Python: Découper les chaînes de caractères

Le découpage ('*slicing*') de chaînes vous permet d'**extraire une sous-chaîne** en spécifiant une plage d'indices. Cela se fait en utilisant la notation `[début:fin]`.

Exploration

Objectif

Apprendre à **extraire une partie** d'une chaîne de caractères grâce au **slicing** (`[début:fin:pas]`).

Découpage de Chaînes

Le découpage ("***slicing***") de chaînes vous permet d'**extraire une sous-chaîne** en spécifiant une plage d'indices. Cela se fait en utilisant la notation `[début:fin]`. Par exemple :

```
chaine = "Python Programming"
sous_chaine = chaine[0:6]
print(sous_chaine)  # Affiche "Python"
```

IDÉES-CLÉS : Le slicing [début:fin] exclut l'indice fin. [:n] = du début à n-1. [m:] = de m jusqu'à la fin.

Donc:

```
texte = "Python"
print(texte[0:3])   # Affiche "Pyt"
```

👉 `texte[0:3]` signifie :

Prendre les caractères **DU 0 INCLUS** jusqu'au **3 exclu**.

Syntaxe complète

```
sequence[debut:fin:pas]
```

- **debut** : index de départ (inclus).
- **fin** : index d'arrêt (exclu).
- **pas** *(optionnel)* : nombre d'éléments à sauter à chaque étape.

Exemples de base

```
chaine = "Hello World"

print(chaine[0:5])   # "Hello" (du 0 au 4)
print(chaine[:5])    # "Hello" (début implicite = 0)
print(chaine[6:])    # "World" (jusqu'à la fin)
print(chaine[:])     # "Hello World" (copie complète)
```

💡 Quand on omet `début` ou `fin`, Python prend **le début ou la fin de la chaîne par défaut**.

Utiliser un pas

Le troisième paramètre (`pas`) permet de **sauter des éléments**.

```
chaine = "123456789"

print(chaine[::2])   # "13579" (1 sur 2)
print(chaine[1::2])  # "2468"  (en partant du 2e)
```

Challenge

Copie-colle cette chaine: `L5rEeuIusM0u EnA3A4te46z,wub ERDvh8hoDKVujOAsZVP ClcêsWwtP1He3Z4s4dh CeAl72MeEdVsg6h JtMmKI1eakAiyPWlCxOlioFeEr8uSEkrUApsvhw` et stocke-la dans une variable `message_secret`.

Essaye de l'afficher entièrement en trouvant le bon pas pour retrouver le message

décodé.

Indices négatifs

Les indices négatifs comptent **depuis la fin**.

```python
chaine = "Python"

print(chaine[-3:])   # "hon"   (3 derniers caractères)
print(chaine[:-3])   # "Pyt"   (tout sauf les 3 derniers)
print(chaine[-4:-1]) # "tho"   (du 4e à partir de la fin jusqu'à l'avant-dernier)
```

Inverser une chaîne

Utiliser un **pas négatif** pour lire à l'envers :

```python
chaine = "Python"
print(chaine[::-1])  # "nohtyP"
```

💡 `[::-1]` = tout, mais en sens inverse.

Ce qu'il faut retenir

Élément	Signification	Exemple	Résultat
`[a:b]`	de a (inclus) à b (exclu)	`"Hello"[1:4]`	`"ell"`
`[:b]`	du début à b (exclu)	`"Hello"[:4]`	`"Hell"`
`[a:]`	de a à la fin	`"Hello"[2:]`	`"llo"`
`[:]`	tout (copie)	`"Hello"[:]`	`"Hello"`
`[::2]`	un caractère sur deux	`"Hello"[::2]`	`"Hlo"`
`[::-1]`	inverse la chaîne	`"Hello"[::-1]`	`"olleH"`

Bonnes pratiques

- Le **slicing ne modifie jamais** la chaîne d'origine (les chaînes sont immuables).
- Utilise `len()` pour découper dynamiquement :

```
milieu = len(chaine) // 2
print(chaine[:milieu])
print(chaine[milieu:])
```

- En cas d'indice trop grand, Python ne renvoie **pas d'erreur** : il s'arrête simplement à la fin.

À retenir 🧠

> Le slicing, c'est **COMME DÉCOUPER UNE TRANCHE DANS UNE BAGUETTE** : tu choisis le point de départ, le point d'arrêt et éventuellement l'épaisseur de coupe.