Informatique – TP19

Vésale Nicolas - Henrik Thys

Exercice 1: Calcul d'une somme.

On pose, pour tout $n \in \mathbb{N}$:

$$S_n = \sum_{k=1}^{n} k^3.$$

1. Donner une fonction `somme1` qui prend pour paramètre n et qui rend S_n, calculée à l'aide d'une boucle `for`.
2. Donner la complexité de `somme1` en fonction de n.
3. On rappelle que S_n vaut $n^2\,(n+1)^2/4$. En déduire une autre fonction `somme2` qui prend pour paramètre n et qui rend S_n, de complexité $\mathcal{O}(1)$.

Exercice 2: Liste des puissances.

Un étudiant propose les deux fonctions suivantes:

```
def puissance(a:float,k:int)->float:
    '''calcule a^k'''
    p=1
    for i in range(k):
        p=p*a
    return p
```

```
def lstPuissances(a:float,n:int)->[float]:
    '''rend la liste [a^1,a^2,...,a^n]'''
    L=[]
    for i in range(1,n+1):
        L.append(puissance(a,i))
    return L
```

1. Donner la complexité de `puissance` en fonction de k. En déduire celle de `lstPuissances` en fonction de n.
2. Proposer une amélioration de la fonction `lstPuissances` pour que sa complexité devienne $\mathcal{O}(n)$.

Exercice 3: Histogramme des éléments d'une liste d'entiers naturels.

1. Donner une fonction `maxi` qui prend pour paramètre une liste `L` d'entiers naturels et qui rend le plus grand élément de cette liste. Donner sa complexité en fonction de `n=len(L)`.
2. Donner une fonction `histogramme` qui prend pour paramètre une liste d'entiers naturels `L` et qui rend la liste $[\mathtt{h}_0, \mathtt{h}_1, \ldots, \mathtt{h}_\mathtt{m}]$ où m est le maximum de la liste `L` et pour tout i compris entre 0 et m, $\mathtt{h}_\mathtt{i}$ désigne le nombre d'apparitions de l'entier i dans la liste `L`.
3. Selon vous, quels sont les paramètres pertinents pour calculer la complexité de `histogramme` ?
4. Calculer la complexité de `histogramme` en fonction des paramètres choisis à la question précédente.

Exercice 4: Présence d'un écart dans une liste.

On se donne une liste `L` d'entiers et un entier `e`. On cherche à créer une fonction qui rend `(i,j)`, deux indices tels que `L[i]-L[j]=e` si ils existent (on supposera que c'est toujours le cas). Par exemple, si `L=[1,8,7,4,12]` et `e=5` notre fonction doit rendre `(4,2)`.

1. **Méthode naïve:** pour résoudre ce problème, la première idée que l'on peut avoir est d'utiliser deux pointeurs `i` et `j` pour parcourir tous les `L[i]-L[j]` pour tester si l'une de ces quantités vaut `e`.
   Écrire une fonction `ecart1(L,e)` qui utilise cette méthode pour répondre à la question et donner sa complexité en fonction de `n=len(L)`.
2. **Méthode avancée:** une autre idée pour résoudre ce problème est de parcourir une première fois la liste `L` pour créer un dictionnaire `d` dont les clés sont les `e+L[j]` et la valeurs associées `j`. Une fois ceci fait, on parcourt une deuxième fois la liste `L` et si l'élément `L[i]` est une clé du dictionnaire, on peut rendre `(i,d[L[i]])`. **Pourquoi est-ce le cas?**
   Écrire une fonction `ecart2(L,e)` qui utilise cette méthode pour répondre à la question et donner sa complexité en fonction de `n=len(L)`.