Assistants de preuve

TD 3- Fonctions Récursives

1 Récursion

1.1 Fibonacci

1- Écrire la fonction calculant les éléments de la suite de Fibonacci en suivant sa définition $F(0) = 0$, $F(1) = 1$ et $F(n+2) = F(n) + F(n+1)$. Évaluer $F(4)$.
2- On souhaite écrire un schéma de fonctions récursives suivant les appels récursifs de Fibonacci. Pour cela, on se donne un type paramétré P tel que $P(n)$ est le type de la valeur retournée pour une entrée n. On suppose aussi donné: la valeur en 0, la valeur en 1, et la valeur en $n+2$ étant donné les valeurs en n et $n+1$:

```
Section FibPrinciple.
Variable P : nat -> Type.
Variable P0: P 0.
Variable P1: P 1.
Variable Pr : forall n, P n -> P (S n) -> P (S (S n)).
```

Écrire une fonction `Pfib1` de type $\forall n, P(n)$.
3- On va maintenant écrire un schéma plus efficace, qui ne fait qu'un seul appel récursif. L'idée est de calculer en même temps $F(n)$ et $F(n+1)$. En utilisant les mêmes paramètres que pour la question précédente, écrire une fonction `Pfib2` de type $\forall n, P(n) * P(n+1)$.
4- Fermer la section:

```
End FibPrinciple.
```

Les constantes `Pfib1` et `Pfib2` ont maintenant 4 arguments supplémentaires correspondant à `P`, `P0`, `P1` et `Pr`. Instancier ces 2 schémas de façon à calculer la suite de Fibonacci, et évaluer chacune de ces fonctions en 4.

1.2 Listes

Nous allons considérer des listes d'éléments d'un type A.

```
Require Import List.
Parameter A : Type.
```

1- Écrire une fonction `split` qui coupe une liste en 2 listes de longueurs équilibrées. En prenant comme modèle la suite de Fibonacci, écrire un schéma de recurrence associé à `split`.

2- Prouver que chaque liste retournée par `split` est de longueur inférieure ou égale à liste d'entrée. On prouvera aussi que si la liste de départ est de longueur supérieure à 1, alors les listes retournées sont de longueur strictement inférieure à la liste entrée.
3- Soit `le:A->A->bool` un ordre sur A. Écrire une fonction prenant en argument une liste et un entier n, et qui retourne la liste triée suivant l'algorithme quicksort, en se limitant à la profondeur n. Montrer que cette fonction est indépendante de n à partir d'une certaine borne.

2 Fonctions partielles

```
Variables (t : A) (F : nat -> nat -> A -> A).
```

On souhaite définir la fonction f telle que $f(n,n) = t$ et $f(n,m) = F(n,m,f(n,m-1))$ pour $n < m$. On prouvera le lemme suivant, la preuve résultante devant être dans tous les cas une sous-preuve de celle de $n \leq m$:

```
Lemma le_pred : forall n m, n <= m -> n <> m -> n <= pred m.
```

Écrire la fonction f ayant comme 3ème argument une preuve que n et m appartiennent à son domaine.