EXAMEN : Algorithme et Programmation 3 Université Paris-Dauphine MIDO L2

Exercice 1 *Recopier en la complétant cette fonction python de tri insertion d'un tableau* `A` *d'entiers.*

```
def trins(A):
        for j in range(1,len(A)):
                x=A[j]
                i=j-1
                while


                A[i+1]=x
```

[1 point] While $i \geq 0$ and $A[i] > x$ do $A[i+1] = A[i]$ and $i = i - 1$.

Exercice 2 *Recopier en la complétant cette fonction python pour qu'elle retourne un triplet* $(u, v, d) \in \mathbb{Z}^3$ *tels que* $au + bv = d$*, où* d *est le pgcd de* a, b*.*

```
def B(a,b):
        if b==0:

        else:
                [x,y,d]=B(b,a%b)
```

[1 point] Return $[1, 0, a]$ puis Return $[y, x - (a//b) * y, d]$

Exercice 3

1. *Quelle est la parité du produit de deux entiers pairs ?*
2. *Quelle est la parité du produit de deux entiers impairs ?*
3. *Démontrez que* $\log_7 4 \notin \mathbb{Q}$*.*

[1 point] 1. pair 2. impair
[1 point] 3. Si $7^{p/q} = 4$ pour $p, q \in \mathbb{Z}_+$ non nuls, alors $7^p = 4^q$, impossible car 7^p est impair et $4^q =$ pair.

Exercice 4 *Soit*

```
y=0
def f(x):
        global y
        if x>0:
                f(x//2)
                f(x//2)
                y=y+x
```

1. *Combien vaut*

$$\left\lfloor \frac{\left\lfloor \frac{\left\lfloor \frac{x}{2} \right\rfloor}{2} \right\rfloor}{2} \right\rfloor$$

que l'on obtient par des divisions de x *par 2 et des arrondis successifs ? Démontrez votre réponse.*

2. *Donnez l'expression de* y *en fonction de* x*, après l'appel à* `f(x)`*.*
3. *Donnez la fonction* $f(x)$ *telle que* $y = \Theta(f(x))$*.*

4. Démontrez la réponse précédente.

5. Quelle est la complexité de `f()` *?*

[1 point] 1. Pour n divisions arrondies l'expression vaut $\sum_{i=n}^{p} a_i 2^{i-n}$, où $x = \sum_{i=0}^{p} a_i 2^i$ avec $a_i \in \{0,1\}$ et $p \in \mathbb{Z}_+$.
[1 point] On a $\sum_{i=n}^{p} a_i 2^{i-n} = \left\lfloor \frac{x}{2^n} \right\rfloor$, puisque $\sum_{i=0}^{n-1} a_i 2^{i-n} \leq \sum_{i=1}^{n} 2^{-i} = \frac{2^{-1}-2^{-n-1}}{1-2^{-1}} < 1$.
[1 point] 2. $y = \sum_{i=0}^{p} 2^i \left\lfloor \frac{x}{2^i} \right\rfloor$ avec $p = \lfloor \log_2 x \rfloor$.
[1 point] 3. $y = \Theta(x \log x)$
[1 point] 4. Il suffit de montrer $(p+1)\frac{x}{2} \leq y \leq (p+1)x$.
[1 point] Cela découle de $2^i \left\lfloor \frac{x}{2^i} \right\rfloor = \sum_{j=i}^{p} 2^j a_j \geq \frac{x}{2}$, pour $i \leq p$.
[1 point] 5. Le temps $T(x)$ d'exécution de `f(x)` est $T(x) = \Theta(x)$.
[1 point] Puisqu'il satisfait $T(x) = 2T(x/2) + \Theta(1)$.

Exercice 5 *Soient $n, b, p \geq 2$ des entiers, et soit $T(n)$ une fonction, représentant le temps d'exécution d'un algorithme en fonction de la taille n de son entrée, telle que $T(b^p) = (p+1)b^{cp}$, lorsque n est une puissance de b. Montrez que $T(n) = \Theta(n^c \log n)$.*

[1 point] Il existe un entier p tel que $b^p \leq n \leq b^{p+1}$, d'où $pb^{cp} \leq n^c \log_b n \leq (p+1)b^{pc+c}$, et donc, pour $p \geq 2$, on a:
[1 point] $\frac{1}{b^c} \times n^c \log_b n \leq \frac{p+1}{b^c} b^{cp+c} = (p+1)b^{cp} = T(b^p) \leq T(n) \leq T(b^{p+1}) = (p+2)b^{cp+c} \leq 2pb^c\, b^{cp} \leq 2b^c \times n^c \log_b n$

Exercice 6 *1. Donnez la définition complète de $f(n) = \Theta(g(n))$.*

2. Sous quelle(s) condition(s) sur l'entier b et sur la fonction $f()$, a-t-on $f(n) - b = \Theta(f(n))$?

3. Démontrez la réponse précédente.

[1 point] 1. Il existe deux constantes réelles positives c, d **et une constante entière** n_0 telles que $cg(n) \leq f(n) \leq dg(n)$ **pour tout** $n \geq n_0$.
[1 point] 2. Aucune condition sur $b \in \mathbb{Z}_+$, il suffit que $f()$ soit croissante, et tende vers $+\infty$.
[1 point] 3. On a toujours $f(n) - b \leq f(n)$. Si $f()$ est croissante et tend vers $+\infty$, pour $n \geq n_0$ ($\exists n_0$), on a : $f(n) \geq 2b$, d'où $\frac{1}{2} f(n) \leq f(n) - b$.
Variante plus précise non-obligatoire mais bien sûr acceptée:
[1 point] 2. Il est nécessaire et suffisant qu'il existe $\alpha > 1$ et n_0 tels $f(n) \geq \alpha b$ pour tout $n \geq n_0$.
[1 point] 3. Si $f(n) \geq \alpha b$, alors $(1 - \alpha^{-1}) f(n) \leq f(n) - b$. Si $cf(n) \leq f(n) - b$ pour $c > 0$, alors $f(n) \geq (1-c)^{-1} b$.

Exercice 7 *Considérons un algorithme dont la taille des entrées peut-être représentée par un entier n, et dont le temps d'exécution $T(n)$ satisfait:*

$$T(n) = \begin{cases} \Theta(1) & \text{si } n < b \\ aT(n-b) + \Theta(n^c) & \text{si } n \geq b \end{cases}$$

où $a, b, c \in \mathbb{N}$ et $b \geq 1$.

1. Donnez la fonction $f(n)$ telle que $T(n) = \Theta(f(n))$, à l'aide des paramètres a, b, c.

2. À quelle condition, nécessaire et suffisante, sur les paramètres, l'algorithme n'est pas exponentiel ?

3. En supposant cette condition satisfaite, donnez le polynôme $p(n)$ telle que $T(n) = \Theta(p(n))$.

4. Démontrez la réponse précédente.

[1 point] 1. $f(n) = a^q + \sum_{i=0}^{q-1} a^i (n - ib)^c$ avec $q, r \in \mathbb{Z}_+$ tels que $n = qb + r$ et $r < b$.
[1 point] 2. L'algorithme n'est pas exponentiel si et seulement si $a = 1$ (ou zéro, si envisageable).
[1 point] 3. $p(n) = n^{c+1}$.
[1 point] 4. Avec $a = 1$, $f(n) = \sum_{i=0}^{q-1} (n - ib)^c = \sum_{i=0}^{q-1} (r + (q-i)b)^c = \sum_{i=1}^{q} (r + ib)^c$, d'où:

$$b^c \sum_{i=1}^{q} i^c \leq f(n) \leq 2b^c \sum_{i=1}^{q} i^c$$

[1 point] Le résultat découle de $\sum_{i=1}^{q} i^c = \Theta(q^{c+1})$.

Exercice 8 *Expliquez l'algorithme de Strassen multipliant les matrices carrées $n \times n$ plus rapidement qu'en $\Theta(n^3)$.*

[1 point] L'algorithme de Strassen calcule les 8 sous-produits $A_{ij}B_{k\ell}$ de la décomposition

$$AB = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = C$$

avec seulement 7 appels récursifs. Il s'exécute en un temps $T(n) = \Theta(n^{\log_2 7})$ puisque $T(n) = 7T(n/2) + \Theta(n^2)$. Il améliore l'algorithme en $\Theta(n^3)$ puisque $\log_2 7 < \log_2 8 = 3$.

[1 point] Précisément, on peut, par exemple, calculer $P_1 = (A_{11}+A_{12})B_{22}$, $P_2 = A_{11}(B_{12}-B_{22})$, $P_3 = (A_{21}+A_{22})B_{11}$, et $P_4 = A_{22}(-B_{11}+B_{21})$; on obtient $C_{12} = P_1 + P_2$ et $C_{21} = P_3 + P_4$. On calcule ensuite $P_5 = (A_{11}+A_{22})(B_{11}+B_{22})$, $P_6 = (A_{12}-A_{22})(B_{21}+B_{22})$, et $P_7 = (-A_{11}+A_{21})(B_{11}+B_{21})$; et on obtient $C_{11} = P_6 + P_5 + P_4 - P_1$ et $C_{22} = P_7 + P_5 + P_2 - P_3$.