

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 1/7
# -*- coding: utf-8 -*-
"""
Created on Sat Sep 17 06:33:04 2016

@author: stephane

Tourne sous 2.7
"""

# Exo 1 : DONE

# Exo 2

somme1 = 0
for k in range(101):
    somme1 = somme1 + k**2

somme2 = sum(k**2 for k in range(101))

print("Exo 2: "+str(somme1)+" vs. "+str(somme2))
"""
Exo 2 : 338350 vs. 338350
"""

# Exo 3

from math import log
S = 0

for i in range(930, 946):
    S = S + 1./i

T = log(946./930)
U = T + (1./930-1./946)/2

print("Exo 3: %.10f\t%.10f\t%.10f" % (S, T, U))
print("\tdelta1 = %.10f\tdelta2 = %.10f" % (S-T, S-U))

"""
Exo 3 : 0.0170670793      0.0170579829      0.0170670761
      delta1 = 0.0000090964      delta2 = 0.0000000032
"""

# Exo 4

somme3 = 0
for k in range(100, 1001):
    if (k**2 + 2*k + 1) % 42 == 0:
        somme3 += k

somme4 = sum([k for k in range(100, 1001) if (k**2 + 2*k + 1) % 42 == 0])

print("Exo 4: "+str(somme3)+" vs. "+str(somme4))
"""
Exo 4 : 11445 vs. 11445
"""

# Exo 5

compteur = 0

```

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 2/7
for i in range(5001):
    if sum(j**4 for j in range(i+1)) % 17 == 0:
        compteur += 1

compteur2 = 0
sommeur = 0
for i in range(5001):
    sommeur += i**4
    if sommeur % 17 == 0:
        compteur2 += 1

print("Exo 5: %i vs. %i" % (compteur, compteur2))
"""
Exo 5 : 1472 vs. 1472

N.B. : plus de 15 secondes pour le premier. Normal : complexités : n**2 vs. n
"""

# Exo 6

def f(x):
    return x**2+2

print("Exo 6: "+str(f(100)))
"""
Exo 6 : 10002
"""

# Exo 7

def f(x):
    if x <= 10:
        return x
    return 5*x-40
    return -3*x+120

import matplotlib.pyplot as pyplot

les_x = range(-10, 41)
les_y = [f(x) for x in les_x]
pyplot.plot(les_x, les_y)

pyplot.grid()
pyplot.axhline(color='black')
pyplot.axvline(color='black')
pyplot.savefig('graphe_f.pdf')

# Exo 8

def factol(n):
    if n == 0:
        return 1
    return n*factol(n-1)

def facto2(n):
    produit = 1
    for k in range(1, 1+n):

```

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 3/7
    produit *= k
    return produit

print ("Exo 8 : %i vs. %i" % (facto1(10), facto2(10)))
"""
Exo 8 : 3628800 vs. 3628800
"""

# Exo 9

def fibo1(n):
    if n <= 1:
        return n
    return fibo1(n-1)+fibo1(n-2)

def fibo2(n):
    if n == 0:
        return 0
    ad, d = 0, 1 # l'avant dernier et le dernier
    for k in range(2, 1+n):
        ad, d = d, d+ad
    return d

print ("Exo 9 : %i vs. %i... et %i" % (fibo1(30), fibo2(30), fibo2(100)))
"""
Exo 9 : 832040 vs. 832040... et 354224848179261915075

Rappel : la version récursive a ici une complexité calamiteuse;
en particulier, fibo1(100) est exclu.
"""

# Exo 10

def syracuse(n):
    if n == 1:
        return 0
    if n % 2 == 0:
        return 1 + syracuse(n//2)
    return 1+syracuse(3*n+1)

def syracuse_it(n):
    cpt, uk = 0, n # uk est le terme courant de la suite, i.e. u[cpt]
    while uk != 1:
        cpt = cpt+1
        if uk % 2 == 0:
            uk = uk//2
        else:
            uk = 3*uk+1
    return cpt

print ("Exo 10 : %i (vs. %i) et %i" % (syracuse(127), syracuse_it(127), \
    max(syracuse(n) for n in range(1,1001))))
"""
Exo 10 : 46 (vs. 46) et 178
"""

# Exo 11

```

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 4/7
t1 = list(range(1, 30, 2))
t2 = [x**2 for x in t1]
t3 = [y for y in t2 if y % 12 == 1]
t4 = [[2**i*(2*j+1) for j in range(4)] for i in range(5)]

print ("Exo 11 : \nt1 = %s\nt2 = %s\nt3 = %s\nt4 = %s" % (t1, t2, t3, t4))
"""
Exo 11 :
t1 = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29]
t2 = [1, 9, 25, 49, 81, 121, 169, 225, 289, 361, 441, 529, 625, 729, 841]
t3 = [1, 25, 49, 121, 169, 289, 361, 529, 625, 841]
t4 = [[1, 3, 5, 7], [2, 6, 10, 14], [4, 12, 20, 28],
      [8, 24, 40, 56], [16, 48, 80, 112]]
"""

# Exo 12

def somme(t):
    s = 0
    for x in t:
        s = s+x
    return s

def produit(t):
    p = 1
    for x in t:
        p = p*x
    return p

def maximum(t):
    assert t != [] # signifie : «ne va pas plus loin si la liste est vide»
    maxi = t[0]
    for x in t:
        if x > maxi:
            maxi = x
    return maxi

def indice_maximum(t):
    assert t != []
    i_maxi = 0
    for i in range(1, len(t)):
        if t[i] > t[i_maxi]:
            i_maxi = i
    return i_maxi

"""
>>> somme(t1), produit(t1), maximum([10, 42, 5, 42, 30]), indice_maximum([10, 42, 5, 42, 30])
(225, 6190283353629375L, 42, 1)
"""

# Exo 13

def inversions(t):
    """ Calcul du nombre d'inversions """
    cpt = 0
    for j in range(1, len(t)):
        for i in range(j):

```

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 5/7

        if t[j] < t[i]:
            cpt += 1
    return cpt

print ("Exo 13: %i et %i" % (inversions([1, 2, 3, 0]), inversions([4, 3, 2, 1])))
"""
Exo 13: 3 et 6
"""

# Exo 14

t = [1]
for j in range(1, 11):
    t.append(sum((i+1)*t[j-1-i] for i in range(j)))
print ("Exo 14: "+str(t))
"""
Exo 14: [1, 1, 3, 8, 21, 55, 144, 377, 987, 2584, 6765]
"""

# Exo 15

def pascal(n):
    if n == 0:
        return [[1]]
    avant = pascal(n-1) # les «k parmi i» pour 0 <= k <= i <= n-1
    derniere = avant[-1] # dernière ligne : les «k parmi n-1»
    nouvelle = [1] # bord gauche
    for k in range(1, n):
        nouvelle.append(derniere[k-1] + derniere[k]) # formule du binôme
    nouvelle.append(1) # bord droit
    avant.append(nouvelle)
    return avant

"""
>>> pascal(4)
[[1], [1, 1], [1, 2, 1], [1, 3, 3, 1], [1, 4, 6, 4, 1]]
"""

# Exo 16

from numpy import array
from numpy.linalg import solve, det, inv

A = array([(i-j)**4 for j in range(5)] for i in range(5))

print ("Exo 16: det = "+str(det(A))+"\nA**3=" +
        str(A.dot(A).dot(A))+"\nA**(-1)="+str(inv(A)))
"""
Exo 16: det = 7962624.0;
A**3=[[ 219232  114998  1186688  6218038  20375648]
 [ 114998  49312  364358  1896608  6218038]
 [ 1186688  364358  136352  364358  1186688]
 [ 6218038  1896608  364358  49312  114998]
 [20375648  6218038  1186688  114998  219232]]
A**(-1)=[[ 0.00376157 -0.02199074  0.05034722  0.01273148 -0.00318287]
 [-0.02199074  0.12962963 -0.30555556  0.01851852  0.01273148]
 [ 0.05034722 -0.30555556  0.76041667 -0.30555556  0.05034722]
 [ 0.01273148  0.01851852 -0.30555556  0.12962963 -0.02199074]
 [-0.00318287  0.01273148  0.05034722 -0.02199074  0.00376157]]
"""

```

```

sept. 17, 16 6:57      tp-spe-reprise-2016-2017.py      Page 6/7

# Exo 17

Y = array([1, 2, 3, 4, 5])
X1 = solve(A, Y)
X2 = inv(A).dot(Y)

print "Exo 17: \n"+str(X1)+" vs.\n"+str(X2)
"""
Exo 17:
[ 0.14583333 -0.54166667  0.75    -0.45833333  0.10416667] vs.
[ 0.14583333 -0.54166667  0.75    -0.45833333  0.10416667]
"""

# Exo 18

from math import cos, sin, sqrt
from scipy.integrate import quad, odeint

def f(t):
    return sin(t+sqrt(t)+1)
integrale = quad(f, 0, 1)

print ("Exo 18: %.5f, précision %.12f" % (integrale[0], integrale[1]))
"""
Exo 18: 0.71699, précision 0.000000000232
"""

# Exo 19

from scipy.optimize import fsolve
from numpy import linspace

def f(t):
    return cos(t**2)+sin(1-t)**2

def g(t):
    return sin(t+sqrt(t)+1)**2

def h(t):
    return quad(g, 0, t)[0] # pour virer la deuxième composante (précision)

def foo(t):
    return h(t)-1

sol1 = fsolve(f, 1.2)
sol2 = fsolve(foo, 2)

pypl.clf()
les_t = linspace(-1, 5, 100)
les_y = map(f, les_t)
pypl.plot(les_t, les_y)
pypl.grid()
pypl.axhline(color='black')
pypl.axvline(color='black')
pypl.savefig('graphe_g.pdf')

```

sept. 17, 16 6:57

tp-spe-reprise-2016-2017.py

Page 7/7

```

pypl.clf()
les_t = linspace(0, 3, 100)
les_y = map(h, les_t)
pypl.plot(les_t, les_y)
pypl.grid()
pypl.axhline(color='black')
pypl.axvline(color='black')
pypl.savefig('graphe_h.pdf')

print ("Exo 19 : %.3f et %.3f" % (sol1, sol2))
"""
Exo 19 : 1.284 et 2.059
"""

# Exo 20

def phi(y, t):
    return cos(t+y**2)

les_t = linspace(0, 3, 100)
les_y = odeint(phi, 0, les_t)

pypl.clf()
pypl.plot(les_t, les_y)
pypl.grid()
pypl.axhline(color='black')
pypl.axvline(color='black')
pypl.savefig('ed1.pdf')

les_t = linspace(0, 2, 100)
y1 = odeint(phi, 0, les_t)[-1] # on ne garde que la dernière valeur

print ("Exo 20 : "+str(y1))
"""
Exo 20 : [0.40007825]
"""

# Exo 21

def phi(z, t):
    return array([z[1], -2*t+cos(z[0])])

les_t = linspace(0, 2, 100)
les_y = odeint(phi, array([0, 0]), les_t)

pypl.clf()
pypl.plot(les_t, les_y[:, 0])
pypl.grid()
pypl.axhline(color='black')
pypl.axvline(color='black')
pypl.savefig('ed2.pdf')

les_t = linspace(0, 1, 100)
y2 = odeint(phi, array([0, 0]), les_t)[-1]

print "Exo 21 : "+str(y2)
"""
Exo 21 : [0.16548171 -0.00511852]
"""

```