```
# -*- coding: utf-8 -*-
"""
Created on Sat Jan  2 17:30:58 2016

@author: Dom
"""

from math import *
import matplotlib.pyplot as plt
import os
import numpy as np

path =r'C:\Users\Admin2\Desktop\Present_ISN' # Att
# si il y a des valeurs numériques derrière \ elle
# de la table utf-8, il faut ajouter la lettre r d

# on teste le répertoire courant de travail
retval = os.getcwd() # cwd veut dire current work
print ("le répertoire de travail est : %s" % retva

# On change de répertoire
os.chdir(path) # chdir pour change directory

# On reteste
retval = os.getcwd()

print("Le répertoire de travail a été modifié selo

#**************************************************
# Lecture du fichier txt
# **************************************************

def lecture(fichier):
    f = open(fichier) # on ouvre le fichier et on
    temps,tensions = [],[] # on prépare deux liste
```

```
    for line in f: # on parcourt les lignes du fi
            line = line.replace(',',';') # on remp
            t,u = line.split(';') # on met les va
            temps.append(float(t)) # on remplit l
            tensions.append(float(u))
    f.close()
    return temps,tensions

# On attribue les noms aux deux listes contenues
t_brut,u_brut=(lecture('test_2.txt'))

# ************************************************
# Tracé des graphes bruts
# ************************************************
plt.plot(t_brut,u_brut)
plt.grid()
plt.title("etoiles doubles")
plt.show()

# ************************************************
# Tracé du graphes des fichiers limités
# ************************************************
# au début et à la fin ce n'est pas propre, donc
def freq_ech(L):
    delta_t=[]
    for i in range (1,len(L)):
        delta_t.append(L[i]-L[i-1])
    return delta_t

def valeur_moyenne(L):
    s=0
    for i in range(len(L)):
        s=s+L[i]
    Moyenne=s/len(L)
    return Moyenne
```

```
T0=valeur_moyenne(freq_ech(t_brut))/1000

deb=30
fin=2500
t=np.arange(fin-deb)*T0 # on reconstruit le vecteu
u=u_brut[deb:fin] # on recopie le nouveau fichier

plt.plot(t,u)
plt.grid()
plt.title("etoiles doubles")
plt.show()

# ************************************************
# Analyse FFT
# ************************************************
from scipy import fftpack
te=2*T0
fe=1/te
# signal fréquentiel : on divise par la taille du
fourier = fftpack.fft(u)/np.size(u)
# axe fréquentiel:
axe_f = np.arange(0.,(fin-deb))/fe
# On plot
plt.plot(axe_f,np.abs(fourier))
plt.axis([0,6,0,40])
plt.xlabel('axe frequentiels en Hertz')
plt.show()

# ************************************************
# Autre méthode
# ************************************************
# on calcule la valeur moyenne et on ne retient qu
# Ce qui revient à ajouter une diode dans le circu
def cent_pos(L):
```

```
    L_c=[]
    u_m=valeur_moyenne(L)
    for i in range(len(L)):
        if L[i] > u_m :
            L_c.append(L[i]-u_m)
        else :
            L_c.append(0)
    return L_c

u_c=cent_pos(u)
plt.plot(t,u_c)
plt.grid()
plt.title("après une diode")
plt.show()

# On applique un filtre
# Passe-bas
def passe_bas(signal,Fc=1):
    'filtre passe-bas équivalent à un premier ordr
    long=len(signal)
    y=[(signal[0]*2*np.pi*Fc*t[1])/(1+2*np.pi*Fc*t
    for n in range(1,long):
        y.append((signal[n]*2*np.pi*Fc*t[1]+y[n-1]
    return y

u1=passe_bas(u_c,Fc=1)
plt.plot(t,u1)
plt.grid()
plt.title("après passe_bas_1")
plt.show()

# On boucle n filtres du premier ordre
def cent(L):
    L_c=[]
    u_m=valeur_moyenne(L)
```

```
    for i in range(len(L)):
        L_c.append(L[i]-u_m)
    return L_c

uk=u_c[:]
for k in range (5):
    uk=cent(uk)
    uk=passe_bas(uk,Fc=1)

plt.plot(t,uk)
plt.grid()
plt.title("après passe_bas_k")
plt.show()

def points_d_annulation(L):
    par_zeros = [] # création d'une liste vide en
    for i in range(len(L)-1): # tous sauf le dern
        if L[i]*L[i+1] < 0 : # si les signes sont
            par_zeros.append(i) # on ajoute l'inde
    return par_zeros # on a fabriqué une liste d'

def pseudo_periode_par_zeros(L1,L2):
    index_zeros = points_d_annulation(L2) # dans
    # des points d'annulation de L2
    Liste_T=[] # on prépare une liste vide pour le
    for i in range(len(index_zeros)-1):
        t1,t2=index_zeros[i],index_zeros[i+1]
        Liste_T.append(2*(L1[t2]-L1[t1])) # on ajo
        # par soustraction des valeurs de L1[i],
        # c'est pour ça qu'il faut L1 !
    return Liste_T

print("la pseudo-période vaut :",round(valeur_moye
print("la fréquence associée vaut : f=",round(1/va
```

```
u_centre=cent(u)
print("la pseudo-période vaut :",round(valeur_moye
print("la fréquence associée vaut : f=",round(1/va
```