```
# -*- coding: utf-8 -*-
"""
Created on Wed May 17 06:26:19 2017

@author: stephane
"""

from numpy import array, linspace, log   # Exos 1, 2
from numpy.linalg import eigvals         # Exo 1
import matplotlib.pyplot as pypl         # Exos 1, 2
from scipy.integrate import odeint       # Exo 2
from numpy.random import randint         # Exo 3


"""
Premier exercice : UN SPECTRE
"""


def M(t):
    return array([[t, 0, 1], [0, 0, 1], [1, 1, 0]])


def tri(t):  # passionant !
    a, b, c = t
    if a <= b:
        if b <= c:
            return [a, b, c]
        else:
            if a <= b:
                return [a, c, b]
            else:
                return [b, a, c]
    else:
        if a <= c:
            return [b, a, c]
        else:
            if b <= c:
                return [b, c, a]
            else:
                return [c, b, a]


def spectre(t):
    return tri(eigvals(M(t)))

les_t = linspace(-3, 3, 100)
sp1, sp2, sp3 = [], [], []
for t in les_t:
    vap = spectre(t)
    sp1.append(vap[0])
    sp2.append(vap[1])
    sp3.append(vap[2])


pypl.plot(les_t, sp1, label=r"$\lambda_1$")
pypl.plot(les_t, sp2, label=r"$\lambda_2$")
pypl.plot(les_t, sp3, label=r"$\lambda_3$")
pypl.plot(les_t, les_t, 'black')
pypl.legend(loc='upper left')
pypl.axhline(0, color='black')
pypl.axvline(0, color='black')
```



```
pypl.grid()
pypl.savefig('dessin−spectre−symetrique.pdf')
pypl.clf()


"""
Deuxième exercice : UNE ÉQUATION DIFFÉRENTIELLE
"""

a, b, c, d = 2, 0.3, 1.5, 0.5


def f0(x):
    if x == 0:
        return 1
    else:
        return -log(1-x)/x

# Lire la documentation de odeint; attention à l'ordre des paramètres !


def F(y, x):
    return f0(x)*y


les_x = linspace(0, 0.99, 100)

solution = odeint(F, 1, les_x)
pypl.plot(les_x, solution, '--', linewidth=4, label='Vraie solution')

les_x = linspace(0, -1.5, 100)

solution = odeint(F, 1, les_x)
pypl.plot(les_x, solution,  '--', color='blue', linewidth=4)

pypl.grid()

a = [1.] * 11
for n in range(1, 11):
    a[n] = sum(a[n-k]/k for k in range(1, n+1))/n


def S(n, x):
    return sum(a[k]*x**k for k in range(n+1))

les_x = linspace(-1.2, 0.99)
for n in range(0, 11, 2):
    les_y = [S(n, x) for x in les_x]
    pypl.plot(les_x, les_y, label=r"$n=%i$" % n)
pypl.legend(loc='upper left')

pypl.savefig('dessin−ed−debalme.pdf')
pypl.clf()
```

```
"""
Troisième exercice : DES PROBABILITÉS
"""


def p(a, b, n):
    if n == 0:
        return 1
    else:  # ATTENTION au décalage
        return (b+(n-1)*a)/((n-1)*a+2.*b)*p(a, b, n-1)


def premiers_termes(a, b, N):
    return [p(a, b, n) for n in range(N)]

"""
>>> premiers_termes(5, 10, 10)
[1, 0.5, 0.3, 0.19999999999999998, 0.14285714285714285, 0.10714285714285714,
0.08333333333333333, 0.06666666666666667, 0.05454545454545455,
0.04545454545454546]
>>> premiers_termes(10, 5, 10)
[1, 0.5, 0.375, 0.3125, 0.2734375, 0.24609375, 0.2255859375, 0.20947265625,
0.196380615234375, 0.1854705810546875]
"""


def simu(a, b):
    bl, no = b, b
    n = 1  # le tirage en cours
    while True:  # On croise les doigts pour que ça termine !
        if randint(1, bl+no) <= bl:
            bl = bl + a
        else:
            return n
        n = n + 1
"""
>>> expe = 10**5
>>> sum(simu(5, 10) for _ in range(expe))/float(expe)
3.52826

>>> [simu(10, 5) for _ in range(5)]
[22, 133, 2, 1, 1]
>>> [simu(10, 5) for _ in range(10)]
[1, 27, 28, 1, 3184, 105, 1, 22, 160, 2]

>>> [simu(5, 5) for _ in range(20)]
[2, 1, 1, 1, 2, 2, 34, 1, 21, 1, 3, 4, 6, 1, 1, 2, 6, 15, 1, 1]

L'espérance est finie si et seulement si b >= a (vérifier le cas d'égalité !)
"""


"""
Quatrième exercice : DES NOMBRES PREMIERS
"""


def P(x):
    return x**2-x+41


def T(x):
```



```
    return x**2-79*x+1601

p, t = P(42), T(80)
"""
>>> [P(42+k*p) % p for k in range(−15, 16)]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0]
"""

# Et si l'examinateur veut un "programme"...

probleme = False
for k in range(-15, 16):
    if T(80+k*t) % t != 0:
        probleme = True
"""
>>> probleme
False
"""

# ou encore :

"""
>>> sum((T(80+k*t) % t)**2 for k in range(−15, 16))
0
"""


def premier(n):
    if n <= 1:
        return False
    k = 2
    while k**2 <= n:
        if n % k == 0:
            return False
        k = k+1
    return True

"""
>>> [n for n in range(50) if premier(n)]
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
"""

tous_premiers = True
for k in range(41):
    if not(premier(P(k))):
        tous_premiers = False
for k in range(80):
    if not(premier(T(k))):
        tous_premier = False

"""
>>> tous_premiers
True
>>> premier(P(41)), premier(T(81))
(False, False)
"""
```