Informatique Pour Tous

Cours 5 – Les chaînes de caractères (string)

Beaucoup de programmes manipulent du texte et pas seulement des valeurs numériques. Un texte est constitué de **caractères** : lettres, nombres, ponctuations, espaces etc… Une **chaîne de caractères** (**string** en anglais) est une séquence de caractères. Par exemple la chaîne de caractères `"carbone-14"` est une séquence de 10 caractères.

A / Accès aux caractères individuels d'une chaîne de caractère

Un langage de programmation tel que Python doit donc être pourvu de mécanismes qui permettent d'accéder séparément à chacun des caractères d'une chaîne. Python considère qu'une chaîne de caractères est un objet de la catégorie des séquences, lesquelles sont des **collections ordonnées d'éléments**. Chaque caractère dans une chaîne peut donc être désigné par sa place dans la séquence, à l'aide d'un **index**.
Pour accéder à un caractère bien déterminé, on utilise le nom de la variable qui contient la chaîne et on lui accole, entre 2 crochets, l'index numérique qui correspond à la position du caractère dans la chaîne.

! Attention : numérotation à partir de 0 !

```
ch='classe préparatoire PTSI'

print(ch[3])
print(ch[20])

x=ch[12]
print(x)

print(type(x))
print(type(ch))
```

```
s
P
r
<class 'str'>
<class 'str'>
```

B / Opérations sur les chaînes de caractères

Python intègre de nombreuses **fonctions** et **méthodes** (une méthode est un autre nom pour une fonction spécifique à type d'objet) qui permettent d'effectuer divers traitements sur les chaînes de caractères : conversion majuscule/minuscule, découpage en chaînes plus petites, recherche de mots, concaténation de mots…

1 - Assembler plusieurs petites chaines pour en construire de plus grandes. On parle de **concatenation**.
On utilise dans Python l'opérateur +.

```
a="Un anneau pour les gouverner tous "
b="et dans les ténèbres les lier."
c=a+b

print(c)
```

```
Un anneau pour les gouverner tous et dans les ténèbres les lier.
```

2 - Déterminer la longueur (nombre de caractères) d'une chaîne, en faisant appel à la `len()`.

```
a="Non Luke, je suis ton père"
print(a)
print(len(a))
```

```
Non Luke, je suis ton père
26
```

3 - Convertir en nombre véritable une chaîne de caractères qui représente un nombre entier : il faut passer par la fonction `int()` qui convertit la chaîne (`string`) en entier (`int`).

```
age=input("Quel est votre âge ?")
print(age)
print(type(age))

x=int(age)
print(x)
print(type(x))
```

```
Quel est votre âge ?26
26
<class 'str'>
26
<class 'int'>
```

On peut faire la même chose pour un nombre réel (`float`).

```
taille=input("Quelle est votre taille (m) ? ")
print(taille)
print(type(taille))

x=float(taille)
print(x)
print(type(x))
```

```
Quelle est votre taille (m) ? 1.90
1.90
<class 'str'>
1.9
<class 'float'>
```

4 - Extraction (ou **slicing** en anglais): extraire une petite chaîne d'une chaîne plus longue, c'est du slicing dans Python. On indique entre crochets les indices correspondants au début et fin de la « tranche ».

```
print()
print()

ch="quantique"

print(ch[0:3])
print(ch[3:7])
print(ch[3:])
print(ch[:3])
print(ch[-1])
```

```
qua
ntiq
ntique
qua
e
```

Attention : Dans la tranche [n,m], le $n^{ième}$ caractère est inclus, mais pas le $m^{ième}$.

5 - Trouver un caractère dans une chaîne : Pour trouver à quel index (ou rang) apparaît un caractère dans une chaîne, on utilise l'instruction `<var>.find(«car»).`

```
citation="Ô Roméo Roméo pourquoi es-tu Roméo ?"
print(citation.find("o"))
```

```
3
```

Attention : il n'indique le rang que du premier rencontré…

6 - Compter le nombre de fois qu'apparait un caractère dans une chaîne : Pour compter le nombre de fois qu'apparaît un caractère dans une chaîne, on utilise l'instruction `<var>.count(«car»).`

```
citation="Ô Roméo Roméo pourquoi es-tu Roméo ?"
print(citation.count("o"))
```

```
8
```

C / Parcourir la chaîne de caractères avec la boucle `for` et la boucle `while`

Souvent, nous avons besoin de parcourir chaque élément (caractère) de notre chaîne. Cela est particulièrement facile avec l'utilisation des boucles comme nous allons le voir sur les cas suivants.

C. 1 – Utilisation de la boucle `for`

Dans le cas ci-dessous, la variable `lettre` prend successivement comme valeur chaque élément de la chaîne de caractères `Pauling`.

```
nom="Pauling"
for lettre in nom:
    print(lettre, end=",")
```

```
P,a,u,l,i,n,g,
```

On peut aussi faire apparaître explicitement chaque élément par son index `nom[]`.

```
nom="Pauling"
for i in range (len(nom)):
    lettre=nom[i]
    print(lettre,end="*")
```

```
P*a*u*l*i*n*g*
```

C. 2 – Utilisation de la boucle `while`

Il est aussi possible d'utiliser une boucle pour parcourir une chaîne de caractères même si cette méthode semble moins utilisée.

```
nom="Pauling"
i=0
while i <len(nom):
    lettre=nom[i]
    print(lettre,end="-")
    i=i+1
```

```
P-a-u-l-i-n-g-
```