



MIAS 1^oA, UE SM22
Informatique – Examen, 1^{ère} session - Grenoble, Valence

Durée 3h, sans documents, appareils électroniques strictement interdits

5 Mai 2003

Il est demandé de respecter le lexique des énoncés. Le barème donné est indicatif. Les exercices sont indépendants. REPENDRE UNIQUEMENT SUR LA FEUILLE FOURNIE EN ANNEXE

Les solutions seront données en utilisant la notation fonctionnelle du cours et des travaux dirigés (et non CAML).

Rappel des notations concernant les séquences

- Constructeurs : [], S•e, e o S ; testeur et sélecteurs : EstVide?, début, dernier, premier, fin.
- La concaténation est notée par le symbole &.

Rappel des notations concernant les arbres

- Constructeurs : /\, /G, r, D\, //r\, //G,r\, //r,D\, //G,r,D\
- Testeurs: EstVide?, EstBinaire?, EstUnaireGauche?, EstUnaireDroit?, EstSingleton?.
- Sélecteurs : Racine, Gauche, Droit

1. Vérification des types des noms d'une expression algébrique [3 points]

On s'intéresse ici à la vérification des types des noms qui apparaissent dans une expression algébrique par rapport aux spécifications des opérations qu'elles mettent en jeu. Par exemple, pour que l'expression **(3 o a)•premier(b)** soit correcte du point de vue des types, les contraintes suivantes doivent être respectées: **a** doit être de type séquence d'entiers et **b** de type séquence non vide d'entiers. Si ces conditions sont respectées, l'expression **(3 o a)•premier(b)** est de type séquence non vide d'entiers.

Q1.

— Pour chacune des expressions nommées E1, E2, E3 et E4 données ci-dessous, donner les contraintes de types que doivent respecter les noms y apparaissant, puis le type de l'expression si ces contraintes sont respectées.

E1 : <a, (a=b), (b < 1), (b > 1), <a,b> >

E2 : si (d=c)=c alors [(c = d) et c] sinon c o [((c = c) ou alors (d=d)) , c]

E3 : < premier(e o "e=f") , e o "faux" > o [<"f", [] >, premier(f)]

E4 : soit <g,h> = <\ o [/ \, "abc", \ \ , \] , faux> dans si h alors dernier(g) sinon \

2. Extraction à intervalles réguliers des éléments d'une séquence [3 points]

Dans cet exercice on souhaite réaliser une fonction nommée SSeq retournant les éléments d'une séquence se trouvant à une position multiple d'un entier strictement positif donné. La spécification de cette fonction est donnée ci-dessous.

SSeq : un entier > 0, une séquence d'Eléments → une séquence d'Eléments

{ SSeq(n,s) est la séquence des éléments de s dont la position est un multiple de n.

Par convention la position du premier élément de la séquence est 1.

Par exemple SSeq(2,"kayak") = "aa", SSeq(5,"particulièrement") = "ièn"

SSeq(1,"kayak") = "kayak", SSeq(17,"kayak") = "" }

Pour réaliser la fonction SSeq, on introduit une fonction intermédiaire nommée F partiellement spécifiée ci-dessous.

F : un entier > 0, un entier > 0, une séquence d'Eléments → une séquence d'Eléments

{ $F(\text{début}, n, s)$ est la séquence des éléments de s }

Par exemple $F(2, 5, \text{"pédagogiquement"}) = \text{"égm"}$, $F(3, 5, \text{"pédagogiquement"}) = \text{"....."}$ }

Q2.

— Compléter la spécification de F fournie ci-dessus.

— Donner une réalisation pour la fonction SSeq en se basant sur la fonction F ainsi spécifiée.

— Donner une définition récursive de F sans introduire d'autres fonctions. On utilisera une combinaison systématique de 2 modèles vus en cours donnant lieu à 4 équations.

3. Texte représentant une séquence d'entiers [7 points]

On souhaite réaliser une fonction nommée SeqVTexte retournant la représentation d'une séquence d'entier sous forme de texte. La spécification de cette fonction est donnée ci-dessous.

SeqVTexte : une séquence d'entiers → un texte non vide

{ $SeqVTexte(s)$ est le texte représentant la séquence d'entiers s en utilisant la notation du cours: les éléments sont séparés par une virgule, et la séquence est délimitée par des crochets.

Par exemple $SeqVTexte([12, -30, 0, 1]) = \text{"[12, -30, 0, 1]"}$, $SeqVTexte([]) = \text{"[]"}$

$SeqVTexte([-10, -3]) = \text{"[-10, -3]"}$, $SeqVTexte([10]) = \text{"[10]"}$ }

On dispose pour cela de deux fonctions nommées respectivement x et y . Ces deux fonctions, dont la réalisation figure ci-dessous, ont été programmées par un programmeur débutant n'ayant aucune conscience de l'importance ni des spécifications, ni des noms. Il s'agit de retrouver la signification précise de ces fonctions ainsi que de corriger les éventuelles erreurs.

$x(a, b)$: si $a=1$ alors premier(b) sinon $x(a-1, \text{fin}(b))$ {1}

$y(c, d)$: soit $e = x((c \text{ reste } d) + 1, \text{"0123456789ABCDEF"})$, $f = (c \text{ quotient } d)$ {2}

dans si $f=0$ alors e {3}

sinon $y(f, d) \cdot e$ {4}

Q3.

— Pour chacune des expressions nommées **E1**, **E2**, **E3** et **E4** ci-dessous, donner leur valeur ou indiquer si l'expression est incorrecte.

E1: $x(1, \text{"un"})$ **E2**: $x(2, \text{"deux"})$ **E3**: $x(0, \text{"trois"})$ **E4**: $x(7, \text{"quatre"})$

— Proposer des noms significatifs pour remplacer les noms x , a et b , et ce afin de faire ressortir la signification de la fonction.

— Donner une spécification très précise de la fonction x en indiquant notamment les contraintes éventuelles sur les paramètres sous forme de précondition.

Q4.

— La fonction nommée y est incorrecte du point de vue des règles de typage. Indiquer comment corriger cette erreur en proposant une modification de la ligne {3}.

— Proposer des noms significatifs pour remplacer les noms y , c , d , e , et f , et ce afin de faire ressortir clairement la signification des noms.

— Donner une spécification très précise de la fonction ainsi corrigée et renommée en faisant particulièrement attention aux cas limites et en donnant des exemples d'utilisation.

Q5.

— Donner une réalisation pour la fonction nommée EntierVTexte spécifiée ci-dessous en se basant sur la fonction nommée y dans la question précédente,

EntierVTexte : un entier → un texte non vide

{ $EntierVTexte(n)$ est le texte représentant l'entier relatif n en base 10.

Par exemple $EntierVTexte(143) = \text{"143"}$, de même $EntierVTexte(-17) = \text{"-17"}$ }

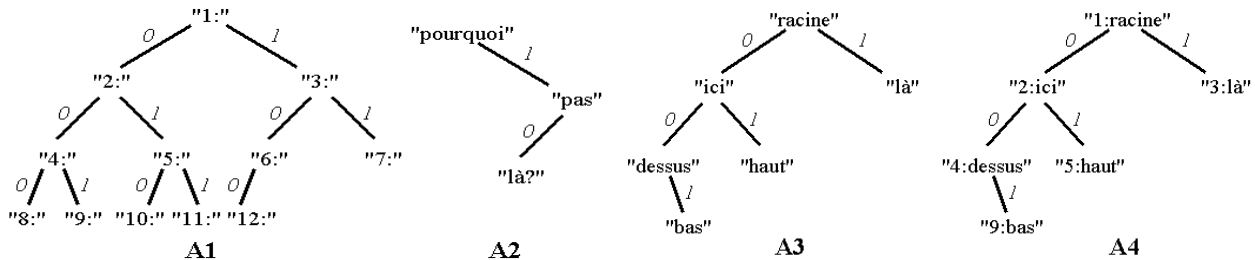
Q6.

— Donner une réalisation pour la fonction SeqVTexte en introduisant une seule fonction intermédiaire qui sera préalablement spécifiée de manière très précise et pour laquelle on donnera des exemples d'utilisation. Cette fonction intermédiaire sera nommée SuiteDEntiers. Une attention toute particulière sera apportée à la position des virgules. Il faut remarquer notamment que le dernier entier n'est pas suivi d'une virgule.

— Donner une définition récursive de la fonction SuiteDEntiers basée sur 3 équations récursives.

4. Positions des nœuds dans un arbre [8 points]

Chaque nœud d'un arbre binaire quelconque peut être identifié de manière unique par un entier naturel non nul. On parlera alors de la **position** d'un nœud dans un arbre pour faire référence à cet entier. Par exemple dans l'arbre A1 les nœuds de l'arbre sont décorés par un texte correspondant à la position de chaque nœud représenté en base 10 suivi par le caractère ":". De même l'arbre A4 correspond à l'arbre A3 dans lequel le texte contenu dans chaque nœud de l'arbre est préfixé par la position du nœud. La position de la racine est l'entier 1. Pour un sous-arbre quelconque dont la racine est à une position n , la racine du sous arbre gauche (si elle existe) est à la position $2*n$, alors que la racine du sous arbre droit (si elle existe) est à la position $2*n+1$.



Dans le cadre de ce problème il est utile de raisonner en terme de chemins d'accès et de représentation binaire. Dans ce problème le **chemin d'accès** à un nœud correspond à la séquence des bits (0 ou 1) portés sur les arcs permettant d'aller de la racine à un nœud donné. Par convention on associe la valeur 0 aux arcs orientés vers la gauche, et 1 aux arcs orientés vers la droite. Les arcs des arbres A1, A2, A3 et A4 ont été ainsi décorés. Par exemple le chemin d'accès au nœud étiqueté "bas" de l'arbre A3 peut être représenté par la séquence [0,0,1] alors que le chemin d'accès au nœud étiqueté "ici" peut être représenté par la séquence [0].

Il existe une relation directe entre le chemin d'accès à un nœud et la position de ce nœud: pour passer d'un chemin d'accès à une position il suffit d'ajouter le bit 1 en début de séquence et de convertir la séquence de bits ainsi obtenue en un entier. Par exemple la position de "bas" dans l'arbre A3 est 9, soit l'entier 1001 en base 2, alors que la position de "ici" est l'entier 2, soit l'entier 10 en base 2.

Pour formaliser les différents concepts décrits ci-dessus le lexique ci-dessous est introduit.

Position : le type entier >0

Bit : le type entier sur [0,1]

SéqBits : le type séquence non vide de Bits { dont le premier élément est toujours le bit 1 }

EvB : un entier >0 → une SéqBits

{ EvB(n) est la séquence bits représentant n en base 2. Par exemple EvB(9)=[1,0,0,1] car $9=1*2^3+0*2^2+0*2^1+1*2^0$. De même EvB(4)= [1,0,0] et EvB(1)=[1] }

BvE : une SéqBits → un entier >0

{ BvE(s) est l'entier représenté par la séquence de bits s . Par exemple BvE([1,0])=2 }

CheminDAccès : le type séquence de Bits

NoeudPositionné : le type < pos : une Position, val : un texte >

SéqNP : le type séquence de NoeudsPositionnés

SNP1 : la SéqNP [<1,"racine">,<2,"ici">,<4,"dessus">,<9,"bas">,<5,"haut">,<3,"là">]

SNP2 : la SéqNP [<1,"ici">,<2,"dessus">,<5,"bas">,<3,"haut">]

SNP3 : la SéqNP [<1,"pourquoi">,<3,"pas">,<6,"là?">]

SNP4 : la SéqNP [<2,"ici">,<4,"dessus">,<9,"bas">,<5,"haut">]

SNP5 : la SéqNP [<3,"ici">,<6,"dessus">,<13,"bas">,<7,"haut">]

ValeurPos : une Position, un arbre binaire de texte → un booléen, un texte

{ ValeurPos(pos,A) est un couple <ok,val>. Si aucun nœud dans A ne se trouve à la position pos alors ok a la valeur faux et val a la valeur "". Dans le cas contraire ok a la valeur vrai et val correspond à la valeur du nœud se trouvant à la position pos dans A . Par exemple ValeurPos(11,A1)=<vrai,"11:">, ValeurPos(6,A2)=<vrai,"là?">, ValeurPos(2,A3)=<vrai,"ici">, ValeurPos(5,A2)=<faux,""> }

Q7.

— Donner la valeur des expressions suivantes.

E1: EvB(13)**E3:**ValeurPos(BvE(1 o [0,0,1]),A4)**E2:** BvE(1 o [1,1,0])**E4:**ValeurPos(BvE(1 o [0,1,0]),A4)

— Donner des définitions récursives de BvE et de EvB en s'attachant à montrer la symétrie entre ces deux fonctions.

Q8.

Pour définir la fonction ValeurPos on introduit une fonction intermédiaire nommée ValeurCA prenant comme premier paramètre un chemin d'accès. La signature de cette fonction est la suivante.

ValeurCA : un CheminDAccès, un arbre binaire de texte → un booléen, un texte

— Donner la spécification de fonction ValeurCA et donner 2 exemples utilisant l'arbre A4.

— Donner une réalisation de la fonction ValeurPos en utilisant la fonction ValeurCA.

— Donner une définition récursive de ValeurCA en se basant sur une combinaison de modèles donnant lieu à 6 équations.

Dans les questions suivantes on désire pouvoir représenter des arbres sous forme de séquences de "nœuds positionnés" et inversement retrouver l'arbre correspondant à une telle séquence. On spécifie pour cela deux fonctions nommées ArbreVSeqNP et SeqNPVArbre.

ArbreVSeqNP : un arbre binaire de texte → une SeqNP

{ ArbreVSeqNP(A) est la séquence des couples formés par la position et la valeur de chaque nœud de A, les nœuds étant ordonnés via un ordre préfixé.

Par exemple ArbreVSeqNP(A2) = SNP3, ArbreVSeqNP(A3)=SNP1 }

SeqNPVArbre : une SeqNP → un arbre binaire de texte

{ SeqNPVArbre(snp) est un arbre contenant un nœud pour chaque nœud positionné de snp.

Le texte vide est associé à tous les nœuds internes de l'arbre dont la valeur n'est pas définie dans snp. Si deux nœuds positionnés de snp correspondent à la même position, seule la dernière valeur sera prise en compte.

Par exemple SeqNPVArbre (SNP3)=A2, SeqNPVArbre (SNP1)=A3

*SeqNPVArbre([<4,"4:">,<3,"3:">,<6,"6:">]) = // // // "4:"\, ""\, "", // // "6:"\, "3:" \ \ \ *

*SeqNPVArbre([<2,"non">,<1,"racine">,<2,"oui">]) = // // "oui"\, "racine" \ *

Q9.

Pour définir la fonction ArbreVSeqNP, on introduit une fonction intermédiaire nommée Ajuste.

Ajuste : une SeqNP, un Bit → une SeqNP

{ Soit snpB = Ajuste(snpA,x). L'arbre représenté par snpA correspond au sous arbre gauche de l'arbre représenté par snpB si x vaut 0, ou au sous arbre droit de snpB si x vaut 1.

Par exemple Ajuste(SNP2,0)=SNP4, Ajuste(SNP2,1)=SNP5 }

— Donner une définition récursive de la fonction ArbreVSeqNP en utilisant la fonction Ajuste.

— Donner une définition récursive pour la fonction Ajuste en utilisant uniquement comme fonction intermédiaire les fonctions nommées EvB et BvE. On raisonnera sur les modifications devant être apportées aux représentations binaires des positions.

Q10.

Pour définir la fonction SeqNPVArbre, on introduit une fonction intermédiaire nommée Modifie spécifiée ci-dessous.

Modifie : un CheminDAccès, un texte, un arbre binaire de textes → un arbre binaire de textes

{ Modifie(cha, texte, A) est l'arbre A dans lequel le nœud dont la position correspond au chemin d'accès cha a été créé ou inséré. La valeur de ce nœud est texte. Si des nœuds sont manquants dans A pour former le chemin cha, alors ces nœuds seront créés et décorés par le texte vide.

*Par exemple Modifie([1,0], "ici", A2) = // "pourquoi", // // "ici"\, "pas" \ \ \ *

Modifie([1,0], "ici", /\, "pourquoi", /\) = // "pourquoi", // // "ici"\, "" \ \ \ }

— Donner une définition récursive de SeqNPVArbre en utilisant uniquement les fonctions Modifie et EvB.

— Donner une définition récursive de Modifie sans utiliser de fonction intermédiaire. On fera particulièrement attention à faire ressortir la combinaison de modèles existant.