Institut **P**réparatoire aux **E**tudes d'**I**ngénieurs de **N**abeul
Informatique : TP Python

TP 1. Simulation Numérique
Modules de calcul scientifique : `Numpy` et `Matplotlib`

Exercice 1 :

Construire un objet de type `numpy.ndarray`

a dont les termes sont les multiples de 3 compris entre 0 et 27 ;
b dont les termes sont les puissances de 2 de $2^0 = 1$ et $2^{12} = 4096$;

Exercice 2 :

Pour cet exercice, on prend n = 1000000. On pourra augmenter ou diminuer cette valeur en fonction de la machine utilisée.

1. Calculer $\sum_{i=0}^{n} i$ sans utiliser `numpy`.
2. Chronométrer le temps nécessaire pour le calcul précédent, par exemple en utilisant `time.clock()`.

```
import time # Permet de mesurer le temps écoulé et le temps CPU
            # (le temps passé par un processus en mémoire centrale)
e0 = time.time() # temps écoulé en sécondes depuis l'époque
                 # (01-01-1970 00:00:00)
c0 = time.clock() # temps CPU total en sécondes passé dans le script
                  # depuis le début de son exécution
temps_ecoule = time.time() - e0
cpu_time = time.clock() - c0
```

3. Utiliser un tableau `numpy` et la méthode `sum` pour calculer à nouveau la somme proposée.
4. Comparer le temps de calcul avec la méthode précédente.

Exercice 3 :

On peut justifier que :

$$\pi = 2\prod_{n=1}^{+\infty} \frac{4n^2}{4n^2-1}$$

appelé le produit de Wallis.

1. Écrire une fonction itérative, d'argument n, calculant :

$$2\prod_{i=1}^{n} \frac{4i^2}{4i^2-1}$$

2. Écrire une fonction utilisant un tableau numpy, effectuant le même calcul.
3. Comparer le temps de calcul de ces deux fonctions.

Fonctions vectorisées de numpy

Le module `numpy` définit un bon nombre de fonctions mathématiques qui sont **vectorialisées**. Cela signifie qu'une telle fonction $x \longrightarrow F(x)$, accepte en arguments des tableaux numériques T et retourne les tableaux $[F(T[1]), \ldots, F(T[l-1])]$, si $l = len(T)$.
Avec $len(T)$ est la longueur du tableau T.

```
>>> import numpy as np
>>> L = np.array([6,4,3,2,1]); L
array([6, 4, 3, 2, 1])
>>> np.pi/L
array([ 0.52359878,  0.78539816,  1.04719755,  1.57079633,  3.14159265])
>>> np.cos(np.pi/L) #essayer avec la fonction cos du module math
array([  8.66025404e-01,   7.07106781e-01,   5.00000000e-01,
         6.12323400e-17,  -1.00000000e+00])
```

Vous pouvez sous `numpy` **vectorialiser vos propres fonctions** avec la fonction **vectorize** comme illustré ci-dessous :

```
>>> import numpy as np
>>> def f(x):
        if(x>0):
                return 1
        elif x==0:
                return 0
        else:
                return -1
>>> t = np.linspace(-2,2,11); t
array([-2. , -1.6, -1.2, -0.8, -0.4,  0. ,  0.4,  0.8,  1.2,  1.6,  2. ])
>>> f(t) # Erreur
>>> vf = np.vectorize(f)
>>> vf(t)
array([-1, -1, -1, -1, -1,  0,  1,  1,  1,  1,  1])
```

Exercice 4 :

Soit la fonction $h(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2}$. Créer des tableaux xt et ht qui contiennent respectivement 41 valeurs reparties uniformement pour $x \in [-4,4]$ et l'évaluation de la fonction h sur ces valeurs.

Exercice 5 :

Ecrire un script qui demande à l'utilisateur des nombres $a_1, a_2, b_1, b_2, c_1, c_2$ et trace le triangle ABC.

Exercice 6 :

Réaliser le graphe de la fonction $y(t) = v_0 t - \frac{1}{2}g t^2$ pour $v_0 = 10$, $g = 9.81$, et $t \in [0, 2v_0/g]$. Le label sur l'axe des x devra être "temps (s)" et le label sur l'axe des y "hauteur (m)".

Exercice 7 :

La fonction $f(x,t) = e^{-(x-3t)^2} sin(3\pi(x-t))$ décrit pour une valeur fixe de t une onde localisée en espace. Faire un programme qui visualise cette fonction comme une fonction de x dans l'intervalle $x \in [-4,4]$ pour $t = 0$.