



PYTHON diapo 12

Listes

Roblet²

dernière MAJ le 06/11/19

CE QU'IL FAUT SAVOIR

- Connaître le rang d'un élément dans une liste
 - Calculer la longueur de la liste « L » avec len(L)
 - Calculer le nombre d'apparitions de dans la liste « L » avec L.count()
 - SLICING en français : « saucissonnage »
 - AJOUT d'un élément
 - SUPPRESSION d'un élément
 - MODIFICATION d'un élément
 - CONCATÉINATION de listes
 - Ranger une liste dans l'ordre croissant avec L.sort() ou sorted(L)
 - Générer une liste de termes d'une suite numérique
 - Appliquer des instructions à des éléments d'une liste
-
- Exercices – tests

Diapo 12 Listes

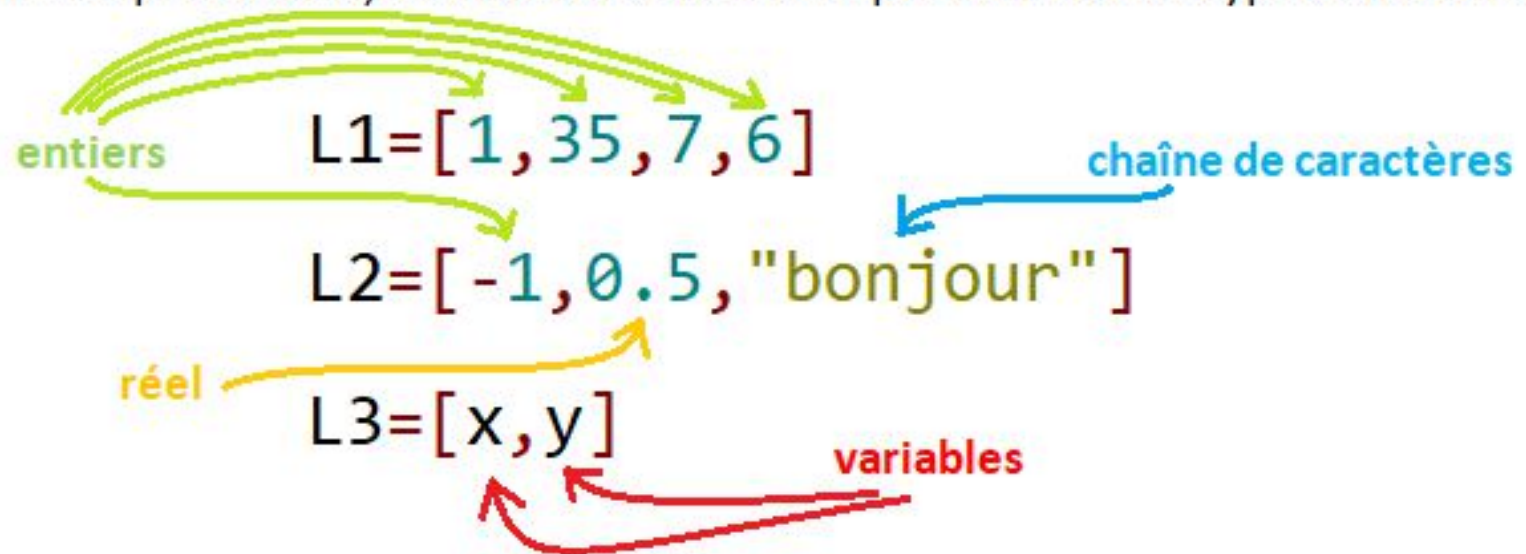
Une **liste** est une collection d'éléments rangés dans un ordre précis.

$L1 = [1, 35, 7, 6]$

$L2 = [-1, 0.5, \text{"bonjour"}]$

$L3 = [x, y]$

Comme on peut le voir, les éléments d'une liste peuvent être de types différents :



Diapo 12 Listes

De plus, le rang d'un élément est son indice. Le premier élément a pour indice 0, le deuxième élément a pour indice 1, etc.

Le premier élément de la liste L1 est noté L1[0].

Le deuxième élément de la liste L1 est noté L1[1].

...

L1 = [1, 35, 7, 6]
 L1[0] L1[1] L1[2] L1[3]

L2 = [-1, 0.5, "bonjour"]
 L2[0] L2[1] L2[2]

L3 = [x, y]
 L3[0] L3[1]

dans la console

```
>>> L2[1]
```

```
0.5
```

```
>>> L2[2]
```

```
'bonjour'
```

Diapo 12 Listes

Remarque importante sur le rang d'un élément

Python attribue aussi le rang **-1** au dernier élément de la liste.

L4 = [180, 240, 150, 290, 450]

0	1	2	3	4
-5	-4	-3	-2	-1

dans la console

```
>>> L4[4]
```

```
450
```

```
>>> L4[-1]
```

```
450
```

```
>>>
```

Diapo 12 Listes

Que va-t'il s'afficher dans la console ?

```
L5=[-4,3,8,-7,10,2,10]
```

dans la console

```
>>> L5[0]
```

```
...
```

```
>>> L5[1]
```

```
...
```

```
>>> L5[6]
```

```
...
```

```
>>> L5[-1]
```

```
...
```

```
>>> L5[-2]
```

```
...
```

Diapo 12 Listes

Que va-t'il s'afficher dans la console ?

```
L5=[ -4, 3, 8, -7, 10, 2, 10]
```

```
    0  1  2  3  4  5  6
```

```
   -7 -6 -5 -4 -3 -2 -1
```

dans la console

réponse

```
>>> L5[0]
```

```
-4
```

```
>>> L5[1]
```

```
3
```

```
>>> L5[6]
```

```
10
```

```
>>> L5[-1]
```

```
10
```

```
>>> L5[-2]
```

```
2
```

Diapo 12 Listes

len(L) calculer la longueur de la liste « L »

exemple L5=[-4,3,8,-7,10,2,10]

dans la console

```
>>> len(L5)
7
```

L.count(x) calculer le nombre d'apparitions de *x*
dans la liste « L »

exemple dans la console

```
>>> L5.count(10)
2
>>> L5.count(3)
1
>>> L5.count(11)
0
```


Diapo 12 Listes

SLICING en français : « saucissonnage »

exemple L5=[-4,3,8,-7,10,2,10]

dans la console

```
>>> L5[2:4]          >>> L5[-5:-2]
[8, -7]              [8, -7, 10]
>>> L5[2:]          >>> L5[2:2]
[8, -7, 10, 2, 10]  [] ← liste vide
>>> L5[:4]
[-4, 3, 8, -7]
```

L5 = [-4, 3, 8, -7, 10, 2, 10]

0	1	2	3	4	5	6	7
-4	3	8	-7	10	2	10	
-7	-6	-5	-4	-3	-2	-1	

Diapo 12 Listes

AJOUT d'un élément

exemple $L5 = [-4, 3, 8, -7, 10, 2, 10]$

$L.append(x)$ ajout de l'élément x à la fin de la liste L

dans la console

```
>>> L5.append(8)
>>> L5
[-4, 3, 8, -7, 10, 2, 10, 8]
```

$L.insert(k,x)$ insertion de l'élément x au rang k

dans la console

```
>>> L5.insert(4,-6)
>>> L5
[-4, 3, 8, -7, -6, 10, 2, 10]
```

$L5 = [-4, 3, 8, -7, 10, 2, 10]$

$L5 = [-4, 3, 8, -7, -6, 10, 2, 10]$

Diapo 12 Listes

SUPPRESSION d'un élément

exemple $L5 = [-4, 3, 8, -7, 10, 2, 10]$

`del L[k]` suppression de l'élément de rang k de la liste L

dans la console

```
>>> del L5[4]
>>> L5
[-4, 3, 8, -7, 2, 10]
```

$L5 = [-4, 3, 8, -7, \text{X}, 2, 10]$

Remarque: on peut aussi supprimer directement une partie de la liste

dans la console

```
>>> del L5[2:6]
>>> L5
[-4, 3, 10]
```

$L5 = [-4, 3, \text{X}, \text{X}, \text{X}, \text{X}, 10]$

Diapo 12 Listes

MODIFICATION d'un élément

exemple `L5=[-4,3,8,-7,10,2,10]`

dans la console

```
>>> L5[2]=11
>>> L5
[-4, 3, 11, -7, 10, 2, 10]

>>> L5[1]="bonjour"
>>> L5
[-4, 'bonjour', 11, -7, 10, 2, 10]
```

Remarque : on peut aussi modifier une partie de la liste

dans la console

```
>>> L5[2:5]=[0,0,1]
>>> L5
[-4, 3, 0, 0, 1, 2, 10]

>>> L5[2:5]=[0,0]
>>> L5
[-4, 3, 0, 0, 2, 10]
```

On remarquera que l'on a remplacé un segment de 3 éléments par un segment de 2 éléments, la longueur de la liste a donc été diminuée de 1.

Diapo 12 Listes

CONCATÉINATION

« Concaténer » des listes signifie les rassembler.

Avec Python on peut concaténer des listes grâce à l'opérateur « + ».

exemple L5=[-4, 3, 8, -7, 10, 2, 10]
L6=[9, 5]

dans la console

```
>>> L5+L6
[-4, 3, 8, -7, 10, 2, 10, 9, 5]
      L5          L6

>>> L6+L5
[9, 5, -4, 3, 8, -7, 10, 2, 10]
  L6          L5

>>> L6+[2,2,0]
[9, 5, 2, 2, 0]
  L6

>>> [18,20,15]+[29,22]
[18, 20, 15, 29, 22]
```

L.SORT() ou SORTED(L) ?

Dans les 2 cas, les éléments de la liste sont rangés dans l'ordre croissant.

exemple L5 = [-4, 3, 8, -7, 10, 2, 10]

Comparons :

dans la console

```
>>> sorted(L5)
[-7, -4, 2, 3, 8, 10, 10]
>>> L5
[-4, 3, 8, -7, 10, 2, 10]
```

Avec « sorted(L5) », la liste L5 n'a pas été modifiée.

```
>>> L5.sort()
>>> L5
[-7, -4, 2, 3, 8, 10, 10]
```

Avec « L5.sort() », la liste L5 a été modifiée.

Remarque si l'on veut ranger une liste dans l'ordre décroissant :

dans la console

```
>>> sorted(L5, reverse=True)
[10, 10, 8, 3, 2, -4, -7]
>>> L5
[-4, 3, 8, -7, 10, 2, 10]
```

L5 n'a pas été modifiée.

OU

```
>>> L5.sort()
>>> L5.reverse()
>>> L5
[10, 10, 8, 3, 2, -4, -7]
```

L5 a été modifiée.

OU

```
>>> L5.sort(reverse=True)
>>> L5
[10, 10, 8, 3, 2, -4, -7]
```

Diapo 12 Listes

GÉNÉRER UNE LISTE de termes d'une suite

`L=[expression de U_n for n in range(k)]`

Crée la liste des U_n
pour n allant de 0 à ($k-1$).

exemple

dans la console

```
>>> L6=[3*n+1 for n in range(11)]  
>>> L6  
[1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31]
```



U_0



U_1



U_2



U_3



U_4

...



U_{10}

$$U_n = 3n + 1$$

APPLIQUER DES INSTRUCTIONS à des éléments d'une liste

for `k` in `liste` : Parcourt les éléments de la liste et applique
 `instruction(s)` les instructions pour chacun d'eux.

exemple

```
L5=[-4,3,8,-7,10,2,10]
for k in L5:
    k=k+1
    print(k)
print(L5)
```

dans la console

```
>>>
-3
4
9
-6
11
3
11
[-4, 3, 8, -7, 10, 2, 10]
```

On remarque que la liste L5 n'a pas été modifiée.

Traduire les commandes Scratch suivantes en commandes Python :

L7

1	12
2	14
3	20
4	16
5	15
6	18
7	23

+ longueur: 7

élément 1 de L7

longueur de L7

ajouter 18 à L7

supprimer l'élément 4 de la liste L7

remplacer l'élément 3 de la liste L7 par 20

insérer 24 en position 5 de la liste L7

mettre nombre de 15 à 0

mettre k à 1

répéter longueur de L7 fois

si élément k de L7 = 15 alors

ajouter à nombre de 15 1

ajouter à k 1

dire nombre de 15


Traduire les commandes Scratch suivantes en commandes Python :

réponse



A Scratch list widget labeled 'L7' containing the numbers 12, 14, 20, 16, 15, 18, and 23. The list is displayed as a vertical stack of orange buttons. Below the list, it shows '+ longueur: 7'.

```
L7=[12,14,20,16,15,18,23]
```

élément 1 de L7	L7[0]
longueur de L7	len(L7)
ajouter 18 à L7	L7.append(18)
supprimer l'élément 4 de la liste L7	del L7[3]
remplacer l'élément 3 de la liste L7 par 20	L7[2]=20
insérer 24 en position 5 de la liste L7	L7.insert(4,24)
 <p>A Scratch script starting with 'mettre nombre de 15 à 0', followed by a loop 'répéter longueur de L7 fois'. Inside the loop, there is an 'if' block 'si élément k de L7 = 15 alors' containing 'ajouter à nombre de 15 1' and 'ajouter à k 1'. The script ends with 'dire nombre de 15'.</p>	L7.count(15)

Diapo 12 Listes

Quelle ligne de commande doit-on taper pour passer de la liste actuelle à la liste voulue ?

liste actuelle

liste voulue

L8=[10,20,30,40]	→	L8=[10,20,30,50]
L9=[0.1,0.3,0.1,0.2,0.2]	→	L9=[0.1,0.3,0.1,0.2,0.2,0.4]
L10=[1,1,0]	→	L10=[1,5,1,0]
L11=[92,95,94,97,93,92,98]	→	L11=[92,92,93,94,95,97,98]
L12=[-1,17,23,48,5]	→	L12=[-1,23,48,5]
L12=[-1,17,23,48,5]	→	L12=[-1,48,5]

Diapo 12 Listes

Quelle ligne de commande doit-on taper pour passer de la liste actuelle à la liste voulue ?

liste actuelle

réponse

liste voulue

L8=[10,20,30,40]

→ L8[3]=50

→ L8=[10,20,30,50]

L9=[0.1,0.3,0.1,0.2,0.2]

→ L9.append(0.4)

→ L9=[0.1,0.3,0.1,0.2,0.2,0.4]

L10=[1,1,0]

→ L10.insert(1,5)

→ L10=[1,5,1,0]

L11=[92,95,94,97,93,92,98]

→ L11.sort()

→ L11=[92,92,93,94,95,97,98]

L12=[-1,17,23,48,5]

→ del L12[1]

→ L12=[-1,23,48,5]

L12=[-1,17,23,48,5]

→ del L12[1:3]

→ L12=[-1,48,5]

Diapo 12 Listes

Donner la commande en Python qui crée la liste des $U_n = 50 + 10 \times n$ pour n allant de 0 à 30.

Diapo 12 Listes

Donner la commande en Python qui crée la liste des $U_n = 50 + 10 \times n$ pour n allant de 0 à 30.

réponse

```
L13=[50+10*n for n in range(31)]
```