Retour sur quelques programmes fondamentaux

Depuis le début de l'année, nous avons construit nos premiers programmes dans le langage Python. D'ailleurs, que ce soit dans la manipulation des fonctions usuelles, ou celles des suites numériques, ces programmes sont fondamentaux et ils devront être maitrisés pour pouvoir aller plus loin.

Définir une fonction mathématique en tenant compte du domaine de définition

Exercice 1 On considère la fonction f définie sur $]0,1[\cup]1,+\infty[$ par $f(x) = \dfrac{\ln(x)}{(x-1)\sqrt{x}}$.

1. Dans le langage Python, construire la fonction f en tenant compte du domaine de définition.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Représenter des fonctions sur un même graphique en affichant une légende

sans utiliser la commande `lambda`

Exercice 2 On considère la fonction f définie sur $\mathbb{R}$ par $f(x) = \arctan(x)$, et on note $T(x) = x - \dfrac{x^3}{3}$ un de ses polynômes de Taylor associés en 0.

1. Dans le langage Python, construire la fonction *approximation* qui pour tout couple (a, b) donné, renvoie les courbes associées à chacune de ces fonctions sur $[a, b]$. On veillera évidemment à ajouter une légende et à ajuster la fenêtre d'affichage.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

en utilisant la commande `lambda`

Exercice 3 On considère la fonction f définie sur $\mathbb{R}$ par $f(x) = \arctan(x)$, et on note $T_3(x) = x - \dfrac{x^3}{3}$ et $T_5(x) = x - \dfrac{x^3}{3} + \dfrac{x^5}{5}$ des polynômes de Taylor en 0.

1. Dans le langage Python, construire la fonction *approximation* qui pour tout couple (a, b) donné, définit les fonctions f, T_3 et T_5 au cours du programme, puis renvoie les courbes associées dans un même graphique.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Déterminer la valeur d'une somme ou d'un produit de termes

avec une boucle `for`

Exercice 4 On considère la suite (S_n) définie pour tout $n \in \mathbb{N}$ par :

$$S_n = \sum_{k=0}^{n} \frac{(-1)^k}{(k+1)^2}$$

1. Dans le langage Python, construire la fonction *somme* qui pour tout entier n donné, renvoie la valeur de S_n. On veillera à utiliser ici une boucle `for`.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

avec une boucle `while`

Exercice 5 Soit $x \in \mathbb{R}$. On considère la suite (S_n) définie pour tout $n \in \mathbb{N}$ par :

$$S_n(x) = \sum_{k=0}^{n} \frac{x^k}{k!}$$

1. Dans le langage Python, construire la fonction *somme* qui pour tout couple (n, x) donné, renvoie la valeur de $S_n(x)$. On veillera à utiliser ici une boucle `while`.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Déterminer le n-ième terme d'une suite récurrente

avec une suite $(u_n) \in \mathbb{K}^{\mathbb{N}}$ vérifiant $u_{n+1} = f(u_n)$

Exercice 6 On considère la suite $(u_n) \in \mathbb{R}^{\mathbb{N}}$ définie par :

$$u_0 = \frac{1}{3} \text{ et pour tout } n \in \mathbb{N},\ u_{n+1} = \frac{1}{5}u_n + 2$$

1. Dans le langage Python, construire la fonction u qui pour tout entier n donné, renvoie la valeur de u_n.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

avec une suite $(u_n) \in \mathbb{K}^{\mathbb{N}}$ vérifiant $u_{n+2} = f(u_n, u_{n+1})$

Exercice 7 On considère la suite $(u_n) \in \mathbb{R}^{\mathbb{N}}$ définie par :

$$u_0 = 0, u_1 = 1 \text{ et pour tout } n \in \mathbb{N},\ 8u_{n+2} - 2u_{n+1} - u_n = 0$$

1. Dans le langage Python, construire la fonction u qui pour tout entier n donné, renvoie la valeur de u_n.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Calculer les termes successifs d'une suite numérique et stocker les résultats dans une liste

Exercice 8 On considère la suite $(u_n) \in \mathbb{R}^{\mathbb{N}}$ définie par :

$$u_0 = \frac{\pi}{3} \text{ et pour tout } n \in \mathbb{N},\ u_{n+1} = \sin(u_n)$$

1. Dans le langage Python, construire le programme *suite* qui pour tout entier n donné, calcule les termes successifs de la suite (u_n) puis renvoie la liste $[u_0, u_1, \ldots, u_n]$, ainsi que la valeur de u_n.
2. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Représenter graphiquement les termes consécutifs d'une suite réelle

Exercice 9 On considère la série harmonique (H_n) définie pour tout $n \in \mathbb{N}^*$ par :

$$H_n = \sum_{k=1}^{n} \frac{1}{k}$$

et on pose $u_n = H_n - \ln(n)$.

1. Dans le langage Python, construire la fonction u qui pour tout entier n non nul, renvoie la valeur de u_n.

2. Construire alors le programme *graphe* qui pour tout entier n non nul, renvoie dans un même graphe les points M_k de coordonnées (k, u_k).

3. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.

Déterminer un seuil satisfaisant une condition donnée

Exercice 10 On considère la fonction $f : x \in \mathbb{R} \longmapsto x^3 - 3$, et on définit les suites (a_n) et (b_n) par $a_0 = 0, b_0 = 3$ et la relation de récurrence suivante :

- si $f(a_n)f(\frac{a_n + b_n}{2}) < 0$, alors $\begin{cases} a_{n+1} = a_n \\ b_{n+1} = \frac{a_n + b_n}{2} \end{cases}$

- si $f(a_n)f(\frac{a_n + b_n}{2}) \geq 0$, alors $\begin{cases} a_{n+1} = \frac{a_n + b_n}{2} \\ b_{n+1} = b_n \end{cases}$

1. Dans le langage Python, construire la fonction *dichotomie* qui pour tout entier n donné, renvoie la valeur des bornes a_n et b_n encadrant l'unique solution de l'équation $x^3 - 3 = 0$.

2. Construire alors le programme *seuil* qui pour tout $\epsilon > 0$ donné, calcule les termes a_n et b_n tant que $b_n - a_n > \epsilon$, puis renvoie le premier indice n_0 pour lequel $b_{n_0} - a_{n_0} \leq \epsilon$.

3. Recopier alors votre programme. On n'oubliera pas les modules à importer pour le bon fonctionnement du programme.