Planche de TD numéro I

Exercice 1

1. Démontrez que la série $\sum_{n=1}^{+\infty} \frac{1}{4^n}$ est convergente, et calculez sa somme.
2. Quelle est l'écriture en base 2 de $1/4$? De $1/4^n$? Et enfin de $4/3$?

Exercice 2 : Sur la représentation des réels en machine

1. Que pensez vous de la suite (u_n) définie par $u_0 = \frac{4}{3}$ et, pour $n \geq 1$, $u_n = 4(u_{n-1} - 1)$?
2. Vérifiez en définissant la procédure suivante :

```
def affiche(N):
    u = 4/3
    for n in range(N):
        print ‘n = ‘, n, ‘ u = ‘,u
        u = 4*(u-1)
```

   puis en exécutant `affiche(30)`.
3. Dans la définition de affiche, remplacez la deuxime ligne $u = 4/3$ par $v = 4/3.$, et dans les lignes suivantes remplacez u par v. La variable v prendra ainsi des valeurs réelles, et non plus fractionnaires. Relancez l'exécution `affiche(30)`. Sauriez vous expliquer ce qui s'est passé ? Vérifiez au passage que $v_{27} = 0$.
4. On imagine que ce résultat surprenant provient de l'impossiblité d'écrire $4/3$ en base 2 avec un nombre fini de chiffres. On suppose donc que la valeur initiale de v mémorisée par la machine n'est pas exactement $\frac{4}{3}$ mais $v = \frac{4}{3} - \varepsilon$ où ε est un petit réel inconnu. *Faisons en outre l'hypothèse que les calculs des 30 premires valeurs de v ont été éffectués exactement.*

   Les valeurs successives de v sont alors les termes de la suite récurrente définie par
   $$v_0 = \frac{4}{3} - \varepsilon, \quad v_{n+1} = 4(v_n - 1).$$
   Pour $n \geq 0$, on définit ε_n par $v_n = \frac{4}{3} - \varepsilon_n$. Exprimez simplement ε_n en fonction de n et de ε.
5. Ayant observé que $v_{27} = 0$, en déduire la valeur de ε.

Exercice 3 : Sur la représentation des réels positifs en machine (suite).

1. Soit un x réel, $x > 0$. Démontrez qu'il existe un unique couple (e, m) où e est un entier relatif, et m un réel satisfaisant $1 \leq m < 2$. Le nombre e est appelé l'*exposant* de x, et m la *mantisse* de x.
   $$x = 2^e m.$$

2. Explicitez la valeur du couple (e, m) pour $x = 1/3$? Pour $x = 4/3$?
3. L'écriture binaire de la mantisse m est en général infinie. Il n'est pas possible de la mémoriser exactement en machine. On se donne un entier p, la *précision*, et on ne conserve en machine qu'une valeur approchée de m n'utilisant que p bits. L'arrondi peut se faire par *défaut*, par *excès* ou au *plus proche*.

   L'arrondi par défaut $\underline{m}$ est obtenu en ne conservant dans l'écriture binaire de m que p chiffres en tout, ou encore $p - 1$ chiffres derrière la virgule.

   L'arrondi $\overline{m}$ est le plus petit nombre $\geq m$ dont l'écriture binaire utilise p chiffres.

   Enfin, l'arrondi au plus proche $\underline{\overline{m}}$ est obtenu en choisissant parmi les 2 valeurs $\overline{m}$ et $\underline{m}$ celle qui est la plus proche de m.

   On utilise une précision de 9 bits. Démontrer que les arrondis de la mantisse de $x = 4/3$ sont

   $$\underline{m} = 1,01010101, \quad \overline{m} = 1,01010110, \quad \underline{\overline{m}} = 1,01010101.$$

4. En `Sage`, sauf mention explicite d'une autre précision, et d'un autre mode d'arrondi, une valeur réelle est mémorisée avec une précision de 53 bits, *avec arrondi au plus proche*. Retrouvez ainsi la valeur du nombre ε calculé dans l'exercice précédent

   $$\varepsilon = 0{,}0000000000000000000000000000000000000000000000000000010101010101\ldots$$

   où le nombre de 0 derrière la virgule, avant le premier 1, est 53.

Remarque : Si l'on souhaite travailler avec des nombres réels avec une précision autre que la précision 53, par exemple avec 100 bits, on pourra écrire

```
R100 = RealField(100)
x = R100(4/3)
```

On peut aussi imposer un autre mode d'arrondi ; pour plus de détails sur les options de `RealField`, frapper `RealField ?`

Exercice 4 : Convergence ou pas convergence

On s'intéresse aux séries de terme général a_n où a_n est donné par

1. $a_n = \dfrac{12}{(-5)^n}$,	2. $a_n = \dfrac{2n^2 - 1}{n^2 + 1}$,
3. $a_n = \tan(n)$,	4. $a_n = (0.6)^{n-1}$,
5. $a_n = \dfrac{1}{n^{1,5}} - \dfrac{1}{(n+1)^{1,5}}$,	6. $a_n = \dfrac{1}{n(n-1)}$, $n \geq 2$.

Pour chacune de ces séries :

1. En utilisant `Sage`, calculez et affichez la suite $(a_1, a_2, \ldots, a_{10})$ et la suite $(S_1, S_2, \ldots, S_{10})$ des 10 premières sommes partielles de la série.
2. Répondez à la question : la série semble t-elle convergente ?
3. Si votre réponse est oui, démontrez la convergence et calculez la valeur de la somme.

Exercicde 5

On considère la série de terme général $u_n = \dfrac{1}{n}$.

1. Calculez et affichez les sommes partielles $S_n = \sum_{k=1}^{n} \dfrac{1}{n}$ pour $n = 10, 20, 30, 40, \dots 100$.
   Cela vous amène t'il à penser que la série est convergente ? ou divergente ?
2. Calculez et affichez la suite
   $$S_1, S_2, S_4, S_8, S_{16}, \dots, S_{2^{10}}.$$
   Quelle est votre impression ?

Exercice 6 : Convergence - une interprétation géométrique

1. Prouvez que la série de terme général $a_n = \dfrac{1}{n(n+1)}$ est convergente et calculez la valeur de sa somme.
2. A l'aide de `Sage`, pour $n = 0, 1, 2, 3, 4, ..., 20$, tracez les courbes d'équations : $y = x^n$, $x \in [0, 1]$.
3. Donner une démonstration géométrique du résultat obtenu à la question 1.

Exercice 7 : Somme d'une série lentement convergente

1. Calculez une primitive de $x \mapsto \dfrac{1}{x \ln^2 x}$. D'abord à la main, puis à l'aide de la commande `integrate`.
2. Pour $n \geq 2$ on note $u_n = \dfrac{1}{n \ln^2 n}$, et $S_n = \sum_{k=2}^{k=n} u_k$. En comparant cette somme à une intégrale, donnez un majorant de S_n. Prouvez que la série $\sum_{n=2}^{+\infty} u_n$ est convergente.
   On note $S = \sum_{n=2}^{+\infty} u_n$ la somme de cette série.
3. Soit $R_n = \sum_{k=n+1}^{k=\infty} u_k$ le reste d'ordre n. Démontrez que $\dfrac{1}{\ln n + 1} < R_n < \dfrac{1}{\ln n}$.
4. Montrez que si $S - S_n < 0.01$ alors $n \geq 10^{43}$.
5. Ouvrez une feuille de calcul `Sage`, calculez S_{20} et un encadrement de R_{20} puis une valeur approchée de S à 3×10^{-3} près.
6. En complètant la cellule :
   ```
   def SumPart(n):
       return sum(...)
   ```
   écrivez la fonction `Sp(n)` qui rend la somme partielle d'ordre n, $S_n = \sum_{k=2}^{n} u_k$.
7. En complètant judicieusement la troisième ligne de la cellule :

```
def GoodApprox(n) :
    Sum = SumPart(n)
    Sum = Sum + ...
    Delta = 0.5*(1./log(n)-1./log(n+1))
    return [Sum, Delta]
```

écrivez la fonction `GoodApprox(n)` qui renvoie le couple `[Somme,Delta]`, et expliquez pourquoi `Sum` est une valeur approchée de S avec une erreur inférieure à `Delta`.

8. Calculez une valeur approchée à 10^{-6} près de $S = \sum_{n=2}^{+\infty} u_n$.