

TP 06

Tableaux, chaînes de caractères et fichiers

Proposition de corrigé

Activité 1 : Analyse d'un dipôle électrique

Q1:

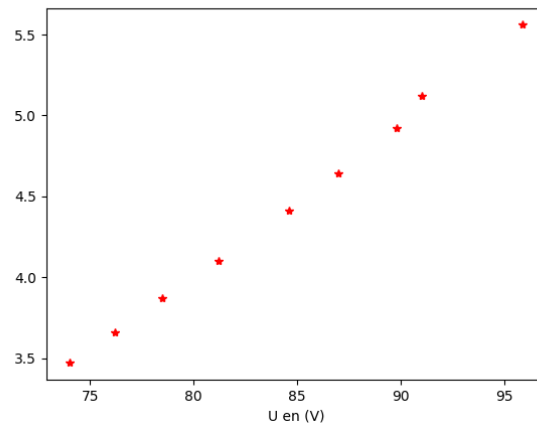
```
def lit_dipole(nom_De_fichier):  
    with open(nom_De_fichier, 'r') as f:  
        L=f.readlines()  
        n=len(L)  
        I=[]  
        U=[]  
        for i in range(1,int(n/2)):  
            I.append(float(L[i]))  
        for j in range(int(n/2)+1,n):  
            U.append(float(L[j]))  
        return I,U
```

Q2:

```
def tracer_dipole(I,U):  
    plt.clf()  
    plt.plot(I,U, 'r*')  
    plt.xlabel('I en (A)')  
    plt.ylabel('U en (V)')  
    plt.savefig('tp06_durif_q02.png')
```

Q3:

Il semble que l'on retrouve bien une relation linéaire entre la tension et l'intensité ce qui est bien conforme à la loi d'ohm.



Activité 2 : Lecture et analyses des caractères d'un texte

Q 4 :

La méthode `read` lit un caractère depuis la position courante, renvoie une chaîne.

La méthode `readline` lit le reste de la ligne depuis la position courante, renvoie une chaîne.

La méthode `readlines` lit toutes les lignes depuis la position courante, renvoie une liste de chaînes.

Q 5 :

Ce sont les caractères « tabulation » et « nouvelle ligne ».

Q 6 :

```
def carac(nom_de_fichier):
    """Renvoie une liste contenant le nombre de caractères
    de chaque ligne de nom_de_fichier"""
    with open(nom_de_fichier, 'r', encoding='utf8') as f:
        lignes = f.readlines()
    return [len(x.strip('\n').strip('\t')) for x in lignes]
```

Q 7 :

```
def somme_carac(nom_de_fichier):
    """Renvoie la somme nombre de caractères
    du texte entier"""
    L=carac(nom_de_fichier)
    S=0
    for x in L:
        S+=x
    return S
```

Q 8 :

```
def compte_carac(carac, nom_de_fichier):
    '''renvoie pour un caractère de l'alphabet son nombre d'occurrence sans tenir
    compte de la casse contenu dans le fichier nom_de_fichier'''
    with open(nom_de_fichier, 'r', encoding='utf8') as f:
        ligne='_'
        S=0
        while ligne!='':
            ligne = f.readline().lower()
            for c in ligne:
                if c==carac.lower():
                    S+=1
        return S
```

On peut utiliser une autre version plus courte.

```
def compte_carac2(carac,nom_de_fichier):
    '''renvoie pour un caractère de l'alphabet son nombre d'occurrence sans tenir
    compte de la casse contenu dans le fichier nom_de_fichier'''
    with open(nom_de_fichier,'r',encoding='utf8') as f:
        texte=f.read()
        return texte.lower().count(carac)
```

Q9:

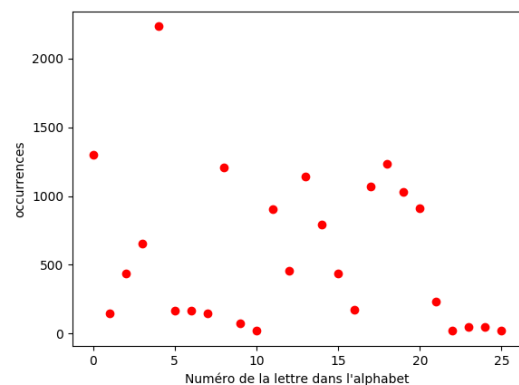
```
def stat_carac(nom_de_fichier):
    '''renvoie une liste de 26 éléments donnant le nombre d'occurrences de chaque
    lettre de l'alphabet contenu dans le texte nom_de_fichier sans tenir compte
    de la casse'''
    alphabet='abcdefghijklmnopqrstuvwxyz'
    occurrences=[]
    for car in alphabet:
        occurrences.append(compte_carac(car,nom_de_fichier))
    return occurrences
```

Q10:

Il faut importer ces modules et fonctions :

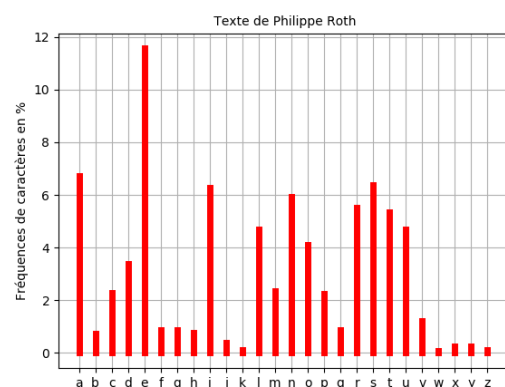
```
import matplotlib.pyplot as plt
from numpy import arange
```

```
def tracer_occurrences(nom_de_fichier):
    ''' trace en fonction du numéro de la
    lettre dans l'alphabet son
    occurrence dans le fichier \pyv\
    nom_de_fichier'''
    y=stat_carac(nom_de_fichier)
    plt.clf()
    plt.plot(y, 'ro')
    plt.xlabel("Numéro de la lettre dans
    l'alphabet")
    plt.ylabel('occurrences')
    plt.savefig('tp06_vosnoms_q10.png')
```



Q11:

```
def tracer_stat_occurrences(
    nom_de_fichier):
    y=stat_carac(nom_de_fichier)
    S=somme_carac(nom_de_fichier)
    plt.clf()
    for i,yi in enumerate(y):
        plt.plot([i,i],[0,100*yi/S], 'r-',
        linewidth=5)
    plt.ylabel('Fréquences de caractères
    en %')
    plt.grid()
    plt.title(" Texte de Philippe Roth",
    fontsize=10)
    plt.xticks(arange(26),tuple(
    alphabet_liste))
    plt.savefig('tp06_vosnoms_q11.png')
```



Q12:

```
def stat_carac_wikipedia(nom_de_fichier):
    '''à partir du fichier nom_du_fichier contenant les statistiques de fréquences de
    caractères renvoie une liste donnant en fonction de la position de la lettre
    dans l'alphabet sa fréquence en %.'''
```

```
occurences=[]
with open(nom_de_fichier, 'r', encoding='utf8') as f:
    ligne = f.readline()
    while ligne!='':
        occurences.append(float(ligne.strip('\n').split(';')[1]))
        ligne = f.readline()
return occurences
```

Q 13:

