TP 06

Tableaux, chaines de caractères et fichiers

Proposition de corrigé

Activité 1 : Analyse d'un dipôle électrique

Q 1 :

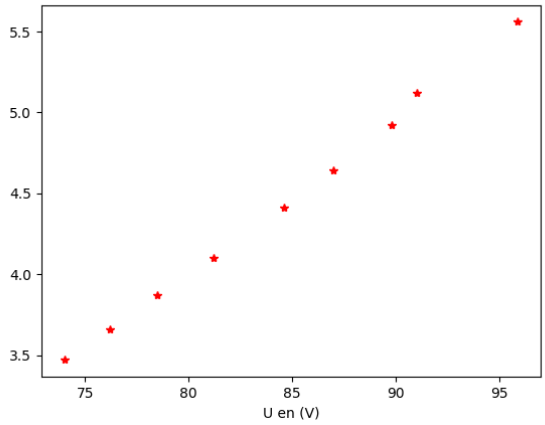
```
def lit_dipole(nom_De_fichier):
    with open(nom_De_fichier,'r') as f:
        L=f.readlines()
        n=len(L)
        I=[]
        U=[]
        for i in range(1,int(n/2)):
            I.append(float(L[i]))
        for j in range(int(n/2)+1,n):
            U.append(float(L[j]))
        return I,U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

Q 2 :

```
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')
```

Q 3 :

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 d'un texte

Q 4 :
La méthode read lit un caractère depuis la position courante, renvoie une chaine.
La méthode readline lit le reste de la ligne depuis la position courante, renvoie une chaine.
La méthode readlines lit toutes les lignes depuis la position courante, renvoie une liste de chaines.
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',) as f:
        lignes = f.readlines()
    return [len(x.strip('\n')) for x in lignes]
```

Q 7 :

```
def compte_carac(carac,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


def compte_carac2(carac,nom_de_fichier):
    with open(nom_de_fichier,'r',encoding='utf8') as f:
        texte=f.read()
        return texte.lower().count(carac)
```

Q 8 :

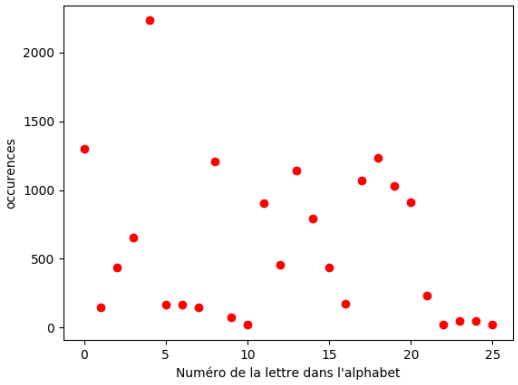
```
def stat_carac(nom_de_fichier):
    alphabet='abcdefghijklmnopqrstuvwxyz'
    occurences=[]
    for car in alphabet:
        occurences.append(compte_carac(car,nom_de_fichier))
    return occurences
```

Q 9 :
Il faut importer ces modules et fonctions :

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

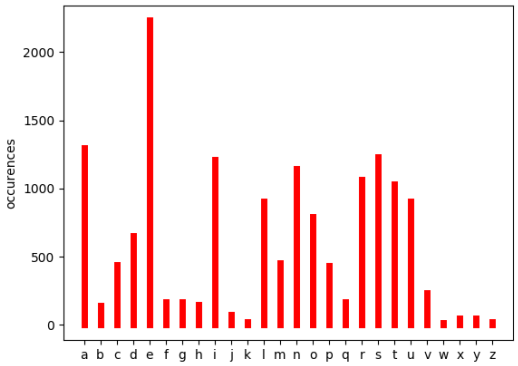
Méthode 1 : graphe

```
def trace_stat_carac(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('occurences')
    plt.savefig('graphe_occurences.png')
```



Méthode 2 : histogramme

```
alphabet='abcdefghijklmnopqrstuvwxyz'
alphabet_liste=len(alphabet)*[]
for c in alphabet:
    alphabet_liste.append(c)
def trace_stat_carac_bis(nom_de_fichier):
    y=stat_carac(nom_de_fichier)
    plt.clf()
    for i,yi in enumerate(y):
        plt.plot([i,i],[0,yi],'r-',↙
            linewidth=5)
    plt.ylabel('occurences')
    plt.xticks(arange(26),tuple(↙
        alphabet_liste))
    plt.savefig('histo_occurences.png')
    plt.show()
```



Activité 3 : Résultats de l'Embrunman