N° etudiant : **Nom :** **Prénom :**

CT - Sujet A

Consignes : Les réponses doivent tenir sur la feuille (il y a largement assez d'espace). Aucun document n'est autorisé. Toute tentative de fraude sera sanctionnée.

Afin de ne pas perdre de points bêtement, lisez bien le sujet en entier avant de commencer. Écrivez le code proprement (au crayon à papier c'est plus simple d'éviter les ratures), les copies trop sales seront pénalisées. Le barême est indicatif et peut être modifié.

Pour les calculs de complexité, attention aux slices : par exemple `tab[:m]` prend $\mathcal{O}(m)$ opérations.

Exercice 1 (1 point)**.** Donnez le terme dominant des fonctions suivantes en terme de $\mathcal{O}()$, il n'est pas nécessaire de justifier votre réponse :

$f(n) = 12n^{1000}\log n + 1000n^{1001} + 0,003n^3\log^{1000} n$

$g(n) = 1,1n^{1,1} + 10n\log n + 0,5n^2$

$h(n) = 10 * 2^n + n! + 4^{n+3}$

Exercice 2 (4 points)**.** Donner les complexités des fonctions `f1` et `f2`, justifiez vos réponses en annotant le code et/ou en donnant l'équation de récurrence et le théorème utilisé pour la résoudre si nécessaire.

```
def g(n,S):
        s = 0
        for i in range(n):
                s += S[i]**2
        return s

def f1(T):
        n = len(T)
        m = len(T[0])
        s = 0
        for i in range(n):
                        s += g(m, T[i])
        return s
```

```
def f2(n):
        if n == 1:
                return 1
        S = 0
        for i in range(n):
                S+=i
        return 2*f2(n/2) + S
```

Exercice 3 (5 points)**.** Un tableau T est un *chameau* s'il est d'abord strictement croissant puis strictement décroissant (par exemple `[1,2,4,7,4,1,-1]` est un chameau, mais `[1,2,3,1,0,2]`, `[4,4,4]` et `[1,2,3,4]` n'en sont pas). Donnez un algorithme qui prend en entrée un chameau et trouve l'indice j tel que $T[j]$ est la valeur la plus grande du tableau. Analysez la complexité de votre algorithme.

Exercice 4 (5 points)**.** Écrivez un algorithme `plusGrandCarre(T1,T2)` qui étant donné deux tableaux bidimensionnels `T1` et `T2`, de même taille et remplis d'entiers, trouve la taille n du plus grand carré qui est identique dans les deux tableaux : pour toute position `i,j` du carré, `T1[i][j]==T2[i][j]`. Donnez la complexité de votre algorithme.

Exercice 5 (5 points)**.** Écrire un algorithme `nombreSolutions(f,n)` qui prend en entrée :

- une fonction `f` qui renvoie toujours 0 ou 1
- qui prend en argument une liste `l` d'entiers entre 0 et 13,
- un entier n, la longueur de la liste `l`,

et qui calcule le nombre de solutions de l'équation $\texttt{f}([x_1,\ldots,x_n]) = 0$. En supposant que le calcul de $\texttt{f}([x_1,\ldots,x_n])$ se fait en $\mathcal{O}(g(n))$, donnez la complexité de votre algorithme.

N° etudiant : **Nom :** **Prénom :**

Exercice 6 (6 points)**.** Étant donné un tableau quelconque d'entiers T, on cherche dans T la plus grande sous-suite [1] d'éléments qui forme un chameau (cf Exercice 3). Donnez un algorithme qui étant donné un tableau T renvoie la taille du plus grand chameau qu'il contient, donnez sa complexité.

1. Les éléments ne sont donc pas nécessairement contigüs.