

Tranches

Fiche de synthèse

① EXTRAIRE UNE TRANCHE

Définition 1 (Tranche) Une **tranche** `L[début:fin]` est la partie d'une liste, d'une chaîne de caractères ou d'un tuple qui va de l'indice **début**, **inclus**, à l'indice **fin**, **exclu**. En anglais, on parle de *slicing*.

0	1	2	3	4	5	
a	b	c	d	e	f	<code>L[1:4]</code>
-6	-5	-4	-3	-2	-1	

```
L = ['a', 'b', 'c', 'd', 'e', 'f']
print(L[1:4])    # ['b', 'c', 'd']
print(L[:3])     # ['a', 'b', 'c']
print(L[3:])     # ['d', 'e', 'f']
```

`L[1:4]` prend les indices 1, 2 et 3 : elle contient $4 - 1 = 3$ éléments. Sans début, la tranche part du premier élément; sans fin, elle va jusqu'au dernier.

② COPIER UNE LISTE

`L[:]`, sans début ni fin, est une nouvelle liste qui contient tous les éléments : c'est une copie. Modifier la copie ne change pas `L`.

③ INDICES NÉGATIFS

```
print(L[-3:])    # ['d', 'e', 'f'] : 3 derniers
print(L[:-1])    # tous sauf le dernier
```

④ LE PAS

```
print(L[::2])    # ['a', 'c', 'e']
print(L[1::2])   # ['b', 'd', 'f']
print(L[::-1])   # de 'f' à 'a'
```

Dans `L[début:fin:pas]`, le **pas** est l'écart entre deux indices pris. Un pas négatif parcourt la séquence à l'envers : `L[::-1]` l'inverse.

⑤ ÉLÉMENT OU TRANCHE

Un indice seul donne un **élément** : `L[2]` vaut `'c'`. Une tranche donne toujours une **nouvelle séquence** du même type : `L[2:3]` vaut `['c']`, une liste d'un élément.

⑥ CHAÎNES ET TUPLES

```
mot = "Python"
print(mot[0:3])    # Pyt
print(mot[::-1])   # nohtyP
date = (14, 7, 1789)
print(date[:2])    # (14, 7)
```

La tranche d'une chaîne est une chaîne, celle d'un tuple est un tuple.

⑦ REMPLACER UNE TRANCHE

```
L = ['a', 'b', 'c', 'd', 'e', 'f']
L[2:4] = ['X', 'Y', 'Z']
print(L)        # ['a', 'b', 'X', 'Y', 'Z', 'e', 'f']
```

Les deux éléments d'indices 2 et 3 sont remplacés par trois éléments : la liste en compte maintenant sept. On ne peut pas remplacer la tranche d'une chaîne ou d'un tuple, qui sont immuables : `mot[0:1] = "P"` provoque l'erreur `TypeError`.

Tranche	Éléments
<code>L[i:j]</code>	des indices <code>i</code> à <code>j - 1</code>
<code>L[:j]</code>	les <code>j</code> premiers
<code>L[i:]</code>	tous sauf les <code>i</code> premiers
<code>L[-k:]</code>	les <code>k</code> derniers
<code>L[:k]</code>	un élément sur <code>k</code>
<code>L[::-1]</code>	tous, à l'envers

À retenir.

- Dans `L[début:fin]`, le début est inclus, la fin exclue.
- `L[:]` copie une liste; `L[::-1]` l'inverse.
- Une tranche est une nouvelle séquence du même type.
- Seule une liste permet de remplacer une tranche.