AlgDem : Raisonnement par induction

Haskell

En Haskell, on ne peut pas :

- Modifier la valeur des variables
- Modifier les objets en mémoire
- Faire des boucles, on utilise que la récursivité

On n'utilise que des expressions

> Example
>
> Exemples de fonctions en Haskell

```haskell
factoriel :: Int → Int
factoriel 0 = 1
factoriel n = n * factoriel (n-1)
```

```haskell
fibAux :: Int → Int → Int → int
fibAux 0 n1 n2 = n2
fibAux k n1 n2 = fibAux (k-1) (n1 +n2) n1

fibonacci :: Int → Int
fibonacci n = fibAux n 1 0
```

> Note
>
> On peut aussi utiliser des conditionnelles et des gardes en haskell

```haskell
fibonacci :: Int → Int
fibonacci n = if n < 2 then n else fibonacci
    (n-1)
                                + fibonacci
    (n-2)
```

```haskell
fibonacci :: Int → Int
fibonacci n | n < 2 = n
            | otherwise = fibonacci (n-1) +
    fibonacci (n-2)
```

Les tuples

```haskell
(a, b), (a, (True, 3))
```

Spécification d'un programme

Spécifier un programme, c'est donner une explication précise de la façon dont il doit fonctionner (précondition et post condition)

La programmation par contrats est une méthode de programmation basée sur l'explicitation des pré et post conditions pour chaque fonction. Couplée à des outs de démonstration automatique, cette méthode permet de certifier du code automatiquement

La notion de variant

Précondition : relation entre les x_i :

Montrer d'un programme vérifie sa spécification

Lorsqu'une fonciton `f :: a1 -> ... -> a_n -> b` est spéficiée par :

$$\forall x1 \dots \forall x_n.pre(x1, \dots nx_n) \Rightarrow post(x1, \dots, x_n)$$

Le principe de réccurence est le suivant :

$$(P(0) \wedge \forall n.P(n) \Rightarrow P(n+1)) \Rightarrow \forall n.P(n)$$

Le principe d'induction lui, est :

$$(\forall n.(\forall k.k < n \Rightarrow P(k)) \Rightarrow P(n)) \Rightarrow \forall n.P(n)$$

La notion de variant

Un variant est une mesure des données en entrée du programme. Il doit :

- être un entier positif
- Diminuer à chaque appel récursif

On raisonne généralement par induction sur le variant