```
# -*- coding: utf-8 -*-
from numpy import *
from numpy.linalg import *
from numpy.random import *
from matplotlib.pyplot import *

print('Valeur de pi :', pi)
print(finfo(float).eps)

#Create a Vector
X1 = array([1, 2, 3, 4, 5])
print('Simple vecteur :',X1)

X2 = arange(0,1,.25)
print('Subdvision uniforme :', X2)

X3 = linspace(0,10,3)
print('Vecteur n-pts :',X3)

X4 = zeros(5)
print('Vecteur zeros :',X4)

#Matrice
M1=array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print('Simple matrice', M1)

M2=zeros((2,3))
print('Matrice zeros', M2)

#Exercice 1
A=-1*(diag(ones(9),1)+diag(ones(9),-1))+2.*eye(10);
A[9,9] = 1
print(A)

#Fonctions
def fonc(x, y, m):
    z = x**2 + y**2
    t = x - y + m
    return z, t
#Test
if(1<2) :
    print('1<2')
elif (1<3) :
    print('1<3')
else :
    print('Aussi non !')

##################
##################
##  Exercice 3  ##
##################
##################
def f(x, m) :
    if (m == 0) :
        y = zeros(size(x))
    elif (m == 1) :
        y = ones(size(x))
    elif (m == 2) :
        y = x
    elif (m == 3) :
        y = x**2
    elif (m == 4) :
        y = 4.*pi*pi*sin(2.*pi*x)
    else :
        print('valeur de m indéfinie')
    return y

def solex(x, ug , m) :
    if (m == 0):
        y = ug*ones(size(x))
    elif (m == 1) :
        y = -0.5*x**2+x+ug
```

```
    elif (m == 2) :
        y = -(1/6)*x**3+(1/2)*x+ug
    elif (m == 3 ) :
        y = -(1/12)*x**4+(1/3)*x+ug
    elif (m == 4) :
        y = sin(2*pi*x)-2*pi*x+ug
    else :
        print('valeur de m indéfinie')
    return y


##################
##################
##  Exercice 2  ##
##################
##################


print('Pour le second membre, choix de la fonction f')
print('Pour m=0 : f=0')
print('Pour m=1 : f=1')
print('Pour m=2 : f=x')
print('Pour m=3 : f=x^2')
print('Pour m=4 : f=4*pi^2*sin(2*pi*x)')
m = int(input("Choix de m = "))

print('Choix de la condition a gauche')
ug = float(input('ug = u(0) ='))

print("Methode pour l'approximation de u'(1) : ")
print("1- decentre d'ordre 1")
print("2- centre d'ordre 2")
meth = int(input('Choix = '))

print('Choix du nombre Ns de points interieurs du maillage')
Ns = int(input('Ns = '))

# Maillage
h=1./(Ns+1)
X=linspace(0, 1., Ns+2)
Xh=linspace(h,1.,Ns+1)

# Matrice du systeme lineaire :
A=-1*(diag(ones(Ns),1)+diag(ones(Ns),-1))+2.*eye(Ns+1);
A[Ns, Ns] = 1
A=1./h/h*A
#Conditionement de la matrice
cond_A=cond(A)
print('Conditionnement de la matrice A :', cond_A)



# Second membre
# b = ... (plus loin, exercice 3)
b = f(Xh, m)
b[0] = b[0] + (Ns + 1)**2*ug


# Transformation de b[Ns] pour prendre en compte u'(1) = 0 (cf TD)
if (meth == 2):
    b[Ns] = b[Ns]/2


# Resolution du syteme lineaire
Uh = solve(A, b) # ceci calcule Uh solution du systeme AU=b

# Calcul de la solution exacte aux points d'approximation

Uex = solex(Xh, ug, m)

# Calcul de l'erreur en norme infini
Uerr = abs(Uex - Uh)
```

```
145  disp(max(Uerr))
146
147  #Graphes
148  Uh = concatenate((array([ug]),Uh))
149  # on complete le vecteur solution avec la valeur ug en 0
150  # On trace le graphe de la fonction solex sur un maillage fin de 100 points
151  plot(linspace(0,1,100),solex(linspace(0,1,100), ug, m), label = 'sol ex')
152
153  # et le graphe de la solution approchée obtenue
154  plot(X, Uh, label = 'sol approchee')
155
156
157  plot(Xh, Uerr, label = 'Erreur')
158  # On ajoute un titre
159  title('d²u(x)/dx²=f(x)')
160
161  # On ajoute les labels sur les axes
162  xlabel('X')
163  ylabel('Y')
164
165  # Pour faire afficher les labels
166  legend()
167  show() # Pour afficher la figure
168
```