TP 4 : Les chaînes de caractères

Objectifs :

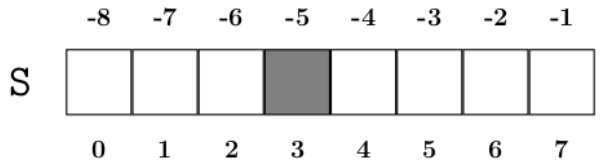
Manipuler les chaînes de caractères et découvrir les fonctions prédéfinis pour ce faire.

Rappel :

- Les chaînes sont des séquences de caractères non mutables. On ne peut pas donc modifier un ou plusieurs caractères.
- Si on ajoute un caractère (ou une autre chaîne) par concaténation à une chaîne existante, le résultat est une nouvelle chaîne avec une nouvelle adresse en mémoire.
- Un seul caractère en Python est simplement une chaîne de longueur 1.

Exercice 1 : Accès aux éléments d'une chaîne

L'accès aux caractères se fait par crochets muni d'index (ou position).



$$S[-5] = S[3]$$

```
>>> ch1 = "Salut tout le monde"
>>> print (ch1[0])
. . . . . .
>>> print (ch1[5])
. . . . . .
```

```
>>> print (ch1[-1])
. . . . . .
>>> print (ch1[-4])
. . . . . .
```

Exercice 2 : Découpage

```
>>> print (ch1[1:12])
. . . . . .
>>> print (ch1[2:])
. . . . . .
>>> print (ch1[:12])
. . . . . .
>>> print (ch1[2:11:2])
. . . . . .
>>> print (ch1[::3])
. . . . . .
>>> print (ch1[::-2])
. . . . . .
>>> print (ch1[::-1])
. . . . . .
>>> ch1[0] = 'C'
. . . . . .
```

Exercice 3 : Operations sur les chaînes de caractères

```
>>> dir (str) #Liste toutes les methodes des chaînes
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__',
    '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__',
    '__getitem__', '__getnewargs__', '__gt__', '__hash__', '__init__',
    '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__',
    '__mod__', '__mul__', '__ne__', '__new__', '__reduce__',
    '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__',
    '__sizeof__', '__str__', '__subclasshook__', 'capitalize', 'casefold',
    'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format',
    'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal',
    'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable',
    'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip',
    'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
    'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith',
    'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
```

Déterminer l'affichage des instructions qui suivent ?

1. ```
   >>> ch1 = "Salut tout le monde"
   >>> ch2 = " Python "
   >>> ch3 = "est le langage de programmation "
   >>> ch4 = "le plus utilisé"
   >>> ch5 = "COUCOU"
   >>> ch6 = ch2+ch3+ch4
   >>> print (ch6)
   . . . . . .
   >>> ch7 = ch2*3
   >>> print (ch7)
   . . . . . .
   >>> print ('La longueur de ch1=',len(ch1))
   . . . . . .
   >>> print (len(ch2))
   . . . . . .
   >>> print (ch2.strip())
   . . . . . .
   >>> print ('La longueur de ch2 = ',len(ch2))
   . . . . . .
   >>> print ('La nouvelle longueur =',len(ch2.strip()))
   . . . . . .
   ```

2. ```
   >>> ch8 = ch2.strip()
   >>> print ('La longueur de ch8 = ',len(ch8))
   . . . . . .
   >>> print (ch5.lower())
   . . . . . .
   >>> print (ch1.upper())
   . . . . . .
   >>> print (ch3.isupper())
   . . . . . .
   ```

3. ```
   >>> ch1 = "Salut tout le monde"
   >>> ch2 = " Python "
   >>> ch3 = "Salut, les collègues, ça roule?"
   >>> ch4 = "coucou"
   >>> print (ch1.find('tout'))
   . . . . . .
   >>> print (ch1.count('ut'))
   . . . . . .
   >>> print (ch2.replace("thon", "charm" ,1))
   . . . . . .
   ```
   ```

```
>>> l = ch3.split(",")
>>> print (l)
. . . . . .
>>> print (type(l))
. . . . . .
>>> print (ch3.split(","))
. . . . . .
>>> print (ch3.split(" "))
. . . . . .
>>> print (max(ch4))
. . . . . .
>>> print (min(ch4))
. . . . . .
>>> print ( "?" in ch3 )
. . . . . .
>>> print ( "le" not in ch1 )
. . . . . .
```

Exercice 4 : Applications

Écrire des programmes qui permettent de :

1. Saisir une chaîne de caractères *ch* et déterminer si elle est palindrome. Une chaîne est dit palindrome si elle se lit de la même façon dans les deux sens.
2. Saisir une chaîne de caractères *ch* et un caractère *c* puis, déterminer si *c* est contenu dans *ch*. Si oui, le programme affiche sa position.
3. Saisir une chaîne de caractères *ch* et déterminer le nombre d'occurrence des caractères distincts dans *ch*.
4. Saisir une chaîne de caractères contenant un ensemble de mots séparés par des espaces. Puis, afficher le premier et le dernier mot de la chaîne.