Le petit point de cours, informatique (1)

1. Proposer deux fonctions, l'une itérative et l'autre récursive permettant de calculer $S_n = \sum_{k=1}^{n} \frac{1}{k^2}$.

   • Fonction récursive.

```
def somme_rec(n):
        if n==0 :
                return 0
        else :
                return somme_rec(n-1)+1/(1+n**2)
```

   • Fonction itérative.

```
def somme_it(n):
        s=0
        for k in range(1,n+1) :
                s=s+1/(1+k**2)
        return s
```

2. On souhaite approcher numériquement l'intégrale $I = \int_0^1 f(t)\,\mathrm{d}t$ où $f(t) = \frac{1}{1+t^2}$ pour $t \in [0,1]$ par la méthode des rectangles point milieu.

   a) Écrire une fonction en Python donnant les valeurs de $f(x)$ pour x réel.

   Voici un script possible.

```
def f(x) : return 1/(1+x**2)
```

   b) Écrire une fonction Python donnant une valeur approchée de I en fonction du nombre n de rectangles utilisés.

   • Je donne une fonction dont les variables sont la fonction que l'on intègre (f), les bornes de l'intervalle d'intégration (a et b) ainsi que le nombre de rectangles utilisé (n). Je rappelle que la valeur approchée de l'intégrale est donnée par : $J = h\sum_{i=0}^{n-1} f(a+(i+1/2)h)$ où h est le pas $h = \frac{b-a}{n}$ puisque l'aire de chaque rectangle est donnée par :

$$h \times f\left(\frac{a+ih+a+(i+1)h}{2}\right).$$

```
def rectmilieu(f,n,a,b):
        h=(b-a)/n
        s=0
        for i in range(0,n):
                s=s+f(a+(i+0.5)h)
        return h*s
```

• Une fonction alternative, avec des listes, est la suivante :

```
def rectmilieu2(f,a,b,n):
        h=(b-a)/n
        L=[]
        for i in range(n):
                L.append(f(a+(i+1/2)*h))
        return h*math.fsum(L)
                # fsum(L) calcul la somme des éléments de la liste L.
```

c) On admet que, pour cette fonction, en notant J_n la valeur approchée fournie par la méthode des rectangles point milieu avec n rectangles, on a :

$$|I - J_n| \leqslant \frac{1}{3n^2}.$$

Faire une fonction Pyhon donnant une valeur approchée de I à 10^{-5} près.

• Je donne une fonction Python qui va calculer en fonction de ε une valeur approchée à ε près de l'intégrale $\displaystyle\int_0^1 \frac{1}{1+t^2}\,\mathrm{d}t$ à l'aide de la méthode des rectangles point milieu.
Noter que l'on a les équivalences :

$$\frac{1}{3n^2} \leqslant \varepsilon \Leftrightarrow n^2 \geqslant \frac{1}{3\varepsilon} \Leftrightarrow n \geqslant \frac{1}{\sqrt{3\varepsilon}}.$$

```
import math
def f(x):
        return 1/(1+x**2)

def valap(eps):
        n=math.floor(1/math.sqrt(3*eps))+1
                # floor calcul la partie entière.
                # On augmente d'un pour avoir un entier supérieur...
        h=1/n
        L=[]
        for i in range(n):
                L.append(f(a+(i+1/2)*h))
        return h*math.fsum(L)
```

3. a) On considère la fonction suivante

```
def g(n,a,b):
        if n==0 :
                return a
        elif n==1 :
                return b
        else :
                return 2*g(n-1,a,b)-g(n-2,a,b)
```

Que produit l'appel `g(10,1,3)` ?

L'appel `g(10,1,3)` va donner le terme u_{10} de la suite définie par :

$$\begin{cases} u_0 = 1,\ u_1 = 3 \\ u_n = 2u_{n-1} - u_{n-2} \text{ pour tout } n \geqslant 2 \end{cases}$$

C'est la ligne `return 2*g(n-1,a,b)-g(n-2,a,b)` qui donne la relation de récurrence.

b) On considère la fonction

```
def f(n,a,b):
        L=[a,b]
        for i in range(2,n):
                L.append(2*L[i-1]-L[i-2])
        return L[n-1]
```

Que produit l'appel `f(10,1,3)` ?

L'appel `f(10,1,3)`, un peu comme à la question précédente, va donner le terme u_9 de la suite définie par :

$$\begin{cases} u_0 = 1,\ u_1 = 3 \\ u_n = 2u_{n-1} - u_{n-2} \text{ pour tout } n \geqslant 2 \end{cases}$$

C'est la ligne `L.append(2*L[i-1]-L[i-2])` qui donne la relation de récurrence : à la case i de la liste `L` on place le nombre `2*L[i-1]-L[i-2]`.

c) Évaluer le nombre $N(n)$ d'opération élémentaires (calculs et tests) nécessaires à la fonction f de la question 3b pour donner un résultat.

• On fait deux calculs dans le bloc `L.append(2*L[i-1]-L[i-2])` et ce bloc est effectué $n-2$ fois : $N(n) = O(n)$.

• C'est la même chose si on prend en compte en plus les affectations (2 au début et une par bloc de la boucle for).

d) On appelle $C(n)$ le nombre d'opération élémentaires (calculs et tests) nécessaires à la fonction g de la question 3a pour donner un résultat.

i) Déterminer $C(0)$ et $C(1)$.

• On a $C(0) = 1$: on effectue juste un test puisque $n = 0$.

• On a $C(0) = 2$: on effectue ici deux tests, puisque $n = 1$.

ii) Justifier que pour $n \geqslant 2$ on a : que $C(n) = C(n-1) + C(n-2) + 2$.

Si $n \geqslant 2$ entier, l'instruction `return 2*g(n-1,a,b)-g(n-2,a,b)` comporte 2 opérations et des appels récursifs. Le nombre d'opérations est donc :

$$C(n) = C(n-1) + C(n-2) + 2.$$

NB. Honte au concepteur (c'est à dire moi), car si on prend en compte les deux tests, il vient :

$$C(n) = C(n-1) + C(n-2) + 4.$$

iii) Pour $n \in \mathbb{N}$ on pose $u_n = C(n) + 2$. Trouver une relation pour $n \geqslant 2$ entier entre u_n, u_{n-1} et u_{n-2}. En déduire $C(n)$.

• Pour $n \geqslant 2$ entier on a : $u_n = C(n) + 2 = C(n-1) + 2 + C(n-2) + 2 = u_{n_1} + u_{n-2}$.

• L'équation résolvante de cette suite récurrente linéaire d'ordre 2 est :

$$r^2 - r - 1 = 0.$$

On trouve de suite que cette équation admet deux solution distinctes qui sont $r_1 = \dfrac{1+\sqrt{5}}{2}$ et $r_2 = \dfrac{1-\sqrt{5}}{2}$.
Il en résulte qu'il existe deux constantes α et β telles que pour tout $n \in \mathbb{N}$ on ait :

$$u_n = \alpha r_1^n + \beta r_2^n.$$

Mais $u_0 = C(0) + 2 = 3$ et $u_1 = 4$. Il vient donc :

$$(S) \left\{ \begin{array}{l|l|l} \alpha + \beta = 3 & r_2 & r_1 \\ \alpha r_1 + \beta r_2 = 4 & -1 & -1 \end{array} \right.$$

Cela signifie que l'on fait $L_1 \leftarrow r_2 L_1 - L_2$ et $L_2 \leftarrow r_1 L_1 - L_2$)

$$(S) \Leftrightarrow \begin{cases} \alpha(r_2 - r_1) = 3r_2 - 4 \\ \beta(r_1 - r_2) = 3r_1 - 4 \end{cases} \Leftrightarrow \begin{cases} \alpha = \dfrac{4 - 3r_2}{\sqrt{5}} = \dfrac{1}{2\sqrt{5}}(5 + 3\sqrt{5}) \\ \beta = \dfrac{3r_1 - 4}{\sqrt{5}} = \dfrac{1}{2\sqrt{5}}(-5 + 3\sqrt{5}) \end{cases}$$

Enfin, pour tout $n \in \mathbb{N}$ on a :

$$C(n) = u_n - 2 = \alpha r_1^n + \beta r_2^n - 2.$$