ISM MPSI/PCSI, Informatique
Année 2023/2024

TP Informatique 16

⚠ On rappelle qu'un script (fichier `*.py`) doit être enregistré et exécuté (touche `F5`) pour que les fonctions saisies dans le script soient utilisables dans la console et que la combinaison de touches `Ctrl+C` permet de casser une boucle infinie.

Exercice 1

1. Effectuer les saisies :

```
>>> 2.**1024
>>> 2**1024
```

2. Commenter les résultats observés. On pourra s'intéresser au type de ce(s) résultat(s).

Exercice 2

1. Saisir les instructions

```
>>> 1+2**(-10)-1==2**(-10)
>>> 1+2**(-100)-1==2**(-100)
```

2. Quel phénomène cette expérimentation illustre-t-elle ?
3. Saisir la fonction suivante à compléter qui détecte le seuil de ce phénomène.

```
def seuil():
    n=1
    while *** :
        n+=1
    return n-1
```

4. Question facultative (après avoir traité toutes autres questions du TP) :
écrire une version dichotomique de la détection de ce seuil.

Exercice 3

1. Tester les instructions suivantes : `.1+.2`, `.3+.6`, `.8+.9`.
Qu'observe-t-on ? Commenter ces résultats.
2. Les instructions suivantes affichent les 20 premières décimales des nombres $\frac{k}{\mathtt{n}}$ pour $k \in [\![1\,;\mathtt{n}]\!]$ avec $n = 10$.

```
n=10
for x in range(1,n+1):
    print(format(x/n,".20f"))
```

Quels sont les nombres dont la saisie est fidèle au codage ?
3. Même question avec $\mathtt{n} = 100$.

Exercice 4

On considère les polynômes à coefficients réels suivants :

$$\mathrm{P} = (\mathrm{X} - 0.2)^2 \quad \text{et} \quad \mathrm{Q} = (\mathrm{X} - 0.5)^2$$

1. Déterminer sur feuille des expressions développées de ces deux polynômes en précisant les valeurs numériques des coefficients.
2. Écrire une fonction `discriminant(a,b,c)` qui teste la nullité du discriminant du polynôme $a\mathrm{X}^2 + b\mathrm{X} + c$ avec a, b, c réels et $a \neq 0$.
3. Tester `discriminant` avec les coefficients des polynômes P et Q. Expliquer la différence de traitement entre les deux.
4. Proposer un nouveau test pour la nullité du discriminant.

Exercice 5

1. Saisir les lignes suivantes. La fonction `taux_acc` calcule le taux d'accroissement $\Delta(h) = \frac{(1+h)-1}{h}$ où $\frac{1}{h}$ parcourt la liste des valeurs $\{a_1, \ldots, a_{\mathtt{n}}\}$ avec $a_k = \mathtt{deb} + k\frac{\mathtt{fin}-\mathtt{deb}}{\mathtt{n}}$. On trace ensuite l'erreur relative (en pourcentage) à savoir $\mathrm{R}(h) = |\Delta(h) - 1|$ en fonction de $\frac{1}{h}$.

```
import numpy as np, matplotlib.pyplot as plt

def taux_acc(deb,fin,n):
    tab=np.linspace(deb,fin,n)
    res=[]
    for x in tab:
        taux=((1+1/x)-1)*x
        res.append(abs(taux-1))
    return tab,res

x,y=taux_acc(10,2**55,100)
plt.plot(x,y,'bo');plt.grid();plt.show()
```

2. Affiner l'étude là où cela semble pertinent, en ajustant le choix des paramètres `deb`, `fin` qui déterminent respectivement le début et la fin de la plage des abscisses du tracé.
3. Commenter les résultats observés.

Exercice 6

1. Écrire une fonction `mant(x,n)` qui renvoie les n premiers chiffres de la mantisse d'un réel $x \in [\,1\,;2\,[$. On pourra utiliser la fonction `bin` qui renvoie l'écriture binaire d'un entier.
2. Expérimenter cette fonction avec $n = 50$ sur les nombres suivants : 1, 1.5, 1.25, 1.125, 1.1, 1.9.
3. Commenter les résultats observés.
4. Que se passe-t-il si on réalise la même expérimentation avec $n = 100$? Expliquer.