (precede 'b '(a b c b d e)
= (a c)
> (follow 'b '(a b c b b d)
= (a c b)
> (precede 'b '(b a c b d b)
= (() c d)

23. Collect-number

Définir la fonction collect-number qui retourne une liste de tous les nombres d'une S-expression passée en paramètre. La S-exp peut être un atome, une liste ou une liste de S-expression.

>(collect-number 1) → (1)
>(collect-number '(a)) → NIL
>(collect-number '(1 (b (2 c) ((3))))) → (1 2 3)

24. deep-map

Ecrire la fonction deep-map qui est l'équivalent de map mais qui travaille en profondeur – Deep-map prend 2 arguments comme map -une fonction f et une liste L d'éléments. La fonction passée en argument dans le cas de deep-map est une fonction à 1 argument qui n'est pas une liste.

Exemple

>(map (lambda (x) (+ x 10)) '(10 (3 100) 9 64)
= erreur car (3 100) n'est pas un nombre
>(deep-map (lambda(x) (+ x 10)) '(10 (3 100) 9 64)
= (20 (13 110) 19 74)

25. Permutation

Ecrire une abstraction fonctionnelle de type fonction permettant de déterminer toutes les permutations possibles obtenues à partir d'un ensemble d'entiers.
Exemple :
>(permute '(1 2 3)) → ((1 2 3) (1 3 2) (3 2 1) (3 1 2) (2 1 3) (2 3 1))
>(permute '(1 2)) → ((1 2) (2 1))
>(permute '(1)) → ((1))
Pour cela on considèrera que permute est une fonction ayant comme argument une liste avec des entiers comme éléments et que la fonction retourne une liste avec comme éléments des listes d'entiers.
Indication : Il semble judicieux d'utiliser une fonction auxiliaire permute-bis ayant deux arguments afin de construire la liste résultat.