Lycée Marcelin BERTHELOT
BCPST 1C – Mme MOREL

Type chaîne de caractères

1 Définition

Un **caractère** est un symbole (lettre, chiffre, ponctuation, espace, ...) délimité par des apostrophes ou des guillemets.
Une **chaîne de caractères** est une suite de caractères délimitée par des apostrophes ou des guillemets: elle peut contenir n'importe quel symbole, sauf des guillemets.

2 Opérations

Commandes opérant sur les chaînes de caractères:

Opérations sur les chaînes	Instruction Python
égalité	==
concaténation (collage)	+
répétition n fois	n*
appartenance	in
conversion nombre → chaîne	str (nombre)
conversion chaîne → entier	int (chaîne)
conversion chaîne → réel	float (chaîne)

Exemple 1 :
(1) `2 * "bon"`
(2) `"BCPST1C"=="BCPST 1C"`

Remarque 1 Lorsqu'il compare deux chaînes, la moindre différence est détectée (espace, accent, virgule, majuscule, ...)
En particulier, le sens de la phrase n'est pas pris en compte: `"3 est un entier"` et `"3 est entier"` sont deux chaînes différentes.

Exemple 2 `"bon"+"jour"`

Remarque 2 La juxtaposition, sans opérateur + , de plusieurs chaînes de caractères devrait provoquer une erreur.

Exemple 3 `"Je passerai le concours en" + 2022`

Remarque 3 PYTHON refuse d'ajouter un nombre et une chaîne de caractères. Pour insérer un nombre dans une phrase, il faut au préalable le transformer en chaîne de caractères. Pour cela, on peut, ou bien mettre le nombre entre guillemets, ou bien utiliser la commande de conversion `str()` (qui transforme un nombre en chaîne de caractères).

3 Accès aux éléments d'une chaîne

L'affectation d'une chaîne à une variable permet d'accéder individuellement aux caractères de la chaîne.
Si `ch` désigne une chaîne de caractères et i, j deux variables de type entier:

Opérations sur les chaînes	instruction PYTHON
Nombre de caractères de `ch`	`len(ch)`
Accès au caractère de ch situé en position i	`ch[i]`
Extraction des caractères de ch compris entre les positions i et $j-1$	`ch[i:j]`

Exemple 4 :
(1) `ch="Python"; len(ch)`
(2) `ch2="informatique"; ch2[2:7]`

Remarque 4 La numérotation des positions des caractères au sein d'une chaîne de caractères commence à 0!! Donc `ch[k]` est le (k+1)ième élément de la chaîne.

Exemple 5 `ch="bonjour"; ch[2]; ch[7]`

Remarque 5 L'expression de `ch[i]` n'a pas de sens si i est supérieur ou égale à `len(ch)`

Remarque 6 L'utilisation d'un indice négatif permet d'accéder aux caractères à partir de la fin de la chaîne. Ainsi `ch[-1]` désigne le dernier élément d'une chaîne, `ch[-2]` l'avant-dernier, etc ...

4 Chaînes de caractères et récursivité

Exemple 6 : Écrire une fonction qui renvoie l'inverse d'une chaîne de caractères.
Le type de données à renvoyer est une chaîne de caractères.
L'inverse d'une chaîne vide ou de longueur un (ne contenant qu'un seul caractère) est elle-même.
Lorsque la chaîne est plus longue (contient au moins deux caractères), on obtient son inverse en :

- plaçant le dernier caractère de la chaîne au début de la chaîne inversée,
- en inversant la chaîne ou le dernier caractère a été exclu.

```
def inverse(ch):
        if  len(ch)<=1 :
                return ch
        return ch[-1] + inverse(ch[: len(ch)-1])
```

Explications avec `ch="oups"`

5 Algorithmes de recherche

Exemple 7 : recherche du mot "ATG" dans un brin d'ADN.
Il s'agit de déterminer la présence ou non d'un codon d'initialisation (ATG) dans un brin d'ADN (constitué des nucléides A, T, C et G).
N'utilisez pas le test `"ATG" in brin` *entre les deux chaines de caractères!!*

Plus généralement, nous allons écrire la méthode "naïve" de recherche d'un mot dans une chaîne de caractères.
Nous introduisons les notations suivantes:

- `texte` désigne une chaîne de caractères.
- `n` la longueur de `texte`.
- `mot` désigne une chaîne de caractères.
- `l` est la longueur de `mot`.

On dit que le mot **apparaît** dans le texte si et seulement si il existe $k \in [\![0, \ldots\ldots]\!]$ tel que `mot=texte[k:k+l]`

1. Écrire une fonction `Position(texte,mot,k)` qui prend en arguments `texte,mot` et un entier $k \in [\![0, \ldots\ldots]\!]$, et qui renvoie `True` si `mot` apparaît dans `texte` en position k et `False` sinon.

2. **algorithme à connaître:**
   Écrire une fonction `Recherchemot(texte,mot)` qui prend en arguments `texte,mot`, et qui renvoie `True` si `mot` apparaît dans `texte` et `False` sinon.

3. Adapter la fonction précédente pour créer une fonctions `nombre(texte,mot)` qui renvoie le nombre d'apparitions de `mot` dans `texte`.