

```

####
# DEVOIR INFORMATIQUE 4
####

## CENTRALE 2015

# Partie I

# A
[1,2,3] + [4,5,6] # donne [1, 2, 3, 4, 5, 6] : concaténation
2 * [1, 2, 3] # donne [1, 2, 3, 1, 2, 3]

# B
def smul(a, L):
    """définit la multiplication externe
de vecteurs """
    res = [0] * len(L)
    for k in range(len(L)):
        res[k] = a * L[k]
    return(res)

def smul2(a,L):
    """définit la multiplication externe
de vecteurs par compréhension"""
    return([a*x for x in L])

# C
def vsom(L1, L2):
    """définit la somme de vecteurs """
    res = [0] * len(L1)
    for k in range(len(L1)):
        res[k] = L1[k] + L2[k]
    return(res)

def vdif(L1, L2):
    """définit la somme de vecteurs """
    return([L1[k] - L2[k] for k in range(len(L1))])

# Partie II
# A ....
# B
# 1)  $y(i+1) = y(i) + h z(i)$  et  $z(i+1) = z(i) + h f(y(i))$ 
#
def euler(f, n, h, y0, z0):
    """ résolution de  $y' = f(y(t))$  on pose  $z = y'$  """
    # Initialisation
    y,z = y0, z0 # Valeurs initiales des suites
    Ly , Lz = [y] , [z] # Listes de stockage
    for i in range(n-1): # n points au total ... dont y0 et z0
        fy = f(y) # Calcul de  $f(y_i)$  avant de changer y
        y = y + z*h # Calcul de  $y_{i+1}$ 
        z = z + fy*h # Calcul de  $z_{i+1}$ 
        Ly.append(y) # Ajout de  $y_{i+1}$  à la liste des y
        Lz.append(z) # Idem pour la liste des z
    return(Ly, Lz) # Retour des listes

#
def verlet(f, n, h, y0, z0):
    """ résolution par Verlet de  $y' = f(y(t))$  on pose  $z = y'$  """
    # Initialisation
    y,z = y0, z0 # Valeurs initiales des suites
    Ly , Lz = [y] , [z] # Listes de stockage
    FI = f(y) # Calcul de  $f_i$ 
    for i in range(n-1): # n points au total ... dont y0 et z0
        y = y + z*h + h*h*FI/2 # Calcul de  $y_{i+1}$ 
        GI = f(y) #  $f(y_{i+1})$ 
        z = z + h*(FI+GI)/2 # Calcul de  $z_{i+1}$ 
        Ly.append(y) # Ajout de  $y_{i+1}$  à la liste des y
        Lz.append(z) # Idem pour la liste des z
        FI = GI # l'ancien  $f_{i+1}$  sera le prochain  $f_i$ 
    return(Ly, Lz) # Retour des listes

```

```

#
def distance(p1, p2):
    p1p2 = vdif(p1,p2)
    dis = 0
    for i in range(len(p1p2)):
        dis += p1p2**2
    return(sqrt(dis))

def force2(m1, p1, m2, p2):
    r12 = distance(p1,p2)
    pos = vdif(p2, p1)
    G = 6.67e(-11)
    coeff = G * m1 * m2/(r12**3)
    return(smul(coeff,pos))

def forceN(j,m,pos):
    force = [0]*len(pos[j])
    for k in range(len(m)):
        if k != j:
            F_k_sur_j = force2(m[j], pos[j], m[k], pos[k])
            force = vsom(force, F_k_sur_j)
    return(force)

def pos_suiv(m, pos, vit, h):
    pos2 = []
    for j in range(len(m)):
        force = forceN(j, m, pos)
        newpj = [0]*len(pos[j])
        for k in range(len(pos[j])):
            newpj[k] = pos[j][k] + h*vit[j][k] +(h*h/2)*force[k]/m[j]
        pos2.append(newpj)
    return(pos2)

def etat_suiv(m, pos, vit, h):
    newpos = pos_suiv(m, pos, vit, h)
    newvit = []
    for j in range(len(m)):
        fi = smul(1/m[j], forceN(j,m,pos))
        fip1 = smul(1/m[j], forceN(j,m,newpos))
        nextvitj = [0]*len(vit[j])
        for k in range(len(vit[j])):
            nextvitj[k] = vit[j][k] + (h/2)*(fi[k] + fip1[k])
        newvit.append(nextvitj)
    return(newpos, newvit)

##
# Exercice II

# a
"""
f(x) = x**3 + 3 * x**2 - 6 * x + 2
On a f'(x) = 3 * ( x**2 + 2*x - 2) = 3( (x+1)**2-3)
f' s'annule une et une seule fois entre 0 et 1 :
en a = sqrt(3)-1
De plus f'(x) < 0 si x<a et f'(x) > 0 si x > a
DOnc f décroît sur [0,a] et croît sur [a,1]
Or f(1) = 0 donc f(a) < 0 et f ne s'annule pas sur [a,1[
On peut aussi dire : f(a) = 10 - 6*sqrt(3) <0
Mais comme f(0) = 1 > 0, f est une bijection de [0,a] vers [f(a), 1]
En particulier f s'annule une et une seule fois sur [0, a]
Ainsi f s'annule une et une seule fois sur ]0,1[
"""

# b
f = lambda x: x**3 + 3* x**2 - 6 *x + 2

```

```
fp = lambda x: 3 * ( x**2 + 2*x - 2)
x0 = 0.
```

```
u, k = x0, 0
while k < 6 :
    u , k = u - f(u) / fp(u) , k + 1
    print(u)
```

```
"""
On trouve successivement  $u = (1/3) = 0.3333\dots$ 
puis  $u = 43 / 99 = 0.434343\dots$ 
"""
```