Présentation dans l'ordre décroissant d'utilisation.

Concaténation de variables avec les "formatted string"

Pour **concaténer plusieurs variables** dans une f-string il suffit de précéder la string de la lettre `f`. et d'englober les variables entre `{}`.

```
x = 2
y = 3

# va nous donner le résultat "L'addition de 2 et 3 est égale à 5"
print(f"L'addition de {x} et {y} est égale à {x + y}")

x = [1, 2, 3]

# va nous donner le résultat "la string de la liste [1, 2, 3]"
print(f"la string de la liste {x}")
```

On peut également faire des f-string multilignes.

```
x = 12

phrase = f"""Longue phrase
avec une variable {x}
dedans.
"""

print(phrase)
```

> Compatibilité
>
> Les f-strings sont apparues avec la version 3.7+ de python et sont maintenant les méthodes à privilégier.

Concaténation de variables avec la méthode format

La méthode format était la méthode principale utilisée pour concaténer des variables dans une string avant l'arrivée des f-strings. Reprenons l'exemple précédent.

```
x = 2
y = 3

print("L'addition de {premier} et {second} est égale à
{addition}".format(premier=x, second=y, addition=x+y))
```

On peut également l'utiliser sans nommer les paramètres à condition de suivre quelques règles.

```
ceci = "ceci"
cela = "cela"

# il faut le même nombre de paramètres que de {}
print("Je peut printer {} et {}".format(ceci, cela))

# ou alors on utilise des indices
print("Je peut printer {0} et {0}".format(ceci))
```

> La fonction format est toujours pertinente
>
> Dans certains cas, la fonction `format()` est plus intéressante que les `f-strings` lorsque l'on veut préparer la chaîne avant l'existence des variables à concaténer.

Options de formatage

Fonctionne avec les f-string et la fonction format. Elles suivent toute la syntaxe suivante : `f"{valeur:formatage}"`

"Padder" un nombre

```
x = 123
formatage = f"formatage de {x:04d} pour être sûr qu'il est composé de 4 chiffres"
print(formatage)
```

Explications : La traduction de "to padd" est compléter.

- le "0" indique que l'expression doit être complétée, le cas échéant, par des zéros.
- le "4" indique que l'expression doit contenir au moins 4 chiffres
- le "d" indique que l'expression doit être formatée comme une décimale.

> 📌 Exemple d'utilisation
>
> Lorsque l'on veut harmoniser les données dans un tableau ou lors du stockage de fichiers numérotés.

>Formater une date

```
from datetime import date

date_us = date(2022, 12, 24) # création d'un object date

date_fr = f"Avec des slashs, c'est quand même plus lisible : {date_us:%d/%m/%Y}"

print(date_fr)
```

Ici nous utilisons les spécifications de formattage d'un objet date issu de datetime. Vous trouverez plus de détails sur les spécifications de format de date à l'adresse suivante : https://docs.python.org/3/library/datetime.html#datetime.date.strftime

Résumé :

Exemples de formatage avec `strftime()` :

- `%Y` : Année sur 4 chiffres (ex: `2024`)
- `%m` : Mois sous forme numérique (ex: `12`)
- `%d` : Jour du mois (ex: `04`)
- `%A` : Nom complet du jour de la semaine (ex: `Wednesday`)
- `%B` : Nom complet du mois (ex: `December`)
- `%H` : Heure en 24 heures (ex: `14`)
- `%M` : Minutes (ex: `30`)
- `%S` : Secondes (ex: `45`)

Tronquer une chaîne de caractères

```
flute_a_bec = """
Souvent moquée et montrée du doigt, la flûte à bec est généralement mise de côté
quand il s'agit d'être choisie quand on veut commencer un instrument. Jugée trop
simple, petite, pas Rock'n Roll, et surtout associée aux cours de musique à
l'école, la flûte n'a pas bonne réputation.
"""

print(f"{flute_a_bec:.35}...") # n'affichera que les 35 premiers caractères suivie
de ...
```

Concaténer avec l'opérateur %

On peut utiliser le caractère % pour intégrer des variables dans une chaîne préparée

```
age = 12

concat = "Bonjour j'ai %d ans" % age
print(concat) # sera casté en integer

concat = "Bonjour j'ai %f ans" % age
print(concat) # sera casté en float
```

Concaténer avec l'opérateur +

Fastidieux, et vulnérable aux erreurs de conversions.

```
x = 1
y = 2

print("Je concatene " + str(x) + " et " + str(y) + " avec +, quel plaisir.")
```

Exercice: "10. Formater des chaînes de caractères"