np_tuto2.py

```
##################################################
###########TP-TD n18: numpy(session 2)###########
##################################################
import numpy as np

####################
# 0-D array
l0=120
a0 = np.array(l0)
print('a0=',a0)
####################
# 1-D array
l1=[3, 1, 7, '5']
a1 = np.array(l1)
print('a1=',a1)
####################
# 2-D array
l2=[[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15+15j]]
a2 = np.array(l2)
print('a2=',a2)

# ##### Dimension d'une matcrice
# syntax:**<array>.ndim**

print('{} est de dimension {}\n--------'.format(a0,a0.ndim))
print('{} est de dimension {}\n--------'.format(a1,a1.ndim))
print('{} est de dimension {}'.format(a2,a2.ndim))

# ##### forme (ou taille) d'une matcrice
# syntax:**<array>.shape**

print(a2[0],a2[0].shape)
print('{} est de la form {}\n--------'.format(a0,a0.shape))
print('{} est de la form {}\n--------'.format(a1,a1.shape))
print('{} est de la form {}\n--------'.format(a2,a2.shape))
# print(f'{a2.shape[0]} est le nombre de lignes et {a2.shape[1]} est le nombre de
colonnes de {a2}')
print('{} est le nombre de lignes et {} est le nombre de colonnes de
{}'.format(a2.shape[0],a2.shape[1],a2))

# ##### Type de donnees d'une matcrice
# syntax:**<array>.dtype**


print('data type de a0 est:',a0.dtype)
print('data type de a1 est:',a1.dtype)
print('data type de a2 est:',a2.dtype)

print('Choix automatique des donnees')
# lors de la definition d'un array sans preciser le type des donnees, le type des
donnees est choisi automatiquement.
a = np.array([3, 1, 7, 5]) # les donnes sont homogenes, donc le type des donnees est int
print(a)
```

```
a = np.array([3, 1, 7, '5']) # presence dun string, donc le type des autres donnees sont string
print(a)
a = np.array([3, 1, 7, 5+1j]) # presence dun complex, donc le type des autres donnees sont complex
print(a)

print('*'*50)
print('fixe des donnees par "dtype"')

# lors de la definition de a, on peut definir le type de donnees (int, float, complex, string)
a = np.array([3, 1, 7, 5],dtype="str")
print('on a fixe les elements de a comme str: ',a)
a = np.array([3.1, 1.1, 7.7, 5.5],dtype="int")
print('on a fixe les elements de a comme int: ',a)
a = np.array([3, 1, 7, 5],dtype="float")
print('on a fixe les elements de a comme float: ',a)
a = np.array([3, 1, 7, 5],dtype="bool")
print('on a fixe les elements de a comme booleen: ',a)
a = np.array([3, 1, 7, 5],dtype="complex")
print('on a fixe les elements de a comme complex: ',a)

# ---------- Iteration classique  ----------

print
# 1-d array iterating
for x in a1:
    print(x)

# 2-d array iterating
print('Iterating over 2 d array')
for x in a2:
    for y in x:
        print(y,type(y))

# Itération sur tous les éléments du tableau à l'aide de nditer:

print(list(np.nditer(a2)))
for x in np.nditer(a1):
    print(x)
print(50*'*')
for x in np.nditer(a2):
    print(x)

# np.sort(<array>)

print('\n---------- Sort ----------')
print('Sorting array:\n', np.sort(a1))
print('Sorting each row in array: \n', np.sort(a2))
print(np.sort(np.array([[5,2,-3],[7,20,-30]])))

print('\n---------- Transposer ----------')
a2_t = a2.T
print('original 2 d:\n', a2, a2.shape)
print('Transposed 2 d:\n', a2_t, a2_t.shape)
```

```
print(np.diag([1, 2, 3, 4]))
print(np.diag(a2))

print('\n---------- Random numbers ----------')
print('Random integer:', np.random.randint(100),type(np.random.randint(100)))
print('Random integer between 2 numbers:', np.random.randint(10, 80))
print('Random float ben 0-1:', np.random.rand())

print('\n---------- Génération de tableaux ----------')
print('Génération de la matrice entière 2x3 avec des nombres entiers aléatoires:\n',
np.random.randint(10, 90, size=(2, 3)))
print('Génération de la matrice reelle 2x3 avec des nombres aléatoires 0-1:\n',
np.random.rand(2, 3))
print('Génération de matrice zéros 2x3:\n', np.zeros((2, 3), float))
print('Génération de matrice "des 1" 2x3:\n', np.ones((2, 3), int))

print('\n---------- copie (listes)----------')
l=[1,2]
l11=l
l12=l.copy()
l.append(99)
l11.pop(0)
print(l,l11,l12)
print('\n---------- copie (array)----------')

b2=a2
a2[0,0]=4444
print(b2,a2)

a2_c = a2.copy()
a2_c[0, 1] = 1000192
print('Original 2d:\n', a2)
print('copied 2d:\n', a2_c)

print('---------- Opérations mathématiques ----------')
a = np.array([1, 2])
b = np.array([3, 4])
print(a+100)
print('Result of a + b: ', np.add(a, b))#print(a+b))
print('Result of a - b: ', np.subtract(a, b))#print(a-b))
print('Result of a * b: ', np.multiply(a, b))
print('Result of a / b: ', np.divide(a, b))
print('Result of a ^ b: ', np.power(a, b))
print('Result of a ^ 2: ', np.power(a, 2))

subd1=np.linspace(0, 10, 5)
subd2=np.linspace(0, 10, 10)
subd3=np.linspace(0, 10, 100)
print('subd1 de l intervalle [0,10]',subd1)
print('subd2 de l intervalle [0,10]',subd2)
print('subd3 de l intervalle [0,10]',subd3)

print('\n---------- Fonctions usuelles ----------')
a = np.array([1, 1,5,10])
print(np.sin(a))
print(np.cos(a))
print(np.tan(a))
```

```
print(np.exp(a))
print(np.log(a))
print(np.sqrt(a))

# Application
# 1
y=np.sin(a)+3*np.cos(a)/(np.exp(a)+np.tan(a))
# print(y)
# 2. dessiner la courbe de f  sur [0,10] point par point
# soi f(x)=sin(x)+3*cos(x)/(exp(x)+tan(x))
abscisses=np.linspace(0,10,5)
ordonnees=np.sin(abscisses)+3*np.cos(abscisses)/(np.exp(abscisses)+np.tan(abscisses))
print(abscisses,ordonnees)

print('\n---------- Fonctions statistiques ----------')

print('Min item in array:', np.amin(a2))
print('Max item in array:', np.amax(a2))
print('Median of array:', np.median(a2))
print('standard deviation of array:', np.std(a2))
print('Average of array:', np.average(a2))
print('Mean of array:', np.mean(a2))
print('Variance of array:', np.var(a2))

print('\n---------- Algebra functions ----------')
a = np.array([2, 4])
b = np.array([3, 5])
print('1-d Matrices multiplication:\n', np.matmul(a, b))

a = np.array([[2, 4], [1, 3]])
b = np.array([[3, 5], [4, 6]])
print('2-d Matrices multiplication:\n', np.matmul(a, b))

print('\n---------- Algebra functions ----------')
a = np.array([1, 2, 3, 5, 6, 1, 2, 3, 4, 1, 1, 2, 6])
print('Histogram of array', np.histogram(a))

a=np.array([[1,2,3],[10,20,30]])
b=np.array([[1,2],[10,20],[5,6]])
aXb=a.dot(b)
bXa=b.dot(a)

print(aXb)
print(5*'*')
print(bXa)
```