PÔLE CHATEAUBRIAND JOLIOT-CURIE

Fiche 07 Corrigé - Début des listes et parcours des structures indicées

Informatique - Syntaxe et algorithmes / c5c4-828739

1. Donner l'affichage réalisé par les trois codes ci-dessous.

```
1 L = "abcd"
2 for e in L:
3     print(e)
```

```
1 L = "abcd"
2 n = len(L)
3 for i in range(n):
4     print(i)
```

```
1 L = "abcd"
2 n = len(L)
3 for i in range(n):
4     print(L[i])
```

Solution : Les premier et troisième codes affichent les éléments de `L` :

a
b
c
d

Le deuxième code affiche les indices des éléments de `L` :

0
1
2
3

Soit un n-uplet `L` ne contenant que des nombres (entiers et flottants).

2. Ecrire un script définissant une variable `m` contenant la valeur minimale de `L`, et affichant `m`.

Solution :

```
1 m = L[0] # initialisation avec le premier élément de L
2 for e in L: # parcours des éléments
3     if e < m:
4         m = e
5 print(m)
```

3. Ecrire une fonction `mini(S)` retournant l'indice de l'élément le plus petit de `S`. `S` est une liste ou un n-uplet. Ecrire le script appelant cette fonction et qui affiche la valeur minimale (pas l'indice) de `L`.

Solution :

```
1 def mini(S):
2     imini = 0 # initialisation avec la première valeur
3     n = len(S)
4     for i in range(1, n): # parcours des indices
5         if S[i] < S[imini]:
6             imini = i
7     return imini
8
9 print(L[mini(L)]) # mini(L) retourne un indice
```

Attention à ne pas confondre les parenthèses et les crochets. Un nom d'objet suivi d'une parenthèse ouvrante correspond à un appel à la fonction : `print(`, `mini(`, `len(`... Si le nom est suivi d'un crochet, il s'agit d'un accès à une structure indicable : `S[`, `L[`...

Ici, `S(` donne une erreur car `S` n'est pas une fonction. De même, `mini[` n'a pas de sens car `mini` est une fonction et n'a pas d'indice.

4. On suppose que `n` et `L` sont définis ainsi :
   `n = 10`
   `L = [0] * 10`
   `L` est donc une liste contenant n entiers 0. Écrire les deux lignes nécessaires pour modifier, ensuite, la liste afin qu'elle contienne des flottants de 1 à 10.

Solution : Avec une boucle `for in`, cela donne

```
for i in range(n):
    L[i] = i + 1
```

5. Ecrire une fonction `occurence(L, x)` retournant le nombre de fois ou l'élément `x` est présent dans la liste `L`.

Solution :

```
def occurence(L, x):
    nb = 0 # initialisation d'un compteur
    for e in L: # parcours des valeurs
        if e == x:
            nb = nb + 1
    return nb
```

La fonction `occurence` précédente fonctionne aussi si `L` est une chaîne de caractères et `x` un caractère. Soit le script suivant :

```
texte = "Voyez-vous, monsieur le ministre, il y a bien des sujets
sur lesquels je suis en desaccord avec Dumbledor... Mais il faut lui
reconnaitre qu'il ne manque pas de sytle..."

lettres = ['v', 'z', 's', 'j']
occurences = [0] * len(lettres) # crée la liste [0, 0, 0, 0]
```

6. Ecrire un script qui modifie la liste `occurences` de tel sorte que `occurences[i]` contienne le nombre d'occurence de `lettres[i]` dans `texte`. Utiliser la fonction `occurence`.

Solution :

```
n = len(lettres) # nombre de lettres à rechercher
for i in range(n): # parcours des indices de lettres
    nb = occurence(texte, lettres[i])
    occurences[i] = nb
```