

## Méthodes relative à la casse

---

La casse ou "case" en anglais décrit ce qui a attrait aux lettres majuscules et minuscules. Python propose plusieurs méthodes pour transformer des strings et modifier leur casse.

### upper() et lower().

```
minuscule = "minuscule"
majuscule = minuscule.upper()

print(majuscule) # "MINUSCULE"

crier = "PAS LA PEINE DE CRIER!"
parler = upper.lower()

print(parler) # "pas la peine de crier!"
```

### capitalize() et title().

```
phrase = "bonjour tout le monde"

print(phrase.capitalize()) # "Bonjour tout le monde"
print(phrase.title())      # "Bonjour Tout Le Monde"
```

Je veux que l'identité des utilisateurs de mon site soit affichée sur mon site selon la norme suivante : **NOM**  
**Prénom**

## Méthodes de remplacement

---

### replace()

Remplace un bout de chaîne par un autre si le premier est présent

```
bonjour = "Bonjour"

print(bonjour.replace("jour", "soir")) # "Bonsoir"
```

### strip()

Retire les espaces et les caractères spéciaux en début et en fin de chaîne si aucun caractère n'est passé en paramètre

```
plein_d_espaces      = " Coucou "
caracteres_speciaux = "\tj'adore faire des tabs\t"

print(plein_d_espaces.strip())      # "Coucou"
print(caracteres_speciaux.strip()) # "j'adore faire des tabs"

# Si des caractères sont présents en paramètres de la méthode alors ce seront ces
derniers qui seront retirés indépendamment de l'ordre
print(plein_d_espaces.strip("uco ")) # "uco"
```

rstrip() et lstrip() ont le même comportement que strip(), sauf que l'on peut choisir si l'on veut l'appliquer uniquement à gauche ou à droite de la chaîne de caractère.

## Méthodes de séparation et de jointure

---

### split()

Transforme une chaîne de caractères en liste

```
string = "1, 2, 3, 4, 5"
liste = string.split(", ")


print(liste) # ['1', '2', '3', '4', '5']
```

### join()

Transforme une liste en chaîne de caractères

```
liste = ['1', '2', '3', '4', '5']
join = ", ".join(liste)

print(join) # "1, 2, 3, 4, 5"
```

 la fonction `join()` n'accepte que des listes de chaînes de caractères en paramètre.

```
liste = [1, 2, 3, 4, 5]
join = "|".join(liste) # soulèvera une erreur
```

## Méthodes de vérification de chaînes

---

### islower() et isupper().

Vérifie si la chaîne est entièrement en minuscule ou en majuscule et renvoi **True** ou **False**

```
print("Bonjour".islower()) # False
print("BONJOUR".isupper()) # True
```

### isdigit().

Vérifie si la chaîne est entièrement composée de chiffres et renvoi **True** ou **False**

```
print("2 et 2 font 4".isdigit()) # False
print("12345".isdigit())         # True
```

## Méthode de comptage

---

### count().

Compte le nombre d'une occurrence dans une chaîne de caractères

```
print("coucou".count("cou")) # 2
print("coucou".count("Cou")) # 0, car case-sensitive
print("coucou".count("cou ")) # 0, car il n'y a pas d'espace : ' ', après 'cou'
dans 'coucou'
```



la fonction est case-sensitive

## Méthode de recherche

---

### find() et rfind().

Find cherche une occurrence dans une chaîne de caractères et renvoi l'index du premier caractères à partir duquel l'occurrence est présente. Renvoie **-1** si l'occurrence est introuvable.

Rfind fait la même chose, mais en commençant par la droite, donc la fin de la chaîne de caractères.

```
phrase = "Le Python, c'est cool. Python est puissant."
print(phrase.find("Python")) # 3
print(phrase.find("python")) # -1, case sensitive
print(phrase.rfind("Python")) # 23
```

### index().

Fonctionne comme **find()** à l'exception que si l'occurrence n'est pas présente, la méthode soulèvera une erreur au lieu de retourner **-1**.

```
phrase = "Le Python, c'est cool. Python est puissant."
print(phrase.index("python")) # erreur
```

### startswith() et endswith().

Détermine si la chaîne commence, ou se termine par la sous-chaîne passée en paramètre.

```
je = "Je commence par je".startswith("je")
print(je) # False

je = "Je commence par je".endswith("je")
print(je) # True
```



Je peux déterminer si un fichier reçu est bien une image. Donc, qui termine par **.png**

## La fonction `help()`

---

Il existe énormément de fonctions en python, rien que pour les string. Il est impossible de toute les retenir. Lorsque vous avez un doute sur le fonctionnement d'une fonction, utiliser la fonction `help()` au lieu d'aller sur google, cela vous fera gagner du temps, et votre source d'information sera sûre.

```
help('str.join')
```



[Liste exhaustive des fonctions de string en python](#)

Exercice: "8. Manipulation de chaînes"