# I Occurrence d'une lettre dans un message

1. `Travail par écrit :`
   on considère l'algorithme suivant

```
Algorithme A1
Début ...
1|   Saisir message
2|   n ← longueur(message)
3|   i ← 0
4|   cpt ← 0
5|   Tant que i<n
6|       Si message[i]="a" alors
7|           cpt ← cpt+1
8|       Fin Si
9|   ..............
10|  Fin Tant que
11|  Afficher cpt
Fin
```

   (a) Lister les variables utilisées ainsi que leur type : `message` de type *string* (chaîne de caractères), `n` de type *int* (entier), `cpt` de type *int*.

   (b) Il manque une instruction à la ligne 9. Laquelle ? $i \leftarrow i+1$ (incrément de la boucle *tant que*)

   (c) Quel est le rôle de cet algorithme ? Il permet de comptabiliser le nombre de 'a' dans la chaîne de caractères `message`.

   (d) Faire fonctionner cet algorithme avec :
   - message = « J'aime les ananas ! ». `L'algorithme affichera 4`
   - message = « Avez vous appris votre cours ? ». `L'algorithme affichera 1` : le 'A' en majuscule n'est pas compté.

   (e) Quelle modification pourrait-on faire à la ligne 6 afin de prendre en compte le fait que la lettre "a" peut être écrite en majuscule ? Par exemple : ... *message*[*i*]="a" ou *message*[*i*]="A" ...

   (f) Proposer une modification $A2$ de l'algorithme $A1$ pour qu'il affiche sous forme de tableau, le nombre d'occurrence de toutes les lettres de l'alphabet.
   `Il suffit d'appliquer l'algorithme A1 à toutes les lettres de l'alphabet et de constituer une liste qui se remplit du nombre d'apparition de toutes les lettres de l'alphabet sont apparues dans le` *`message`*. Voir les `fonctions` dans la question suivante.

2. `Travail sur poste informatique :`

   (a) Algorithme $A1$ en Python :

Une correction

```
## fonctions
#
def A1(message,lettre):
    n=len(message)
    i=0
    cpt=0
    while i<n:
      if message[i] in [lettre,lettre.upper()] :
         cpt+=1
      i+=1
    return cpt
#
## fin fonctions
#
## L'appel de la fonction A1("J'aime les ananas !","a") renvoie 4
#
```

   (b) Algorithme $A2$ en Python :

Une correction

```
## fonctions
#
alphabet="abcdefghijklmnopqrstuvwxyz"
esponc=[" ",",",";",":",".","!","?",chr(39)]
#
def A2(message):
    occ=[]
    for lettre in alphabet:
      occ.append(A1(message,lettre))
    occmax=max(occ)
    lmax=alphabet[occ.index(occmax)]
    return occ,lmax
#
## fin fonctions
#
## L'appel de la fonction A2("Bonjour, comment allez-vous?")
renvoie ([1,1,1,0,2,0,0,0,0,1,0,2,2,2,4,0,0,1,1,1,2,1,0,0,0,1],"o")
#
```

(c) Modifier ce dernier programme pour qu'il affiche la lettre la plus fréquente du message. Si plusieurs lettres ont le même nombre d'occurrences, on fera apparaître la première dans l'ordre alphabétique : déjà traité dans la question précédente : fonction A2.

# II Codage affine

1. `Travail par écrit` : Codage César.
   on considère l'algorithme suivant

**Algorithme** $B1$
**Début**
1| *alphabet* ← "abcdefghijklmnopqrstuvwxyz"
2| **Saisir** *mot*
3| *nouveaumot* ← ""
4| **Pour** *i* **allant de** 1 **à** longueur(*mot*)
5| *indice* ← position dans l'alphabet de *mot*[i]
6| *nouvelindice* ← (*indice* + 3)%26
7| *nouveaumot* ← *nouveaumot* + *alphabet*[*nouvelindice*]
8| **Fin Pour**
9| **Afficher** *nouveaumot*
**Fin**

on pourra dans cette partie utiliser la correspondance suivante : $a \to 0$, $b \to 1$, $c \to 2$, $\ldots$, $z \to 25$.

(a) Lister les variables utilisées ainsi que leur type : `alphabet`, `mot` et `nouveaumot` de type *string* (chaîne de caractères), `indice` de type *int* (entier), `nouvelindice` de type *int*.

(b) Quel est le rôle de cet algorithme ? Il affiche une chaîne de caractères (`nouveaumot`) dans laquelle chaque caractère est décalé de trois lettres par rapport au `mot` saisi par l'utilisateur.

(c) *mot*=**"bonjour"** donne *nouveaumot*=**"erqmrxu"**

(d) Proposer un modification de l'algorithme de l'algorithme $B1$ pourqu'il puisse gérer des messages avec espaces ou ponctuation :

Une correction

```
## fonctions
#
alphabet="abcdefghijklmnopqrstuvwxyz"
esponc=[" ",",",";",":",".","!","?",chr(39)]
#
def B1(mot,decalage):
    nveaumot=""
    for i in range(len(mot)):
      if mot[i] not in esponc:
         nouvelindice=(alphabet.index(mot[i])+decalage)%26
         nveaumot+=alphabet[nouvelindice]
    return occ,lmax
      else:
         nveaumot+=mot[i]
    return nveaumot
#
## fin fonctions
#
## L'appel de la fonction B1("bonjour!",3)
renvoie "erqmrxu!"
#
```

(e) Proposer une modification de l'algorithme $B1$ pour qu'il décode le message suivant, crypté avec le codage César : **« dx uhyrlu »** : B1("dx uhyrlu",−3) décodera pour donner "au revoir".

2. `Travail sur poste informatique : Décalage inconnu`

(a) Implémenter et tester l'algorithme $B1$ : voir plus haut.

(b) On considère une nouvelle méthode pour crypter un message. On décale toujours les lettres mais pas forcément de 3 rangs. On considère le message suivant, codé avec un décalage inconnu :

**« ath bpiwtbpixfjth, r'thi qxtc ! »**

i. En utilisant la partie A, analyser les fréquences d'apparition de chacune des lettres du message.

ii. Modifier le programme précédent pour qu'il repère la lettre la plus fréquente, en déduire le décalage qui a pu être appliqué pour coder le message, et décoder le message.