Manipulation de chaînes de caractères en Python

Objectifs :

- Comprendre ce qu'est une **chaîne de caractères** (`str`).
- Savoir **lire**, **écrire**, **modifier** et **analyser** du texte simple.
- Utiliser quelques **méthodes utiles** sur les chaînes (`upper()`, `lower()`, `find()`, etc.).

Table des matières

1. Introduction simple....1
2. Exemples de base à tester....1
3. Une chaîne de caractères est un objet....1
   a. Variable vs Objet : les différences essentielles....2
   b. Exemple en Python....2
   c. Pour simplifier :....2
4. Les principales méthodes sur une chaîne de caractères....2
5. À savoir absolument :....3
   a. Résultat final :....3
6. Exercices pratiques....3
   a. Exercice 1 : Premiers pas....3
   b. Exercice 2 : Détecteur de mot....3
   c. Exercice 3 : Découpage de phrase....4
7. Défis supplémentaires pour les rapides....4
   a. Défi 1 — Inverser une chaîne....4
   b. Défi 2 — Compter les voyelles....4
   c. Défi 3 — Détecter un palindrome....4

1. Introduction simple

- En Python, **un texte** est une **chaîne de caractères** (type `str`).
- On l'écrit **entre guillemets** : `'Bonjour'`, `"Bonjour"` ou encore `"""Bonjour"""`.

2. Exemples de base à tester

```
texte = "Bonjour"

# Longueur du texte
print(len(texte))  # 7 la fonction len() appliquée à une str renvoie sa longueur

# Accès à une lettre
print(texte[0])    # 'B' est le premier caractère de "Bonjour"

# Parcourir la chaîne de caractères
for lettre in texte:
    print(lettre)

phrase =  "Bonjour" + "Marcel" # + opérateur de concaténation de chaînes
print(phrase)  # affichera Bonjour Marcel
```

3. Une chaîne de caractères est un objet

Une chaîne de caractère est plus qu'une valeur, c'est un objet

En informatique une **variable** et un **objet** sont deux concepts liés mais distincts.

a. Variable vs Objet : les différences essentielles

Élément	Variable	Objet
Définition	Nom ou référence à une donnée en mémoire	Instance concrète d'un type (ou d'une classe)
Rôle	Sert à désigner ou manipuler une valeur	Contient les données et les comportements (méthodes)
Contenu	Stocke une **référence** vers un objet	Contient des **données** et parfois des **méthodes**
Création	`x = ...` (on attribue un nom à un objet)	`objet = MaClasse()` (on crée une instance d'une classe)
Durée de vie	Liée à la portée (scope) de la variable	L'objet peut continuer à exister si d'autres références y mènent

b. Exemple en Python

```
class Voiture:
    def __init__(self, marque):
        self.marque = marque

v1 = Voiture("Peugeot")
```

- `v1` est une **variable** : c'est le **nom** qui désigne quelque chose.
- `Voiture("Peugeot")` crée un **objet** (une instance de la classe `Voiture`).
- `v1` **référence** cet objet.

c. Pour simplifier :

- La **variable est une étiquette**, l'**objet est la chose étiquetée**.
- Plusieurs variables peuvent référencer le **même objet** :

```
a = "Bonjour"
b = a
```

Ici `a` et `b` sont deux **variables** qui pointent vers le **même objet** (la même chaîne de caractères).

4. Les principales méthodes sur une chaîne de caractères

Méthode	Utilité	Exemple
`.lower()`	Tout en minuscules	`"Python".lower()` → "python"
`.upper()`	Tout en majuscules	`"Python".upper()` → "PYTHON"
`.find(mot)`	Chercher une position (ou -1 si absent)	`"Salut".find("lu")` → 2
`.replace(ancien, nouveau)`	Remplacer un morceau de texte	`"Salut".replace("lu", "lo")` → "Salot"
`.strip()`	Supprimer les espaces inutiles	`" texte ".strip()` → "texte"
`.split()`	Découper en morceaux (liste)	`"un deux trois".split()` → `["un", "deux", "trois"]`

Toutes les méthodes : https://docs.python.org/fr/3.13/library/stdtypes.html#string-methods

5. À savoir absolument :

- Les chaînes sont **immuables ➔** on ne peut pas changer une lettre directement.
- Il faut créer une **nouvelle chaîne** si on veut modifier.

  Exemple : `nom_complet = "Mauryce Baldani"`
  `prenom = nom_complet.split()[0]`
  `prenom[4] = "y"`
  `TypeError: 'str' object does not support item assignment`

`Lorsqu'on exécute ces deux instructions en Python :`

```
prenom = "Mauryce"
prenom = prenom.replace('y','i')
```

voici ce qui se passe étape par étape :

1. **Première instruction** :

```
prenom = "Mauryce"
```

→ La variable `prenom` reçoit la chaîne de caractères `"Mauryce"`.

2. **Deuxième instruction** :

```
prenom = prenom.replace('y','i')
```

→ Python exécute la méthode `.replace('y', 'i')` sur la chaîne `"Mauryce"`. Cela renvoie une **nouvelle** chaîne où toutes les occurrences de `'y'` sont remplacées par `'i'`, soit `"Maurice"`.

→ Cette nouvelle chaîne est **réaffectée** à la variable `prenom`. L'ancienne valeur `"Mauryce"` est alors perdue (sauf si elle était utilisée ailleurs).

a. Résultat final :

```
print(prenom)  # Affichera : Maurice
```

un autre exemple à tester :

```
a = bonjour
b = a
c = b
c = a.replace('b','B')
print( a , b , c )
```

6. Exercices pratiques

a. Exercice 1 : Premiers pas

Écrire un programme qui :

1. Demander à l'utilisateur d'écrire son prénom.
2. Afficher le prénom en :
   - **Majuscules**,
   - **Minuscules**,
   - **Nombre de lettres**.

b. Exercice 2 : Détecteur de mot

Écrire un programme qui :

1. Demander une phrase à l'utilisateur.
2. Demander un mot à chercher.
3. Afficher si le mot est présent ou non dans la phrase.

Exemple :

```
Entrez une phrase : Python est génial.
Mot à chercher : Python
--> Le mot est présent !
```

c. Exercice 3 : Découpage de phrase (exercice difficile ! Peut être passé)

Écrire un programme qui :

- Demander une phrase,
- Afficher chaque mot sur une **nouvelle ligne**.

Exemple :

```
Entrez une phrase : J'aime programmer en Python.

Affichage :
J'aime
programmer
en
Python.
```

7. Défis supplémentaires pour les rapides

a. Défi 1 — Inverser une chaîne

Demander une phrase et l'**afficher à l'envers**.

Exemple :

```
Entrez un mot : Python
Affichage : nohtyP
```

💡 Indice : utiliser l'astuce Python `texte[::-1]`.

b. Défi 2 — Compter les voyelles

Écrire un programme qui :

- Demander une phrase,
- Compter **le nombre de voyelles** (`a`, `e`, `i`, `o`, `u`, `y`) dans la phrase.

Exemple :

```
Entrez une phrase : Bonjour
Résultat : 3 voyelles
```

c. Défi 3 — Détecter un palindrome

Un **palindrome** est un mot qui se lit pareil à l'endroit et à l'envers (ex : "kayak", "radar").

Programme demandé :

- Demander un mot,
- Afficher s'il est **palindrome** ou non.