Mathématiques supérieures — le 1er Décembre 2017

TP n°4 - Noté

TP Noté - Durée 1h30

Tous les documents sont autorisés.

5 exercices

Exercice 1.

On s'intéresse à la série $\sum_{k\geq 1} \frac{1}{k^2}$.

1. Écrire une fonction **itérative** `serieIt(n)` qui renvoie la somme $\sum_{k=1}^{n} \frac{1}{k^2}$.
2. Écrire une fonction **récursive** `serieRec(n)` qui renvoie la somme $\sum_{k=1}^{n} \frac{1}{k^2}$.

Exercice 2.

Déterminer la complexité temporelle des algorithmes récursifs suivants en terme d'opérations arithmétiques :

```
1 def f1(n):
2     if n == 0:
3         return 1
4     return f1(n-1)*(n**2)
```

```
1 def f2(n):
2     if n == 0:
3         return 0
4     return 2**(f2(n-1))+f2(n-1)
```

Exercice 3.

On considère la fonction `mystere(L1,L2)` suivante, qui prend en argument deux listes `L1` et `L2` :

```python
def mystere(L1,L2):
    if L1 == []:
        return L2
    else:
        s = L1.pop(0)
        if s not in L2:
            L2.append(s)
        return mystere(L1,L2)
```

On rappelle que `L.pop(i)` supprime l'élément d'indice `i` de la liste `L` et renvoie la valeur de `L[i]`.

1. Que renvoie la fonction `mystere(L,[])` pour `L=[1,2,3,3,1,5,8,1,4,9,9,8,9]`.
   En déduire le fonctionnement général de la fonction `mystere`
2. Déterminer la complexité temporelle de la fonction `mystere` dans le pire des cas en termes d'opérations `pop()` et `append()`.

Exercice 4. *Tri à bulles*

Le but de cet exercice est de coder l'algorithme de **tri à bulles** qui permet de trier une liste de nombres. On considère une liste `L` de taille `n=len(L)`. Le principe du tri à bulle est le suivant :

> i) On parcourt la liste `L` de gauche à droite avec `i` allant de `0` à `n` et on échange `L[i]` et `L[i+1]` si ces deux éléments sont mal ordonnés i.e. si `L[i]>L[i+1]`.
>
> ii) Une fois le premier parcourt terminé, on recommence le parcourt de la liste du début et on recommence les échanges. Et ainsi de suite, on recommence à parcourir/échanger tant qu'il reste des échanges à effectuer.
>
> iii) On renvoie la liste : elle est triée !

En annexe, on peut voir le déroulement de l'algorithme de tri à bulles sur la liste `L=[1,6,4,2,5,3]`

1. Expliquer pourquoi le tri à bulles renvoie bien une liste triée.
2. a. Écrire une fonction `parcourt(L)` qui parcourt la liste `L` de gauche à droite et qui effectue les échanges comme indiqué au point i). Cette fonction renvoie `False` si elle a effectué des échanges lors du parcourt et `True` sinon.
   b. Écrire une fonction `triabulles(L)` qui effectue l'algorithme de tri à bulles - on se servira bien sûr de la fonction `parcourt(L)`.
3. Quel est la complexité de cet algorithme de tri dans le pire des cas en terme d'échanges ?

Exercice 5.

On dispose de briques de longueur 2 et 3, et on veut connaître le nombre de façons de construire une rangée de longueur n. Par exemple, voici deux façons de construire une rangée de longueur 13. Il en existe 16 au total :

Écrire un algorithme permettant de calculer ce nombre de façons pour une longueur totale donnée. (On pourra étudier le nombre de possibilités de construire des rangées de longueur $n-2$ et $n-3$).