TP2 Python
Listes

Une liste est un tableau unidimensionnel modifiable qui contient une suite de valeurs de type int, float, str ou bool.
Ces valeurs sont séparées par des virgules et placées entre crochets.

Exemple
```
animaux = ["girafe","tigre","gorille","souris"]
```
Exemple
```
taille=[500,100,175,5]
```
Exemple
```
mixte=["girafe",500.4,"tigre",100,"gorille",175.7,"souris",5].
```

I)Opérations sur les listes

I.1)Sélection d'un élément dans une liste

L[k]

renvoie l'élément d'indice k de la liste L.

 **le premier élément de la liste porte l'indice zéro**.

Exemple
```
L=["âne","chèvre","fleur","wagon"]
```
Les instructions L[0], L[1], L[2] et L[3] renvoient respectivement les mots âne, chèvre, fleur et wagon.

Exemple
```
L=[5,10,7]
L[1]=3
print(L)
```
Ce programme remplace l'élément d'indice 1 de la liste par 3, puis affiche la liste modifiée : [5,3,7].

Remarque
Par convention, l'instruction L[-1] renvoie le dernier élément de la liste L.

I.2)Sélection d'éléments consécutifs d'une liste

L[i :j]

renvoie les éléments de la liste L dont les indices sont compris entre i et j-1.

Exemple
```
L=[10,45,20,30,100,8,6]
print(L[2:5])
```
Le programme affiche la liste : [20,30,100].

I.3)Ajout d'un élément en fin de liste

L.**append**(a)

ajoute la valeur a, tout à la fin de la liste L.

Exemple

```
L=[1,3,8,5]
L.append(12)
```

A l'issue, la liste vaut : [1,3,8,5,12].

I.4)Suppression d'un l'élément dans une liste

L.**pop**(k)

supprime l'élément d'indice k de la liste L et renvoie cet élément.

Exemple :

```
L=[10,20,30,40]
x=L.pop(2)
print(x)
print(L)
```

Python supprime l'élément d'indice 2 de L, c'est-à-dire 30, affiche cette valeur ainsi que la nouvelle liste modifiée : L=[10,20,40].

I.5)Longueur d'une liste

len(L)

compte le nombre d'éléments de la liste L.

Exemple

```
maths=["théorême","constante","intégrale","moyenne"]
print(len(maths))
```

A l'écran s'affiche la valeur 4.

I.6)Concaténation de deux listes

On peut concaténer deux listes L et M, c'est-à-dire mettre ces deux listes bout à bout, en utilisant le symbole d'addition.
On obtient alors une nouvelle liste N qu'on définit par : N=L+M.

Exemple

```
L=[2,7,8]
M=[5,12]
N=L+M
print(N)
```

Le programme renvoie la liste : [2,7,8,5,12].

I.7)Réplication d'une liste

On peut aussi répliquer (ou répéter) une liste.

Exemple

```
L=[0]
print(L*4)
```

Le programme affiche la liste : [0,0,0,0]

Exemple

```
L=[1,2,3]
M=L*4
print(M)
```

Le programme affiche la liste [1,2,3,1,2,3,1,2,3,1,2,3]

II)Les classiques

On donne ici quelques méthodes très classiques à connaître.

II.1)Test d'appartenance

Pour tester si un élément a appartient à une liste L, il y a deux méthodes.

• une méthode directe :

if a **in** L **:**

Exemple :

```
L= [10, 20, 30, 40,20]
if 20 in L:
    print("20 est dans la liste")
```

• une méthode indicielle :

if L[k]==a **:**

Exemple :

```
L=[10,20,30,40,20]
for k in range(5):
    if L[k]==20:
        trouve=True
if trouve==True:
    print("20 est dans la liste")
```

Remarque

Le programme ci-dessous serait moins satisfaisant. Pourquoi ?

```
L=[10,20,30,40,20]
for k in range(5):
    if L[k]==20:
        print("20 est dans la liste")
```

II.2)Parcours d'une liste

Pour parcourir les éléments d'une liste L, il y a deux méthodes.

• une méthode directe :

for a **in** L **:**

Exemple :

```
L=["pomme","banane","orange"]
for a in L:
    print(a)
```

Python affiche : pomme, banane et orange.

• une méthode indicielle :

for k **in** **range**(**len**(L)) :

Exemple :

```
L=["pomme","banane","orange"]
for k in range(3):
    print(L[k])
```

Python affiche : pomme, banane et orange.

II.3)Création d'une liste par compréhension

C'est une méthode très rapide pour créer une liste.

L=[..... **for** k **in range**(...)]

Exemple

```
L=[2*k for k in range(5)]
```

On a alors : L=[0,2,4,6,8].

II.4)Création d'une liste par ajouts successifs

On commence par créer une liste vide, puis on la construit élément par élément.

L=[ ]
L.**append**(....)

```
L=[]
for i in range(1,6):
    L.append(i**2)
print(L)
```

Python affiche L=[1,4,9,16,25]

III)Exercices

Exercice 1★ ☆ ☆☆
Pour chacun des cas, explicitez tous les éléments de la liste.

1)L=[2**k for k in range(1,6)]

2)M=[k for k in range(20) if k%2==0]

3)N=[i*j for i in range(1,4) for j in range(1,5)]

Exercice 2★ ☆ ☆☆
Utilisez une compréhension de liste pour générer les listes ci-dessous.

1)L=[1,3,5,7,...,143]

2)M=[1,4,9,16,25, ...,100]

3)N=$\left[2, \frac{3}{2}, \frac{4}{3}, ..., \frac{100}{99}\right]$.

Exercice 3 ★ ☆ ☆☆
1)On considère la liste L= [3*k for k in range(100) if k%5==0].
(i)Expliciter tous les éléments de L. Vérifier en affichant cette liste dans la console.
(ii)Sans expliciter tous les éléments, créer et afficher la liste M formée de tous les éléments de L, sauf les 5 premiers et les 5 derniers.
2)Créer par compréhension la liste I des inverses des 100 premiers entiers naturels non nuls.

Exercice 4 ★ ☆ ☆☆
On considère la fonction `mystere` ci-dessous, d'argument une liste L.

```
def mystere(L):
    indices=[]
    for k in range(len(L)):
        if L[k]==2:
            indices.append(k)
    return indices
```

1)Expliquez ce que renvoie cette fonction.

2)Recopier le programme et appeler la fonction en prenant L = [1,2,8,7,4,5,6,7,2], puis L = [5,4,6,3,1,8].

Exercice 5 ★ ★ ☆☆
Ecrire une fonction d'en-tête `def nb_occurrences(a,L):` prenant comme paramètre un élément a et une liste L et renvoyant le nombre de fois où a est présent dans L, c'est-à-dire le nombre d'occurences de a.

Exercice 6 ★ ★ ☆☆
Ecrire une fonction d'en-tête `def rang(a,L):` prenant comme paramètre un élément a et une liste L et renvoyant le rang de la première occurence de a si a$\in$L et -1 sinon.

Exercice 7 ★ ★ ☆☆

1)Ecrire une fonction d'en-tête `def somme(L):` prenant comme paramètre une liste L de nombres et renvoyant la somme de ses éléments.

2)En déduire une fonction d'en-tête `def moyenne(L):` prenant comme paramètre une liste L de nombres et renvoyant la moyenne de ses éléments.

Exercice 8 ★ ★ ☆☆

1)Ecrire une fonction d'en-tête `def minimum(L):` prenant comme paramètre une liste L non vide de nombres et renvoyant son plus petit élément.

(On pourra réfléchir à ce qu'il faudrait modifier pour que la fonction retourne le plus grand des éléments ...)

2)Ecrire une fonction d'en-tête `def plusproche(L):` prenant comme paramètre une liste L et renvoyant l'écart minimum entre deux éléments distincts quelconques de L, c'est-à-dire le nombre :

$$\min_{0\leq i<j\leq len(L)-1} \big|L[i]-L[j]\big|$$

Indication : *utiliser la fonction `minimum` de la première question.*

Exercice 9 ★ ★ ☆☆

Problème : on souhaite supprimer d'une liste L les éléments qui valent a.
On construit alors une fonction d'arguments a et L qui retourne la nouvelle liste débarrassée des a.

1)Taper le programme ci-dessous :

```
def supprime(a,L):
    for k in range(len(L)):
        if L[k]==a:
            L.pop(k)
    return L
```

Sur la console, prendre a=1, L=[2,1,4,1,5], puis appeler la fonction par la commande `f(a,L)`.
La console renvoie le message d'erreur : *list index out of range.*
Expliquer pourquoi.

2)Pour ne plus avoir cette erreur, on utilise une boucle `while`. Compléter la fonction ci-dessous pour qu'elle résolve le problème.

```
def supprime(a, L):
    k=0
    while k<len(L):
        if L[k]==a:
            ...........
        else:
            ...........
    return L
```

3)On voit que modifier la liste L en cours de boucle peut être un peu dangereux. C'est pourquoi il est préférable de construire une nouvelle liste M. Compléter la fonction ci-dessous pour qu'elle réponde au problème.

```
def supprime(a, L):
    M=[]
    for k in range(len(L)):
        if L[k]..... :
            M.append(.......)
    return M
```

4)Une dernière façon, et c'est la plus rapide, est de construire la nouvelle liste par compréhension.
Compléter la fonction ci-dessous pour qu'elle réponde au problème.

```
def supprime(a, L):
    M=[x for x in L if ..........]
    return M
```

Exercice 10★ ★ ☆☆
Ecrire une fonction d'en-tête `envers(L)` : d'argument une liste L et retournant cette même liste, mais écrite à l'envers.
Par exemple, si L=[1,7,5], la fonction doit retourner [5,7,1].

Exercice 11★ ★ ★☆
Ecrire une fonction d'en-tête `tri(L)` : d'argument une liste L et retournant la liste des éléments de L écrits dans l'ordre croissant.

indication : on pourra en s'inspirant de l'exercice 8, commencer par ecrire une fonction d'en-tête indicemini(L) : qui retourne l'indice du plus petit élément de L.