Cours : Parcours séquentiel dans une chaîne

Introduction

Une **chaîne de caractères** (type `str` en Python) est une **séquence ordonnée** de symboles (lettres, chiffres, espaces, etc.). On peut la parcourir **séquentiellement** pour accéder, lire ou modifier des informations.

Exemples :

```
mot = "Bonjour"
print(mot[0])   # B (premier caractère, indice 0)
print(mot[-1])  # r (dernier caractère)
```

Parcours avec une boucle `for`

Le plus simple est d'utiliser une boucle `for` **directement sur la chaîne** :

```
mot = "Python"
for lettre in mot:
    print(lettre)
```

Affiche chaque caractère l'un après l'autre.

On peut aussi utiliser les **indices** :

```
mot = "Python"
for i in range(len(mot)):
    print(i, mot[i])
```

Parcours avec une boucle `while`

Avec une boucle `while`, on utilise un **compteur** :

```
mot = "Python"
i = 0
while i < len(mot):
    print(mot[i])
    i += 1
```

Quelques méthodes utiles sur les chaînes

- `len(mot)` : longueur

- `mot.upper()` : majuscules
- `mot.lower()` : minuscules
- `mot.count("a")` : nombre d'occurrences
- `mot.find("on")` : position de la sous-chaîne `"on"`

TD – Parcours de chaînes et boucles

Exercice 1 – Comptage de lettres

Écrire un programme qui compte combien de fois la lettre `"a"` apparaît dans une chaîne saisie par l'utilisateur.

Exercice 2 – Comptage de voyelles

Écrire une fonction `nb_voyelles(texte)` qui retourne le nombre de voyelles (`a, e, i, o, u, y`) dans la chaîne `texte`.

Exercice 3 – Détection de palindrome

Écrire un programme qui teste si une chaîne donnée est un **palindrome** (ex. "radar", "ressasser"). *Indice :* comparer la chaîne avec sa version inversée.

Exercice 4 – Parcours avec indices

Écrire une fonction qui prend un mot et affiche :

```
0 -> B
1 -> o
2 -> n
...
```

pour chaque caractère du mot.