TD-- Théorèmes pour les listes

Mots-clés preuve par induction, théorème
C.B version 2

Soit la spécification sucvante des listes linéaires LISTE0:

```
spéc LISTE0[EGAL0] étend BOOL,NAT
sortes Liste
opérations
    vide      :                →  Liste    -- liste vide
    adjt      : Liste x S      →  Liste    -- adjoindre en tête
    adjq      : Liste x S      →  Liste    -- adjoindre en queue
    supt      : Liste          →  Liste    -- supprimer la tête
    conc      : Liste x Liste  →  Liste    -- concaténer
    vide?     : Liste          →  Bool     -- test si vide
    dans?     : Liste x S      →  Bool     -- test d'appartenance
    tête      : Liste          →  S        -- la tête
    long      : Liste          →  Nat      -- longueur

variables l,l':Liste; x,y:S

préconditions

    tête(l) : non vide?(l)
    supt(l) : non vide?(l)

axiomes
    A1  vide?(vide)          = T
    A2  vide?(adjt(l,x))     = ⊥
    A3  dans?(vide,x)        = ⊥
    A4  dans?(adjt(l,y),x)   = x==y ou dans?(l,x)
    A5  tête(adjt(l,x))      = x
    A6  supt(adjt(l,x))      = l
    A7  long(vide)           = 0
    A8  long(adjt(l,x))      = succ(long(l))
    A9  adjq(vide,x)         = adjt(vide,x)
    A10 adjq(adjt(l,x),y)    = adjt(adjq(l,y),x)
    A11 conc(vide,l)         = l
    A12 conc(adjt(l,x),l')   = adjt(conc(l,l'), x)
fspéc
```

Les constructeurs choisis sont **vide** et **adjt**; **supt**, **adjq**, **conc** sont des générateurs secondaires.

QUESTION 0

On considère des listes de naturels.

1) Simplifier le terme
t = conc(adjq(adjt(vide,1),2), adjt(vide,3))

2) Démontrer, par induction structurelle, que toute liste peut se réécrire sous la forme canonique f=vide ou f=adjt(l,x), où l est une forme canonique.

3) En déduire le principe d'induction suivant sur les objets de sorte Liste:

Si P(vide) et $\forall$ **l,x. P(l)** $\Rightarrow$ **P(ajdt(l,x)) alors** $\forall$ **l. P(l)**

4) Trouver un autre jeu de constructeurs et décrire les axiomes correspondants.

QUESTION 1

Montrer que lg(ajdq(l,x)) = succ(lg(l))

QUESTION 2 Concaténation

Pour l'opérateur conc(l,l'), démontrer par induction :

- lg(conc(l,l')) = lg(l)+lg(l')
- conc(conc(l,l'), l") = conc(l, conc(l',l"))

QUESTION 3 Inversion d'une liste

Spécifier l'opération rev(l) qui inverse une liste.

Démontrer que
- lg(rev(l)) = lg(l)
- rev(conc(l,l')) = conc(rev(l'), rev(l))
- rev(rev(l)) = l

Note On sera amené à démontrer des théorèmes supplémentaires.