TD D'INFORMATIQUE 1
Analyse de programme : terminaison et correction

1 Recherche d'indice

1. Écrire une fonction `indice` prenant en argument une liste `L` et un élément `x`, et renvoyant un indice `i` tel que `L[i]` = `x`, ou `None` si `x` n'apparaît pas dans `L`.
2. Étudier la terminaison et la correction de la fonction précédente.

2 Analyse d'une fonction récursive

On considère la fonction suivante :

```
def FF(a,b):
    assert b > a
    if b == a+1:
        return b
    else :
        c = (a+b)//2
        return FF(a,c) * FF(c,b)
```

1. Calculer `FF(3,8)`. En déduire une spécification pour cette fonction. En particulier, quelle conjecture peut-on proposer pour la valeur renvoyée par `FF(0,n)` ?
2. Démontrer la terminaison et la correction (relativement à la spécification proposée précédemment) de cette fonction.

3 Algorithme d'Euclide étendu

Pour $(a, b) \in \mathbb{N}^2$, le théorème de Bézout énonce que l'équation $au + bv = \texttt{pgcd}(a, b)$, où u et v sont des inconnues dans $\mathbb{Z}$, a au moins une solution (et donc une infinité).

L'algorithme d'Euclide étendu permet de calculer une telle solution, en plus du pgcd :

```
def pgcd_bezout(a,b):
    r0, r1 = a, b
    u0, u1 = 1, 0
    v0, v1 = 0, 1
    while r1 != 0:
        q = r0 // r1
        r0, r1 = r1, r0-q*r1
        u0, u1 = u1, u0-q*u1
        v0, v1 = v1, v0-q*v1
    return (r0, u0, v0)
```

1. Exécuter à la main cette fonction sur l'entrée $(14, 49)$. Préciser à quoi correspond le résultat.
2. Démontrer la terminaison de cette fonction
3. Démontrer la correction de cette fonction. On pourra admettre :

$$\forall x \in \mathbb{N},\ \forall y \in \mathbb{N}^*,\quad \text{pgcd}(x, y) = \text{pgcd}(y, x \bmod y)$$

4 Exponentiation rapide

On rappelle que l'exponentiation rapide est un algorithme permettant de calculer efficacement x^n en utilisant les décompositions $x^{2k} = x^k * x^k$ et $x^{2k+1} = x * x^k * x^k$.

1. Écrire une fonction récursive implémentant cet algorithme.
2. Démontrer la terminaison et la correction de cette fonction.

5 Tri par sélection

Le tri par sélection consiste à chercher le minimum et à le ranger à sa place, puis chercher le deuxième plus petit élément et le ranger à sa place et ainsi de suite. Lors du TP8, nous avons proposé le code suivant pour ce tri :

```
def indice_minimum(L, i):
    imin = i
    for k in range(i+1, len(L)):
        if L[k]<L[imin]:
            imin = k
    return imin

def tri_selection(L):
    n = len(L)
    for i in range(0,n-1):
        imin = indice_minimum(L,i)
        L[i], L[imin] = L[imin], L[i]
```

Étudier la terminaison et la correction de ces deux fonctions.

6 Évaluation de polynômes

Dans cet exercice, on représente un polynôme $a_0 + a_1X + \ldots + a_nX^n$ par la liste `[`$a_0, a_1, \ldots, a_n$`]`

1. Écrire une fonction `evaluer` prenant en argument une liste de flottants `L` et un flottant `x`, et renvoyant la valeur en `x` du polynôme représenté par `L`.
2. Que calcule la fonction suivante ?

   ```
   def horner(L,x):
       r=0
       n=len(L)
       for i in range(n-1,-1,-1):
           r=r*x+L[i]
       return r
   ```

3. Démontrer la correction de cette fonction.