Correction TP4 PCSI Python

Exercice 1

1)Programme :

```
def exponaive(a,n):
    p=1
    for k in range(n):
        p=p*a
    return p
```

`exponaive(3,10)` et `3**10` renvoient la valeur de 3^{10}, c'est-à-dire 59049.

2)Le temps de calcul de `exponaive(3,1000000)` est de 28 secondes (ça dépend du processeur utilisé).

Exercice 2

1)Tableau :

tour	p	a	n	$p \times a^n$
0 (départ)	1	5	43	1×5^{43}
1	5	5^2	21	$5 \times \left(5^2\right)^{21}$
2	5^3	5^4	10	$5^3 \times \left(5^4\right)^{10}$
3	5^3	5^8	5	$5^3 \times \left(5^8\right)^{5}$
4	5^{11}	5^{16}	2	$5^{11} \times \left(5^{16}\right)^{2}$
5	5^{11}	5^{32}	1	$5^{11} \times 5^{32}$
6	5^{43}	5^{64}	0	5^{43}

a)On remarque que $p \times a^n$ garde toujours la valeur de 5^{43}, c'est un invariant de boucle.

b)6 tours de boucle sont effectués.

c)`exporapide(5,43)` et `5**43` renvoient la valeur de 5^{43}.

2)Pour calculer $3^{1000000}$, la fonction `exporapide` met très peu de temps (le module `time` affiche 0,2 seconde), alors que la fonction `exponaive` met 28 secondes.

Exercice 3

Théorème :

Tout entier naturel $n \geq 1$ s'écrit de manière unique comme une somme de puissances de 2 distinctes.

Autrement dit, il existe $j \in \mathbf{N}$ et $r_0, \ldots, r_j \in \{0, 1\}$ avec $r_j = 1$ tels que :

$$n = \sum_{k=0}^{j} r_k \times 2^k \qquad \text{(décomposition binaire)}$$

Preuve de l'existence : partant de $u_0 = n$, les divisions euclidiennes successives par 2 amènent des entiers u_i et des entiers $r_i \in \{0,1\}$ tels que $u_i = 2u_{i+1} + r_i$, c'est-à-dire : $u_i - 2u_{i+1} = r_i$ $(*)$
Tant que $u_i > 0$, on a $u_{i+1} < u_i$. La suite (u_i) est donc strictement décroissante jusqu'à atteindre 0. Soit j l'entier tel que $u_j = 1$ et $u_{j+1} = 0$.

En multipliant membre à membre $(*)$ par 2^i, on a : $2^i u_i - 2^{i+1}u_{i+1} = 2^i r_i$
En sommant l'égalité pour i allant de 0 à j :
$\sum_{i=0}^{j} \left(2^i u_i - 2^{i+1}u_{i+1}\right) = \sum_{i=0}^{j} 2^i r_i$.

Puis par télescopage : $2^0 u_0 - 2^{j+1}u_{j+1} = \sum_{i=0}^{j} 2^i r_i$, c'est-à-dire $n = \sum_{i=0}^{j} 2^i r_i$.

Et on a : $r_j = u_j - 2u_{j+1} = 1 - 2 \times 0 = 1$.

1)

$$\begin{aligned} 37 &= 2 \times 18 + \mathbf{1} \\ 18 &= 2 \times 9 + \mathbf{0} \\ 9 &= 2 \times 4 + \mathbf{1} \\ 4 &= 2 \times 2 + \mathbf{0} \\ 2 &= 2 \times 1 + \mathbf{0} \\ 1 &= 2 \times 0 + \mathbf{1} \end{aligned}$$

Donc $37 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 0 \times 2^4 + 1 \times 2^5$.

$$\begin{aligned} 43 &= 2 \times 21 + \mathbf{1} \\ 21 &= 2 \times 10 + \mathbf{1} \\ 10 &= 2 \times 5 + \mathbf{0} \\ 5 &= 2 \times 2 + \mathbf{1} \\ 2 &= 2 \times 1 + \mathbf{0} \\ 1 &= 2 \times 0 + \mathbf{1} \end{aligned}$$

Donc $43 = 1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 0 \times 2^4 + 1 \times 2^5$.

2)• Méthode alternative :
$2^9 = 512$ et $2^{10} = 1024$ donc $2^9 \leq 1000 < 2^{10}$.
$1000 - 2^9 = 488$ (1)
$2^8 = 256$ et $2^9 = 512$ donc $2^8 \leq 488 < 2^9$.
$488 - 2^8 = 232$ (2)
$2^7 = 128$ et $2^8 = 256$ donc $2^7 \leq 232 < 2^8$.
$232 - 2^7 = 104$ (3)
$2^6 = 64$ et $2^7 = 128$ donc $2^6 \leq 104 < 2^7$.
$104 - 2^6 = 40$ (4)
$2^5 = 32$ et $2^6 = 64$ donc $2^5 \leq 40 < 2^6$.

$40 - 2^5 = 8$ (5)

$2^3 = 8$ et $2^4 = 16$ donc $2^3 \leq 8 < 2^4$.

$8 - 2^3 = 0$ (6)

En ajoutant les égalités (1) à (6), on obtient :

$1000 = 2^3 + 2^5 + 2^6 + 2^7 + 2^8 + 2^9$.

• Méthode des divisions euclidiennes :

$$\begin{aligned} 1000 &= 2 \times 500 + \mathbf{0} \\ 500 &= 2 \times 250 + \mathbf{0} \\ 250 &= 2 \times 125 + \mathbf{0} \\ 125 &= 2 \times 62 + \mathbf{1} \\ 62 &= 2 \times 31 + \mathbf{0} \\ 31 &= 2 \times 15 + \mathbf{1} \\ 15 &= 2 \times 7 + \mathbf{1} \\ 7 &= 2 \times 3 + \mathbf{1} \\ 3 &= 2 \times 1 + \mathbf{1} \\ 1 &= 2 \times 0 + \mathbf{1} \end{aligned}$$

On a donc : $1000 = 0 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 0 \times 2^4 + 1 \times 2^5 + 1 \times 2^6 + 1 \times 2^7 + 1 \times 2^8 + 1 \times 2^9$.

Et on retrouve la même chose qu'avec la méthode des divisions euclidiennes.

3)On applique la formule sur la somme des termes consécutifs d'une suite géométrique : $1 + q + q^2 + ... + q^n = \dfrac{q^{n+1} - 1}{q - 1}$ avec $q = 2$ et $n = 9$.

On obtient : $2^{10} - 1 = 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + \cdots + 1 \times 2^9$.

Exercice 4

1)On applique la méthode des divisions euclidiennes :

$$\begin{aligned} 125 &= 2 \times 62 + \mathbf{1} \\ 62 &= 2 \times 31 + \mathbf{0} \\ 31 &= 2 \times 15 + \mathbf{1} \\ 15 &= 2 \times 7 + \mathbf{1} \\ 7 &= 2 \times 3 + \mathbf{1} \\ 3 &= 2 \times 1 + \mathbf{1} \\ 1 &= 2 \times 0 + \mathbf{1} \end{aligned}$$

Donc $125 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 + 1 \times 2^6$.

L'écriture binaire de 125 est : $(1111101)_2$.

2)a)Programme :

```
def binaire(n):
    if n==0:
        return [0]
    L=[]
    while n>0:
        L=[n%2]+L    # concaténation
        n=n//2
    return L
```

b)`binaire(125)` renvoie la liste [1, 1, 1, 1, 1, 0, 1], c'est conforme avec 1).

3)a)Tableau :

x	u
– – –	0
1	1
1	3
0	6
1	13

La valeur renvoyée par la fonction est 13.

b)Lorsque L représente l'écriture binaire d'un entier, `basedix(L)` renvoie son écriture en base 10.

Exercice 5

1)Supposons que $n = (r_j, r_{j-1}, ..., r_1, r_0)_2$.

On a alors : $n = \sum_{k=0}^{j} r_k \times 2^k = r_0 + \sum_{k=1}^{j} r_k \times 2^k$.

Donc $\frac{n}{2} = \frac{r_0}{2} + \sum_{k=1}^{j} r_k \times 2^{k-1} = \frac{r_0}{2} + \sum_{k=0}^{j-1} r_{k+1} \times 2^k$.

Or, $\frac{r_0}{2}$ vaut 0 ou $1/2$ et $\sum_{k=0}^{j-1} r_{k+1} \times 2^k$ est entier.

Donc $n//2 = \left\lfloor \frac{n}{2} \right\rfloor = \sum_{k=0}^{j-1} r_{k+1} \times 2^k = r_1 \times 2^0 + \cdots + r_{j-1} \times 2^{j-2} + r_j \times 2^{j-1}$.

Ce qui prouve que $n//2 = (r_j, r_{j-1}, ..., r_1)_2$

2)La fonction `bits(n)` renvoie le nombre de bits de n.

3)a)L'écriture binaire de n est : $n = (r_j, r_{j-1}, ..., r_1, r_0)_2$.

Donc $n = \sum_{k=0}^{j} r_k \times 2^k$ avec $r_j = 1$.

$\forall k \in [\![0, j-1]\!],\ r_k \times 2^k \geq 0$ et $r_j \times 2^j = 2^j$ donc $n \geq 2^j$.

De plus, $\forall k \in [\![0, j]\!],\ r_k \leq 1$ donc $\sum_{k=0}^{j} r_k \times 2^k \leq \sum_{k=0}^{j} 2^k$.

Avec $\sum_{k=0}^{j} 2^k = \frac{2^{j+1}-1}{2-1} = 2^{j+1}-1 < 2^{j+1}$. Donc $n < 2^{j+1}$.

Finalement, on a : $2^j \leq n < 2^{j+1}$.

On passe au logarithme :

$\ln\left(2^j\right) \leq \ln(n) < \ln\left(2^{j+1}\right)$

$j\ln(2) \leq \ln(n) < (j+1)\ln(2)$

$j \leq \log_2(n) < j+1$.

Donc $j = \lfloor \log_2(n) \rfloor$.

b)Dans l'écriture binaire $n = (r_j, r_{j-1}, ..., r_1, r_0)_2$, on voit que n possède $j+1$ bits.

Donc le nombre de bits de n vaut : $\lfloor \log_2(n) \rfloor + 1$.

4)a)Pour $n = 13$, la fonction renvoie 4.

b)Pour un entier $n \geq 1$, l'entier `mystere(n)` est tel que $2^{\texttt{mystere(n)}} > n$ et $2^{\texttt{mystere(n)-1}} \leq n$.

Donc `mystere(n)`$= j+1 =$ nombre de bits de n.

Exercice 6

1)Soit $\mathscr{P}(n)$ la proposition : « $M^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}$ ».

On a : $F_2 = F_1 + F_0 = 1+0 = 1$, ce qui donne immédiatement : $M = \begin{pmatrix} F_2 & F_1 \\ F_1 & F_0 \end{pmatrix}$

donc $\mathscr{P}(1)$ est vraie.

Soit $n \geq 1$. Supposons $\mathscr{P}(n)$ vraie. Montrons que $\mathscr{P}(n+1)$ est vraie.

$$\begin{aligned} M^{n+1} &= M^n M \\ &= \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} F_{n+1}+F_n & F_{n+1} \\ F_n + F_{n-1} & F_n \end{pmatrix} \\ &= \begin{pmatrix} F_{n+2} & F_{n+1} \\ F_{n+1} & F_n \end{pmatrix} \end{aligned}$$

Donc $\mathscr{P}(n+1)$ est vraie.

On conclut que $\forall n \geq 1,\ M^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}$.

2)a)$M^2 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$

$M^4 = M^2 M^2 = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}$

$M^8 = M^4 M^4 = \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}\begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix} = \begin{pmatrix} 34 & 21 \\ 21 & 13 \end{pmatrix}$

$M^{12} = M^8 M^4 = \begin{pmatrix} 34 & 21 \\ 21 & 13 \end{pmatrix}\begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix} = \begin{pmatrix} 233 & 144 \\ 144 & 89 \end{pmatrix}$

$M^{13} = M^{12} M = \begin{pmatrix} 233 & 144 \\ 144 & 89 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 377 & 233 \\ 233 & 144 \end{pmatrix}$.

b)On déduit : $F_{13} = 233$.

3)a)Programme :

```
def produit(A,B):
    C=[[0,0],[0,0]]
    for i in range(2):
        for j in range(2):
            C[i][j]=A[i][0]*B[0][j]+A[i][1]*B[1][j]
    return C
```

b)Programme :

```
def expomat(A,n):
    p=[[1,0],[0,1]]
    while n>0:
        if n%2==1:
            p=produit(p,A)
        A=produit(A,A)
        n=n//2
    return p
```

c)`expomat([[1,1],[1,0]],13)` renvoie `[[377, 233], [233, 144]]`, et on retrouve la matrice M^{13} trouvée à la question 1).

d)Programme :

```
def fiborapide(n):
    M=[[1,1],[1,0]]
    p=expomat(M,n)
    return p[1][0]
```

4)a)Programme :

```
for n in range(31):
    a=fibo(n)==fiborapide(n)
    print(a)
```

A l'exécution, on obtient 31 fois la valeur `True`.

b)Le temps d'exécution de `fibo(1000000)` est de 7 secondes environ.
Celui de `fiborapide(1000000)` est de 1 seconde environ.

Exercice 7

1)a)Première suite : $2 = 1 + 1$, $3 = 2 + 1$, $6 = 3 + 3$, $9 = 6 + 3$, $10 = 9 + 1$.
Seconde suite : $2 = 1 + 1$, $4 = 2 + 2$, $8 = 4 + 4$, $10 = 8 + 2$.

Dans les deux cas, la suite est strictement croissante, commence par 1, finit par 10, et chaque terme est la somme de deux termes précédents : ce sont des chaînes d'additions pour 10.

b)• Chaîne $1-2-3-6-9-10$:
$a^2 = a \times a,\ a^3 = a^2 \times a,\ a^6 = a^3 \times a^3,\ a^9 = a^6 \times a^3,\ a^{10} = a^9 \times a.$

Cela fait 5 multiplications.

• Chaîne $1-2-4-8-10$:
$a^2 = a \times a,\ a^4 = a^2 \times a^2,\ a^8 = a^4 \times a^4,\ a^{10} = a^8 \times a^2.$

Cela fait 4 multiplications.

> Remarque
> Une chaîne de $j+1$ termes $(a_0, \ldots, a_j)$ donne un calcul de a^n en j multiplications.

2)a)$2 = 1+1,\ 3 = 2+1,\ 6 = 3+3,\ 12 = 6+6,\ 24 = 12+12,\ 48 = 24+24,$
$51 = 48+3,\ 102 = 51+51,\ 153 = 102+51,\ 255 = 153+102.$

La suite est strictement croissante, commence par 1, finit par 255 et chaque terme est la somme de deux termes précédents : c'est une chaîne d'additions pour 255.

b)On calcule successivement :
$a^2 = a \times a,\ a^3 = a^2 \times a,\ a^6 = a^3 \times a^3,\ a^{12} = a^6 \times a^6,\ a^{24} = a^{12} \times a^{12},$
$a^{48} = a^{24} \times a^{24},\ a^{51} = a^{48} \times a^3,\ a^{102} = a^{51} \times a^{51},\ a^{153} = a^{102} \times a^{51},$
$a^{255} = a^{153} \times a^{102}.$

La chaîne comporte 10 multiplications.

3)a)$255 = 2^8 - 1 = 1 \times 2^0 + 1 \times 2^1 + \cdots + 1 \times 2^7$ (voir exercice 3, question 3).

Donc $255 = (1,1,1,1,1,1,1,1)_2$, avec $j = 7$ et $r_0 = \cdots = r_7 = 1$.

b)L'exponentiation rapide effectue $j+1 = 8$ tours de boucle.

• Multiplications `p=p*a` : une par bit égal à 1, soit 8. La première (`p=1*a`) est une multiplication par 1 : elle n'est pas comptée. Il reste 7 multiplications.

• Carrés `a=a*a` : un par tour, soit 8. Le dernier est inutile : il reste 7 carrés.

Au total : $7 + 7 = 14$ multiplications.

4)La chaîne d'additions de la question 2) nécessite 10 multiplications contre 14 pour l'exponentiation rapide : $10 < 14$.

Donc l'exponentiation rapide n'est pas toujours optimale : pour certains exposants, une chaîne d'additions bien choisie permet de calculer a^n avec moins de multiplications.

En revanche, trouver une chaîne optimale pour un exposant donné est difficile, alors que l'exponentiation rapide s'applique tout de suite à n'importe quel exposant.