Contrôle Continu - Mars 2009

Durée : 1h30

Les notes de cours, de travaux dirigés et de travaux pratiques sont autorisées. Le sujet comporte deux pages et trois parties. Le barème est donné à titre indicatif. Vous vous attacherez à soigner la présentation générale et l'orthographe, et vous serez notamment vigilant quant à la syntaxe lors de l'écriture de fragment de code devant être accepté par le système Coq.

1 Questions de cours (8 points)

Question 1 On rappelle les types de quelques constantes :

```
or_ind : forall A B P : Prop, (A -> P) -> (B -> P) -> A \/ B -> P
or_intror : forall A B : Prop, B -> A \/ B
or_introl : forall A B : Prop, A -> A \/ B
and_ind : forall A B P : Prop, (A -> B -> P) -> A /\ B -> P
conj : forall A B : Prop, A -> B -> A /\ B
```

Précisez les types des trois termes de preuve suivants :

```
a) fun (A B : Prop) (H : A /\ B) =>
and_ind A B A (fun (H0 : A) (_ : B) => H0) H

b) fun (A B C : Prop) (H : (A \/ B) /\ C) =>
       and_ind (A \/ B) C (A /\ C \/ B /\ C)
         (fun (H0 : A \/ B) (H1 : C) =>
          or_ind A B (A /\ C \/ B /\ C)
            (fun H2 : A => or_introl (A /\ C) (B /\ C) (conj A C H2 H1))
            (fun H2 : B => or_intror (A /\ C) (B /\ C) (conj B C H2 H1)) H0)
         H

c) fun (A B C : Prop) (HABC : A -> B -> C) (HB : A -> B) (HA : A) =>
HABC HA (HB HA)
```

Question 2 On modifie la définition usuelle des entiers naturels :

```
Inductive natbis : Set := O : natbis | 1 : natbis | plus2 : natbis -> natbis.
```

Ecrivez le principe de raisonnement par induction associé à cette définition.

2 Réels et fonctions (6 points)

Question 3 Proposer un type inductif `fonction` permettant de représenter les fonctions (toutes dérivables en tout point de R) qui sont obtenues à partir de l'identité, des fonctions constantes, de l'exponentielle et des fonctions trigonométriques (sinus et cosinus) et par l'application des opérations d'addition, soustraction, produit, composition de fonctions déjà obtenues.

```
Inductive fonction : Set := ...
```

Question 4 Comment définir en Coq les fonctions suivantes $x \mapsto sin(x) + exp(x) * cos(x)$ et $x \mapsto x * sin(x^2)$ avec le type de données de la question précédente ?

Question 5 Programmer une fonction d'interprétation `interp : fonction -> (R -> R)` interprétant une expression de type `fonction` comme une fonction de R dans R.

3 Listes (6 points)

Question 6 Rappelez la définition des listes d'entiers en Coq. Rappelez également l'énoncé du principe d'induction sur les listes tel qu'il est défini en Coq.

Question 7 On considère une liste des effectifs d'un lycée, chacun élément de la liste est le nombre d'élèves dans une classe donnée. On souhaite transformer cette liste pour avoir la liste des effectifs cumulés croissants des élèves de toutes les classes déjà considérées. Par exemple, si on a la liste suivante en entrée [12 ; 35 ; 20 ; 30], la liste des effectifs cumulés sera [12 ; 47 ; 67 ; 97], ce qui correspond à [12 ; 12+35 ; 12+35+20 ; 12+35+20+30]. Le dernier élément de la nouvelle liste donne l'effectif total d'élèves dans le lycée.

Programmez en Coq une fonction passant de la liste des effectifs à la liste des effectifs cumulés croissants. Vous pouvez utiliser toutes les opérations sur les listes vues en cours et en TDs (en rappelant leur comportement).

Question 8 Supposons maintenant que l'on dispose d'une liste d'entiers quelconques, comment tester en Coq si cette liste est triée dans l'ordre croissant ? On pourra utiliser le théorème suivant `le_dec : forall n m:nat, {n<=m}+{n>m}` via le fragment de code Coq suivant qui permet d'écrire une expression conditionnelle :

```
if (le_dec n m) then (* n<=m *) ... else (* n>m *) ...
```

On pourra aussi imaginer faire un filtrage à deux niveaux de manière à pouvoir comparer les deux premiers entiers de la liste et traiter la suite récursivement.