TD : preuves et terminaisons

1 Terminaison

Exercice 1. On considère le programme suivant

```
fonction Aleatoire(int x, int y, int m):
    Entree : x,y entiers , m > 0
    Sortie : ce que sont devenus x,y
    debut
    tant_que (x ≠ 0 ET y ≠ 0)
        si y>0:
            y ← y − 1;
        sinon si x > 0:
            x ← x − 1;
            y=randint(−m,m);
        sinon :
            x ← −x − 1;
   renvoyer x,y
   fin
```

L'appel `randint(-m,m)` renvoie un nombre aléatoire entre $-m$ et m.

Etudier la terminaison du programme.

La remarque selon laquelle on ne peut pas trouver de suite infinie d'entiers posititfs strictement décroissante est valable pour toute puissance de $\mathbb{N}$ munie de l'ordre lexicographique.

Par exemple, il n'existe pas de suite $((x_n, y_n, z_n))_{n\in\mathbb{N}}$ strictement décroissante à termes dans $\mathbb{N}^3$. On peut donc étendre la notion de variant aux puissances de $\mathbb{N}$ et s'en servir pour établir la terminaison d'algorithmes.

Exercice 2. Un enfant très ordonné entreprend de ranger ses petits trains. Chaque train est formé d'une suite d'éléments, indistinctement wagons ou locomotives. L'enfant suit la procédure suivante :

- Prendre un train non rangé.
  - S'il contient un seul élément le ranger dans la caisse.
  - sinon, le séparer en deux trains plus petits, qui sont remis dans les trains non rangés.
- Recommencer tant qu'il reste des trains à ranger.

L'enfant ne suit aucune stratégie pour choisir le prochain train à traiter, ni pour sélectionner l'endroit où couper un train en deux.

Ce processus va-t-il s'arrêter ?

2 Correction

Exercice 3. On donne le code suivant, dit d'« exponentiation rapide » :

```python
def expoRap(a, n):
    # précondition : n entier >= 0
    assert n >= 0
    p = 1
    b = a
    m = n
    while m > 0:
        if m % 2 == 1:
            p = p * b
        b = b * b
        m = m // 2
    return p
```

Il est basé sur la propriété suivante :

$$a^b = \begin{cases} a^k \times a^k \text{ si } b = 2k \\ a \times a^k \times a^k \text{ si } b = 2k+1 \end{cases}$$

Montrer sa terminaison et sa correction.

Exercice 4. Montrer la terminaison et la correction de la fonction `puissance_rapide` récursive.

```python
def puissance_rapide(x, n):
    if n == 0:
        return 1
    elif n % 2 == 0:
        return puissance_rapide(x * x, n // 2)
    else:
        return x * puissance_rapide(x, n - 1)
```

Exercice 5. On considère la fonction de McCarthy :

```python
def f (n):
   if n>100:
     return n-10
   else:
     return f(f(n+11))
```

1. Montrer que $f(x)$ termine pour tout $x \geq 100$
2. En raisonnant par récurrence descendante, montrer que $f(n)$ termine pour tout $n \leq 100$ et déterminer sa valeur.