Dichotomie

Exercice 1 : **Manipulation de données par dichotomie**

Le fichier `liste_triee.txt` contient des valeurs numériques entières triées par ordre croissants.

Q1 : Ecrire un programme qui importent ces données dans la mémoire vive sous la forme d'une liste d'entiers.

Q2 : Ecrire une fonction qui prennent en argument une liste de données triées par ordre croissant et un nombre x, et qui retourne les deux valeurs successives de la liste encadrant strictement le nombre x (on étudiera les différents cas particuliers que l'on peut rencontrer).

Q3 : Ecrire une fonction qui prennent en argument une liste de données triées par ordre croissant et un nombre x, et qui insère x dans la liste au bon endroit (on étudiera les différents cas particuliers que l'on peut rencontrer). Une nouvelle liste est renvoyée.

Q4 : Utiliser les fonctions ci-dessus pour trier la liste de valeurs contenus dans le fichier `liste_non_triee.txt`.

Exercice 2 : **Dichotomie sur une fonction**

On cherche à déterminer les racines du polynôme $f(x)$, avec une précision ε de 10^{-16} :

$$f(x) = x^3 - 2x^2 + 1$$

Partie 1 : Prise en main :

- Créer la fonction python qui, à l'argument x, renvoie :

$$f(x) = x^3 - 2x^2 + 1$$

- Tracer la courbe associée à cette fonction dans l'intervalle $[-1; 2]$ (pas de discrétisation : 0,01).

Partie 2 : Dichotomie

Q1 : A quelle condition la résolution par dichotomie est-elle opérationnelle ?

Q2 : Rappeler les arguments nécessaires à la méthode de dichotomie

Q3 : A partir du tracé de la courbe de f, préciser trois intervalles $[a_i, b_i]$ ($i \in \{1,2,3\}$) incluant une racine et où la fonction est monotone.

En utilisant l'algorithme de dichotomie vue en cours, programmer une fonction qui aux trois arguments : $fonction$, $[a, b]$, ε, renvoie la racine trouvée.

Q4 : Quelles valeurs de racine trouvez-vous ?

Codage des entiers et des flottants

Le module numpy permet d'imposer le type de certaines variables. Pour cela, il faut importer les types nécessaires :

☞ Taper les instructions :

```
from numpy import uint8
from numpy import float64
from numpy import float16
```

Exercice 3 : **Codage des nombres entiers naturels sur 8 bits.**

✍ Justifier le résultat de l'instruction : `print(uint8(260))`

On pose `a = uint8(280)` et `b = uint8(240)`.

✍ Que valent `a`, `b`, `a+b`, `a-b`, `a//b` et `a/b` ? (justifier)

Exercice 4 : **Codage des nombres à virgule flottante**

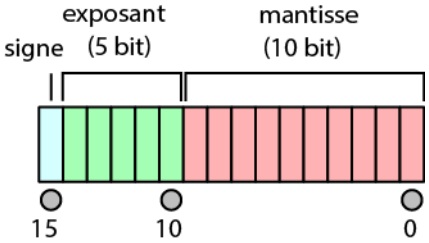
☞ Taper les instructions :

```
x=0.00005
x16=float16(x)
x64=float64(x)
print(1/x16)
print(1/x64)
print(format(1/x16,'.50f'))
print(format(1/x64,'.50f'))
```

✍ Justifier chaque affichage

Exercice 5 : **Décodage des nombres à virgule flottante sur 16 bits**

Un nombre mystère est codé en machine en respectant la convention `float16`, c'est-à-dire en virgule flottante sur 16 bits. On rappelle ci-dessous la répartition des bits qui le compose, et les relations permettant de décoder sa valeur :



$$x = (-1)^s \cdot m \cdot 2^e$$
$$m = 1 + \sum m_i . 2^{-i}$$
$$e = \sum e_i . 2^i - 15$$

L'expression du nombre en machine est la suivante :

0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0

Q1. Quelle est la valeur décimale de ce nombre ?

La syntaxe ci-dessous permet d'afficher le code binaire d'un nombre `float`.

```
From numpy import float16
a=float16(???)
print(bin(a.view('int16')))
```

Q2. En utilisant cette syntaxe, vérifier la valeur décimale trouvée.

Q3. Quelle est la valeur du double de ce nombre ? Quelle est son expression codée en `float16` ?

Q4. Quelle est la précision sur ce nombre ?

Exercice 6 : **Calculs corrects et incorrects en nombres flottants.**

La syntaxe ci-dessous permet d'afficher les valeurs de la mantisse et de l'exposant d'un nombre flottant.

```
from math import frexp
print("mantisse : %1.26f , exposant : %2d" % frexp(???))
```

(remplacer les ??? par la valeur étudiée)

Q1. En utilisant le cde de la partie précédente, comparer les valeurs des mantisses et exposants des nombres suivants : 0.1 0.2 et 0.3

Q2. Justifier que le test 0.1+0.1==0.2 soit vrai, et que 0.1+0.1+0.1==0.3 soit faux