Exercices section 6.

Manipulations de tests SI

Cette section nécessite, pour ses programmes, l'importation de la bibliothèque `math`.

1. Voici un programme :

```
def testsi(x):
        if x < 0 :
                p=-x
        else :
                p=x
        return p
m=float(input("donner un réel"))
print(testsi(m))
```

   Effectuer quelques tests numériques que vous relèverez puis identifier l'objet mathématique que ce programme cherche à calculer.
   A quelle fonction Python peut-on racccrocher ce dernier ?

2. Voici un programme Python :

```
x=float(input("donner un réel"))
if x==floor(x) :
        print(".......")
else :
        print(".......")
```

   Compléter les ".......” par un texte adapté à la situation en précisant vos choix.
   Que chercher à tester ce programme ?

3. Programmer une fonction Python qui prend en entrée un nombre n et calcule $(-1,02)^n$ si $n \in \mathbb{Z}$ et renvoie -1 sinon.
   Tester votre programme.

Des fonctions avec SI

Le but de section est l'obtention de tracés de fonctions définies avec des SI, type de fonctions étudié dans le cours $\mathcal{A}_5$: *continuité*.
On définit la fonction g sur $\mathbb{R}$ par : $g : x \mapsto \begin{cases} 3x-1 & \text{si } x \leq 3 \\ x^2-3x+1 & \text{si } x > 3 \end{cases}$

1. Proposez un programme Python permettant le calcul des images de la fonction g.
2. A l'aide de votre programme, complétez le tableau :

x	-5	-3.5	$-\sqrt{2}$	-0.2	0	0.2	$\sqrt{3}$	2	2.25	3	5
$g(x)$											

3. Calculez ensuite une liste d'images sur $[-5; 5]$ puis conjecturez les variations d'après ces résultats.
4. On examine maintenant le comportement de g à l'approche de 3. Complétez le tableau suivant :

x	2.5	2.6	2.7	2.8	2.9	3	3.1	3.2	3.3	3.4	3.5
$g(x)$											

   Que pourrait-on dire de g en $x = 3$ d'après ces résultats ? *(vous pourrez calculer d'autres images si besoin)*

Exercices section 7.

Dans cette section, il est demandé de programmer des itérations de calcul de termes de suites à l'aide de boucles `for`. L'usage des bibliothèques `math` ou `numpy` est alors requis.

Suites récurrentes

1. *Cas des suites arithmético-géométriques*
   (a) Calculer le 12[ième] terme de la suite récurrente $(u_n)_{n\in\mathbb{N}}$ arithmético-géométrique de premier terme $u_0 = 5$ et vérifiant $u_{n+1} = 2u_n - 3$ à l'aide d'une boucle `for` en Python.
   (b) Créer une fonction qui prend n en entrée et revoie le terme u_n de la suite précédente.
   Créer alors la liste des termes de u_{20} à u_{25} au moyen de votre programme.
   (c) Comparer les résultats avec les valeurs obtenues directement à partir de la formule générale de calcul du n^{ime} terme d'une suite arithmético-géométrique.
   (d) Généraliser la démarche à toute suite arithmético-géométrique : les paramètres d'entrée de votre fonction Python seront alors n, q, r et u. L'usage d'une boucle `for` est attendu.
   Utiliser ainsi votre procédure pour générer la liste des 15 premiers termes de la suite définie par :
$$u_0 = -1 \quad \text{et} \quad u_{n+1} = 1,32u_n + 1$$
   Comparer avec les résultats obtenus directement à partir de la formule générale de calcul du n^{ime} terme d'une suite arithmético-géométrique.
2. *Avec une situation de récurrence plus générale :*
   (a) On se donne une suite récurrente $(v_n)_{n\in\mathbb{N}}$ définie par $v_0 = a \in \mathbb{R}$ et $v_{n+1} = f(v_n)$ où :
$$f(x) = \frac{2x-1}{1+x^2}$$
   Ecrire un script Python qui prend a et n en entrées et renvoie le $n^{\text{ième}}$ terme de cette suite.
   On pourra programmer la fonction f dans un bloc à part.
   (b) Générer une liste des 10 premiers termes de la suite $(v_n)_{n\in\mathbb{N}}$ pour : $a = 0$ puis $a = 2$ et enfin $a = -3$.
   (c) Pour chacun des trois cas précédents, conjecturer le sens de variations de la suite $(v_n)_{n\in\mathbb{N}}$.

Calculs de Sommations

L'objectif de ces exercices est de comprendre le lien étroit entre `for` et l'usage de Σ.

1. A partir de l'exemple de l'aide-mémoire, organiser le calcul de $S = \sum_{k=1}^{2021} k$ puis vérifier le résultat avec la formule adéquate du cours.
2. Reproduire la méthode pour obtenir un résultat des sommes $T = \sum_{k=1}^{202} k^2$ et $V = \sum_{k=4}^{20} (2k-1)^2$
   Comparer les résultats obtenus la formule obtenue en TD n°5 exercice $\boxed{6}$: $\sum_{k=0}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$
3. Pour chacune des sommes suivantes, programmer une fonction Python utilisant une boucle `for` permettant le calcul, n étant donné en entrée, de :
$$1)\ B_n = \sum_{k=0}^{n} \left(2(-1)^k + 1\right) \quad 2)\ H_n = \sum_{k=1}^{n} \frac{1}{k} \quad 3)\ S_n = \sum_{k=0}^{n} |15-k|$$
   puis tester vos fonctions sur chacune de ces sommes pour $n = 21$ et indiquer les résultats.

4. Générer la liste des valeurs $[B_0\ ;\ B_1\ ;\ \dots\ ;\ B_{20}]$.
   Quelle conjecture peut-on effectuer sur le comportement de la suite $(B_n)_{n\in\mathbb{N}}$?
5. Programmer la fonction `factorielle` qui prend n en entrée et renvoie la valeur de $n! = n \times (n-1) \times \cdots \times 2 \times 1$ avec convention $0! = 1$ vue en TD. On pourra compléter le bloc :

```
def facto(n) :
        if n==0 :
                ........................
        else
                ........................
```

6. Sommations par commande directe : Au moyen de la commande `import numpy as np`, on peut utiliser directement ces instructions Python pour réaliser des calculs de sommes :
   — On peut utiliser `np.sum({ndarray})` pour le calcul de la somme des éléments listés dans un objet de type `ndarray` : Effectuer les calculs des valeurs A, T et V au moyen des commandes `np.array` et `np.sum`
   — Tester la commande `np.sum({ndarray})` sur les éléments listés dans un objet de type `ndarray`. On pourra exploiter les listes déclarées en question précédente.
   Donnez vos interprétations sur l'effet de cette commande à partir des observations.

Tant que vous pouvez encore

A partir de l'exemple de l'aide-mémoire (et de SciLab) résoudre les problèmes suivants :

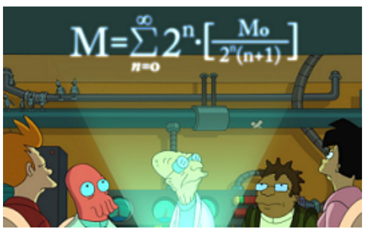
1. *Howard Philip J. Frye a été cryogénisé par accident le soir du réveillon de l'an 2000. L'appareil qui le retient prisionner ne le libérera que dans 1000 années. Au moment de l'incident, son compte bancaire disposait de 5 centimes et le taux de rémunération était de 2,75%.*

   Les intérêts étant calculés chaque 1er Janvier et le montant du capital étant arrondi au cent supérieur de façon systématique, en quelle année notre héros deviendra-t-il milliardaire (même si congelé) ?

2. *Deux frères jumeaux Ki et Hi récupèrent 10 000 euros chacun d'une succession. Le premier, Ki, décide de déposer son héritage sur un compte rémunéré 3% (composés) annuel sans condition. Le second, Hi, préfère faire travailler cet argent par leviers financiers (risqués) et en tire chaque année 400 euros (constants fixes).*

   Etablir le nombre d'années nécessaires à Ki pour montrer à Hi que la patience lui aura été plus bénéfique.

3. La suite H_n définie par $H_n = \displaystyle\sum_{k=1}^{n} \frac{1}{k}$ dite *harmonique* semble croître <u>très</u> lentement.
   Déterminer le rang du premier terme à partir duquel H_n dépasse 10. La suite $(H_n)_{n\in\mathbb{N}}$ vous semble-t-elle majorée ?
4. Dans la série animée *Futurama*, une formule de calcul (*sauvage*) apparaît :



   (a) Frye ne comprend pas très bien les mathémtiques et lit 8 au lieu de ∞. Sachant qu'on lui a expiqué que $M_0 = 0,6$, quelle valeur va-t-il trouver ?
   (b) Simplifier la formule de Calcul mathématiquement dans le cas $M_0 = 1$. Quelle suite reconnaît-on ?
   (c) Programmer une fonction Python qui prend M_0 un `float` et n un `int` et renvoie la valeur de $M_n = \displaystyle\sum_{k=0}^{n} 2^n \frac{M_0}{2^n(n+1)}$.
   Effectuer quelques tests de valeurs que vous reporterez.
   (d) Pour $M_0 = 0,6$, à partir de quelle valeur de $n \in \mathbb{N}$ a-t-on $M_n \geq 100$? (Soyez patient...)