Programmation logique et fonctionnelle

Examen terminal - deuxième session

Modalités

Les documents de cours, TD et TP, ainsi que les notes personnelles, sont autorisés. Les calculatrices sont autorisées. Il sera tenu compte de la clarté et de la concision de vos réponses.

1 λ-calcul (4 points)

Soit T le terme de λ-calcul suivant :

$$(\lambda x.(xx)(\lambda y.yy))(\lambda x.x)$$

1. Ce terme comporte t-il des occurrences de variables libres, et si oui lesquelles ?
2. Ce terme comporte t-il des redex et si oui lesquels ?
3. Donnez toutes les étapes de l'évaluation complète de ce terme.

2 Logique propositionnelle (3 points)

2.1 Satisfaisabilité et validité

Soit ϕ la formule suivante :

$$\phi = (\ ((a \vee b) \rightarrow c) \wedge ((a \vee d) \rightarrow c)\ ) \rightarrow (a \rightarrow c)$$

Cette formule est elle satisfaisable ? Est elle valide ? Justifiez votre réponse.

2.2 Modélisation

On considère une urne, une boule rouge, une boule bleue, et une boule verte. Chaque boule est susceptible de se trouver ou non dans l'urne, et rien d'autre ne peut se trouver dans l'urne. Par convention, la variable b vaut `vrai` si et seulement si la boule bleue est dans l'urne, la variable r vaut `vrai` si et seulement si la boule rouge est dans l'urne, et la variable v vaut `vrai` si et seulement si la boule verte est dans l'urne.
Proposez une formule modélisant la propriété « Il y a exactement deux boules dans l'urne ».

3 Logique des prédicats (3 points)

Soit ϕ la formule suivante :

$$(\ \neg(\forall X \forall Y p(X,Y))\ ) \rightarrow (\ \forall X \exists Y \neg p(X,Y)\ )$$

Cette formule est elle satisfaisable ? Est elle valide ? Justifiez votre réponse.

4 CAML (5 points)

4.1 Listes

Proposez le définition complète d'une fonction CAML `addl` telle que si `w` est une liste d'entiers alors `addl w x` retourne la liste obtenue en ajoutant l'entier `x` à chaque élément de `w`. Par exemple, `addl [2; 2; 3], 2`

retourne `[4; 4; 5]`.

4.2 Arbres

On représente par le type suivant des arbres binaires étiquetés par des entiers appelés poids :

```
type arbre = F | N of int * arbre * arbre;;
```

On appelle signature d'un chemin la liste des poids associées à ses noeuds depuis la racine jusqu'au noeud ayant pour fils la feuille qui termine ce chemin. Par exemple, un chemin ayant pour signature `[2;1;3]` commence par la racine, qui a un poids égal à 2, puis continue avec un des fils de la racine qui a un poids égal à 1, puis continue avec un des fils de ce noeud qui a un poids égal à 3 puis se termine par une feuille. Complétez la définition suivante de manière à ce que si `t` est un arbre et `w` une liste d'entier, `csign t w` retourne `true` si `t` comporte un chemin ayant pour signature `w,` et `false` sinon.

```
let rec csign a w =
match w with
  |[] -> a = F
  |t::q ->
     match a with
       |F -> .....
       |N(p,g,d) ->
         if p<>t then .....
         else if csign ..... then true
         else csign ..... ;;
```

5 PROLOG (5 points)

5.1 Faits et clauses

On considère un prédicat planter définit par des faits qui mettent en relation un légume et les mois pendant lesquels on peut le planter. Par exemple les faits suivants :

```
planter(choux, mai).
planter(choux, juin).
planter(choux, juillet).

planter(haricot, avril).
planter(haricot, mai).
```

spécifient qu'on peut planter les choux en mai, juin ou juillet et les haricots en avril ou en mai. On suppose que d'autres faits peuvent être ajoutés. Spécifiez à l'aide d'une ou plusieurs clauses chacun des prédicats suivants :

1. `p1` tel que `p1(X)` soit satisfait si `X` est un légume pouvant être planté en avril ou si `X` est un légume pouvant être planté en juillet ;
2. `p2` tel que `p2(X)` soit satisfait si `X` est un légume pouvant être planté en avril et en juillet ;
3. `p3` tel que `p3(X)` soit satisfait si `X` est un mois au cours duquel on peut planter au moins deux légumes différents.

5.2 Listes

Complétez la spécification suivante de manière à ce que le but `scd(N, L, A, B)` permette de scinder une liste `L` en deux liste `A` et `B` telle que `A` contienne les `N` premiers éléments de `L` et `B` les éléments restants. Par exemple, l'évaluation de `scd(2, [1,2,3,4], A, B)` doit avoir pour résultat `A = [1,2]  B = [3,4]`.

```
scd(0,L, ....., .....) :- !.
scd(N,[T|Q],[T|R],W) :- N1 is ....., scd( ..... ).
```