

```

# TP 10 corrigé.py

## TP 10 corrigé

## exo 1
# q1
def premier_dernier(texte):
    return texte[0], texte[-1]

# q2
def presence(texte, lettre):
    for x in texte :
        if x == lettre :
            return True
    return False

# q3
def trois_lettres(texte):
    return texte[1]+texte[3]+texte[4]

# q4
def extrait(texte,L):
    res = ""
    for k in L :
        res = res + texte[k]
    return res

# q5
def premier_mot(texte):
    res = ""
    for x in texte:
        if x!=" ":
            res = res + x
        else :
            return res

## exo 2
# q1
def frequence(texte, lettre):
    compt = 0
    for x in texte :
        if x == lettre:
            compt = compt + 1
    return compt/len(texte)

# q2
import string
alphabet = string.ascii_lowercase

# q3
def list_frequence(texte):
    L = []
    for lettre in alphabet:
        L.append(frequence(texte,lettre))
    return L

```

```
# q4
texte = "ceci est un texte sans aucun accent ni apostrophe ni majuscule
ni ponctuation pour ceux qui sont trop paresseux pour les utiliser il
est toutefois mieux de ne pas faire pareil dans une copie de
mathematique ou de biologie pour la suite je vais utiliser certains
mots qui utilisent normalement des accents sans les mettre pour me
faciliter la chose dans la recherche de mots et de phrases voici
quelques betises qui nont pas de liens entre elles simplement pour
augmenter le nombre de mots du texte et avoir une meilleure idee des
frequences des lettres j'aime les tigres car les tigres cest magnifique
il faudrait aller trouver des phrases toutes faites sur internet pour
la suite mais je nai pas envie de modifier tous les mots avec des
accents pour ce texte alors je vais me contenter de raconter des
nounouilleries sans interet d'ailleurs vous devriez arreter de lire cela
et vous concentrer sur lexercice a faire parce que franchement cest pas
serieux non non non vous perdez du temps sur le tp et monsieur fahel ne
va pas etre content quant a mr caillaud peutetre meme quil vous
grondera tellement il est mechant bref je crois que ce texte est assez
long quoique je dois bidouiller un peu car il manque des a et des i
dapres les statistiques de wikipedia ahaha mais ca ca va pas etre
facile tant pis"
```

```
absi = [x for x in alphabet]
ordo = list_frequence(texte)
import matplotlib.pyplot as plt
plt.bar(absi,ordo)
plt.show()
```

```
# q5
# faj hjxfw (c'est le titre de l'exo !)
```

```
# q6
# c'est (k+5)%26
```

```
# q7
def indice(lettre):
    for k in range(len(alphabet)):
        if alphabet[k]==lettre:
            return k
```

```
# q8
def codage(texte,d):
    res = ""
    for lettre in texte:
        if lettre != " ":
            k = indice(lettre)
            new_indice = (k+d)%26
            new_lettre = alphabet[new_indice]
        else :
            new_lettre = " "
        res = res + new_lettre
    return res
```

```
##
```

```
# q9
mystere = "wvcztzkrkzfej mflj rmvq kiflmv cr sfeev ivgfejv tvjk kifg
jlgvi vk avjgviv hlv mflj ve vkvj wzvi szve fe grjjv r cvof urgivj"
```

```
absi2 = [x for x in alphabet]
ordo2 = list_frequency(mystere)
plt.bar(absi2,ordo2)
plt.show()
```

on constate alors que c'est le v qui est la lettre la plus employée. Comme indice("v") vaut 21 et que indice("e") vaut 4, cela veut dire que le décalage utilisé pour coder le message est de 17.
pour le décoder, on utilise donc un décalage de -17 (ou de 9 = 26-17).

```
mystere_decode = codage(mystere, -17)
```

exo 3

```
# q1
```

```
import random as rd
```

```
def alea(n):
    L = ["A", "T", "C", "G"]
    return [random.choice(L) for k in range(n)]
```

```
# q2
```

```
def adn_to_arm(L):
    for k in range(len(L)):
        if L[k]=="A":
            L[k]="U"
        elif L[k]=="T":
            L[k]="A"
        elif L[k]=="G":
            L[k]="C"
        else:
            L[k]="G"
    return L
```

```
# q3
```

```
def bases_to_codon(L):
    return L[0]+L[1]+L[2]
```

```
# q4
```

```
def arm_to_codons(L):
    codons = []
    n = len(L)
    nb_codons = n//3
    for k in range(nb_codons):
        sous_liste = [L[3*k],L[3*k+1],L[3*k+2]] # ou L[3*k:3*k+3]
        codons.append( base_to_codon(sous_liste) )
    return codons
```

```
# q5
```

```
list_codons =
['AGG', 'AGA', 'AGC', 'AGU', 'AAG', 'AAA', 'AAC', 'AAU', 'ACG', 'ACA', 'ACC', 'ACU
```

```
' , 'AUG' , 'AUA' , 'AUC' , 'AUU' , 'CGG' , 'CGA' , 'CGC' , 'CGU' , 'CAG' , 'CAA' , 'CAC' , 'CAU' , 'CCG' , 'CCA' , 'CCC' , 'CCU' , 'CUG' , 'CUA' , 'CUC' , 'CUU' , 'UGG' , 'UGA' , 'UGC' , 'UGU' , 'UAG' , 'UAA' , 'UAC' , 'UAU' , 'UCG' , 'UCA' , 'UCC' , 'UCU' , 'UUG' , 'UUA' , 'UUC' , 'UUU' , 'GGG' , 'GGA' , 'GGC' , 'GGU' , 'GAG' , 'GAA' , 'GAC' , 'GAU' , 'GCG' , 'GCA' , 'GCC' , 'GCU' , 'GUG' , 'GUA' , 'GUC' , 'GUU' ]
```

```
list_aa =  
['Arg' , 'Arg' , 'Ser' , 'Ser' , 'Lys' , 'Lys' , 'Asp' , 'Asp' , 'Thr' , 'Thr' , 'Thr' , 'Thr' , 'Met' , 'Ile' , 'Ile' , 'Ile' , 'Arg' , 'Arg' , 'Arg' , 'Arg' , 'Gln' , 'Gln' , 'His' , 'His' , 'Pro' , 'Pro' , 'Pro' , 'Pro' , 'Leu' , 'Leu' , 'Leu' , 'Leu' , 'Trp' , 'STOP' , 'Cys' , 'Cys' , 'STOP' , 'STOP' , 'Tyr' , 'Tyr' , 'Ser' , 'Ser' , 'Ser' , 'Ser' , 'Leu' , 'Leu' , 'Phe' , 'Phe' , 'Gly' , 'Gly' , 'Gly' , 'Gly' , 'Glu' , 'Glu' , 'Asp' , 'Asp' , 'Ala' , 'Ala' , 'Ala' , 'Ala' , 'Val' , 'Val' , 'Val' , 'Val' , 'Val' , 'Val' ]
```

```
def arnm_to_aa(L):  
    codons = arnm_to_codons(L)  
    aa = []  
    for x in codons :  
        for k in range(len(list_codons)):  
            if x==list_codons[k]:  
                aa.append(list_aa[k])  
    return aa
```