

# Les chaines de caractères

Fonctions et méthodes utiles

# Plan

- La taille d'une chaîne
- Les tranches dans une chaîne
- Ajouter une chaîne à une autre
- Des méthodes et fonctions utiles:
  1. `upper()` et `lower()`
  2. `isupper()` et `islower()`
  3. `capitalize()`
  4. `split()`
  5. `find()`
  6. `replace()`
  7. `count()`
  8. `list()`
  9. `join()`

# Les chaines en python

- Les chaînes de caractères peuvent être considérées comme des listes (de caractères) un peu particulières. Chaque caractère possède un **indice** qui commence à 0 de même que les listes.
- Il existe aussi les indices négatifs de même que pour les listes.

```
animaux = "girafe tigre"  
print(animaux)  
L=len(animaux)  
print(L)  
print(animaux[3])
```

- Résultat:

```
'girafe tigre'  
12  
'a'
```

# Les tranches de chaines

- Nous pouvons utiliser certaines propriétés des listes comme les tranches :

```
animaux = "girafe tigre"  
print(animaux[0:4])  
print(animaux[9:])
```

- Résultat:

```
'gira'  
'gre'
```

# Ajouter un caractère ou une chaîne à une autre chaîne

1. La première façon c'est d'utiliser l'opérateur **+** qui est un opérateur de **concaténation** de deux chaînes de caractères.

```
Ch1="Bonjour"  
Ch2=ch1 + "les amis"  
print(Ch2)                                #Bonjour les amis
```

## 2. *Remarques:*

- les méthodes **append** et **insert** ne sont pas valables pour les chaînes de caractères.
- La modification d'un caractère par une affectation n'est pas possible pour les chaînes de caractères.

# Modifier la casse d'une chaine de caractères

- Convertir une chaine en majuscule à l'aide de la méthode **upper()**:

```
animaux = "girafe tigre"  
animaux=animaux.upper()  
print(animaux)
```

GIRAFE TIGRE

- Convertir une chaine en minuscule à l'aide de la méthode **lower()**:

```
animaux = "GIRAFE TIGRE"  
animaux=animaux.lower()  
print(animaux)
```

girafe tigre

# Vérifier la casse d'un caractère

- La méthode `islower()` permet de vérifier si une chaîne est minuscule ou non. Elle retourne `True` s'elle est minuscule et `False` sinon.
- La méthode `isupper()` permet de vérifier si une chaîne est majuscule ou non. Elle retourne `True` s'elle est majuscule et `False` sinon.

```
x="Bonjour"  
if (x[o].islower()==True):  
    print("miniscule")  
else:  
    print("majuscule")
```

# La première lettre d'une chaîne en majuscule

- Pour mettre en majuscule la première lettre seulement, on peut faire :

```
X="girafe"  
X[0].upper() + X[1:]
```

- Ou utiliser la méthode `capitalize()` et l'utiliser comme suit:

```
X="girafe"  
X.capitalize()
```



# Modifier un caractère de la chaîne

- Contrairement aux listes, **on a pas la possibilité de modifier un caractère de la chaîne via une simple affectation.**

```
animaux = "girafe tigre"  
animaux[4] = "F"
```

- **Résultat:**

```
Traceback (most recent call last):  
  File "D:\TDM\langage python\tests\test3.py", line 257, in <module>  
    animaux[4] = "F"  
TypeError: 'str' object does not support item assignment
```

# Remplacer une chaîne par une autre

- La méthode **.replace()** permet de substituer une chaîne de caractères par une autre :

```
Machaine.replace(oldStr, newStr, nombre)
```

- La méthode **replace()** prend trois paramètres:
- **oldStr**: La chaîne à rechercher(**Obligatoire**)
- **newStr** : La chaîne par laquelle remplacer l'ancienne chaîne (**Obligatoire**)
- **Nombre** : Un nombre spécifiant le nombre d'occurrences de l'ancienne chaîne que vous souhaitez remplacer. La valeur par défaut est toutes les occurrences (**Optionnel**)

# Remplacer une chaine par une autre

- Exemple 1:

```
animaux = "girafe tigre"  
animaux=animaux.replace("tigre", "singe")  
print(animaux)
```

```
>>  
'girafe singe'
```

- Exemple 2:

```
str = "Mohamed Mohamed Mohamed Ali"  
res = str.replace(" Mohamed ", "Salim", 2)  
print(res)
```

```
>>  
Salim Salim Mohamed Ali'
```

# Chercher une chaîne dans une autre chaîne(1)

- La méthode `.find()` permet de faire la **recherche** d'une chaîne de caractères passée en argument , et **retourne l'indice** où elle se trouve (première occurrence ):
- Si l'élément n'est pas trouvé, alors la valeur **-1 est renvoyée**.
- **Syntaxe:**

```
Machaine.find(valeur, start, end)
```

- **valeur(Obligatoire)** : La valeur à rechercher
- **start(Optional)** : Où commencer la recherche. La valeur **par défaut est zéro (0)**
- **end(Optional)** : Où terminer la recherche. La valeur **par défaut est à la fin de la chaîne (len()-1)**

# Chercher une chaine dans une autre chaine(2)

- Exemple 1:

```
animal = "girafe"  
X=animal.find("i") ; print(X)  
Y=animal.find("ra") ; print(Y)  
Z=animal.find("ti") ; print(Z)
```

1  
2  
-1

```
Machaine= "Bonjour les amis"  
i = Machaine.find("u", 3, 15)  
print(i)
```

5

# Vérifier l'existence d'un caractère dans une chaîne

- Si on a besoin juste de vérifier si un caractère se trouve dans une chaîne, on peut se servir uniquement de l'opérateur **IN** :
- **Exemple:**

```
Chaine='Bonjour'  
  
if 'B' in chaine :  
    print('existe')  
else :  
    print('n existe pas ')
```

# Nombre d'occurrence d'une chaine

- La méthode `count()` compte le nombre d'occurrences d'une chaîne de caractères passée en argument :
- | code  | résultat                 |
|---|--------------------------|
| <pre>animaux = "girafe tigre"<br/>C=animaux.count("i")<br/>print(C )<br/><br/>N=animaux.count("tigre")<br/>print(N )<br/><br/>M=animaux.count("z")<br/>print(M)</pre> | <pre>2<br/>1<br/>0</pre> |

# Convertir une chaîne de caractère en une liste de caractère

- La fonction `list()` prend une chaîne de caractère en paramètre et renvoie une liste.

```
chaîne="Python"  
S=list(chaîne)  
print(S)
```

- Résultat:

```
['P', 'y', 't', 'h', 'o', 'n']
```



# Découper une chaîne de caractères

- Pour découper une chaîne de caractère, on peut utiliser la méthode `split()` comme suit:

```
animaux = "girafe tigre singe souris"  
T=animaux.split(" ")  
print(T)
```

- Le résultat est :

```
['girafe', 'tigre', 'singe', 'souris']
```

- On peut préciser un caractère de découpage de la chaîne, de cette manière:

```
animaux = "girafe,tigre,singe,souris"  
T=animaux.split(",")  
print(T)
```

- Le même résultat sera obtenu.

# Conversion d'une liste de chaînes de caractères en une chaîne de caractères

- La conversion d'une liste de chaînes de caractères en une chaîne de caractères fait appelle à la méthode `.join()`.

```
seq = ["A", "T", "G", "A", "T"]  
X="-".join(seq)  
Y=" ".join(seq)  
Z="".join(seq)  
  
print(X)  
print(Y)  
print(Z)
```

Résultat

```
>>  
'A-T-G-A-T'  
'A T G A T'  
'ATGAT'
```

- Attention, la méthode `.join()` ne s'applique qu'à une liste de chaînes de caractères.