DST 2 IPT

Exercice 1 - Quelques fonctions en python.

1. Vous avez oublié quelle commande python renvoie la valeur absolue. Proposer une fonction *valeur_abs(x)* qui renvoie la valeur absolue de x.
2. Etant donnée une liste L de nombres réels, proposer une fonction *mon_min(L)* qui renvoie son minimum.
3. Proposer une fonction *change_point(chaine)* qui prend en argument une chaîne, et qui remplace toutes les virgules par des points.
   On rappelle qu'une chaîne n'est pas dynamique, c'est-à-dire ne peut pas être modifiée en place : il faudra créer une nouvelle chaîne.
4. Soit la suite définie par $u_{n+1} = 2u_n(1 - u_n)$ et $u_0 = \frac{1}{4}$.
   Ecrire une fonction *calcul(N)* qui calcule les termes de la suite jusqu'à u_N, et renvoie la liste des valeurs de la suite, de u_0 jusqu'à u_N.
5. Pour la même suite, écrire une deuxième fonction *calcul_2(r)*, qui dépend d'un réel $r > 0$, et qui calcule les termes de la suite jusqu'à ce que $|u_{n+1} - u_n| < r$, ou bien qui s'arrête à la centième itération si le critère d'arrêt $|u_{n+1} - u_n| < r$ n'a jamais été vérifié. La fonction renverra le dernier terme calculé.

Exercice 2 - Représentation des nombres.

1. Donner (en détaillant) la représentation en base 10 des nombres suivants :
   - **a.** Le nombre $(10100101)_2$.
   - **b.** Le nombre $(1A3)_{16}$.
2. Donner (en détaillant) la représentation en base 2 des nombres suivants :
   - **a.** Le nombre $(14)_{10}$.
   - **b.** Le nombre $(315)_{10}$.
   - **c.** Le nombre $(28)_{16}$.
3. On rappelle qu'en représentation en complément à 2 sur n bits (complément à 2^n pour être exact !), on peut représenter tous les nombres entiers de l'intervalle $[-2^{n-1}, 2^{n-1} - 1]$: si un nombre x vérifie $0 \leqslant x \leqslant 2^{n-1} - 1$, on le code normalement en binaire, sinon on le code par la représentation binaire de $x + 2^n$.
   - **a.** Coder sur 8 bits, en complément à deux, les nombre $x = 73$ et $y = -89$
   - **b.** Donner la valeur des entiers relatifs suivants, codés sur 6 bits en complément à deux : $M = (101010)_2$ et $N = (010101)_2$.
4. Dans cet exercice, on considérera des nombres flottants, donnés par leur représentation par signe, exposant et mantisse. On utilise ici une représentation de flottants sur 9 bits « maison » (ça fait moins de chiffres !)
   - Le premier bit donne le signe,
   - Les 4 suivants donnent l'exposant e, et l'exposant final est donné par $E = e - E_{max}$ où $E_{max} = 2^3 - 1$.
   - Les 4 suivants donnent la mantisse, qui se calcule en puissances négatives de 2 de manière standard. Elle est comprise entre 1 et 2 grâce à son « bit caché ».
     La formule finale est celle habituelle :
     $$x = (-1)^s \times m \times 2^E.$$
   - **a.** Quel est le plus grand réel que l'on peut coder avec cette représentation ?
   - **b.** Que vaut le nombre $A = 101011000$ dans cette représentation ?