□ Les Chaînes de Caractères (str)

Imagine un collier de perles □ — chaque perle est une lettre. Une chaîne de caractères c'est exactement ça : une suite de lettres, chiffres ou symboles entre guillemets. En Python, on appelle ça un **str**.

python

```python
# Créer une chaîne
mot    = "Bonjour"
phrase = "Python c'est cool !"
vide   = ""                    # chaîne vide
# La longueur : combien de caractères ?
print(len("Bonjour"))          # affiche 7
```

□ Les indices commencent à **0**. La première lettre de **mot** est à **mot[0]**, la deuxième à **mot[1]**… et la dernière à **mot[len(mot)-1]**.

□□ 1. Les Opérations Essentielles

Opération	Code Python
Longueur	`len(ch)`
Accéder à la lettre i	`ch[i]`
Extraire une partie	`ch[debut:fin]` ← fin exclu
Supprimer de d à f	`ch = ch[:d] + ch[f:]`
Tout en MAJUSCULES	`ch.upper()`
Tester si c'est un chiffre	`ch.isdecimal()` → True / False
Convertir en texte	`str(x)`
Convertir en entier / réel	`int(ch)` | `float(ch)`
Trouver la position	`ch.find('mot')` → -1 si absent
Coller deux chaînes	`ch1 + ch2`

Code ASCII d'un caractère	`ord(c)`
Caractère depuis code ASCII	`chr(d)`

□ Accès et slicing

python

```python
mot = "Bonjour"
# Accéder à une lettre
print(mot[0])              # B
print(mot[6])              # r
print(mot[len(mot)-1])     # r ← dernière lettre
# Slicing ch[debut:fin] (fin non inclus)
print(mot[0:3])            # Bon
print(mot[3:7])            # jour
print("informatique"[8:12]) # ique
```

□ Majuscules · find · suppression

python

```python
phrase = "Bonjour tout le monde"
# Tout en majuscules
print(phrase.upper())      # BONJOUR TOUT LE MONDE
# Trouver la position d'un mot
print(phrase.find("tout")) # 8
print(phrase.find("Java")) # -1 ← absent
# Supprimer les caractères de l'indice 8 à 12
ch = "Bonjour !"
ch = ch[:7] + ch[8:]       # Bonjour!
```

□ Inverser une chaîne — avec une boucle

python

```python
# ch[::-1] est interdit — on utilise une boucle
mot     = "radar"
inverse = ""
for i in range(len(mot)-1, -1, -1) :
    inverse = inverse + mot[i]
```

```
print(inverse)              # radar
```

□ **find()** retourne -1 si le mot n'est pas trouvé. Toujours vérifier le résultat avant de l'utiliser !
Slicing **ch[debut:fin]** : le caractère à l'indice **fin** est exclu.

□ 2. La Concaténation

Concaténer, c'est **coller** des morceaux de texte ensemble avec **+**. Si tu veux coller un nombre avec du texte, il faut d'abord le convertir avec **str()**.

python

```python
prenom = "Ali"
nom    = "Ben"
# Coller deux chaînes
print(prenom + " " + nom)        # Ali Ben
# Coller du texte avec un nombre
age = 17
print("J'ai " + str(age) + " ans") # J'ai 17 ans
```

□□ Impossible : **"Age : " + 17** → Python bloque ! On ne peut pas coller du texte et un nombre directement.
Solution : **"Age : " + str(17)**

Opération	Code Python
Coller deux chaînes	`ch1 + ch2`
Coller texte + nombre	`ch + str(nombre)`
Code ASCII → caractère	`chr(65) → 'A'`
Caractère → code ASCII	`ord('A') → 65`

□ 3. Exercices — Opérations de base

□□ Exercice 1 — Palindrome

□ **Objectif :** Demander un mot. Vérifier si c'est un palindrome (il se lit pareil à l'endroit et à l'envers).

```
terminal
Entrez un mot :
› radar
→ radar
← radar
C'est un palindrome !
```

Autre exemple :

```
terminal
Entrez un mot :
› python
→ python
← nohtyp
Pas un palindrome.
```

□□ Exercice 2 — Lettre dans un mot

□ **Objectif :** L'utilisateur entre une lettre. Vérifier si elle est présente dans le mot "informatique". Utiliser **find()** et tester si le résultat est différent de -1.

```
terminal
Entrez une lettre :
› a
'a' est dans 'informatique' !
```

Autre exemple :

```
terminal
Entrez une lettre :
› z
'z' n'est pas dans 'informatique'.
```

□□ Exercice 3 — Compter les voyelles

□ **Objectif :** Demander une phrase. Afficher combien de voyelles (a, e, i, o, u, y) elle contient. Parcourir lettre par lettre avec une boucle for.

```
terminal
Entrez une phrase :
› Bonjour tout le monde
Nombre de voyelles : 8
```

□□ Exercice 4 — Censurer un mot interdit

□ **Objectif :** Demander une phrase. Remplacer tous les mots "bête" (peu importe les majuscules) par "***". Utiliser **find()** pour localiser le mot, et **ch[:d] + ch[f:]** pour le supprimer.

```
terminal
Entrez une phrase :
› C'est une Bête idée !
Résultat : C'est une *** idée !
```

□□ Exercice 5 — Extraire les initiales

□ **Objectif :** Demander le nom complet. Afficher les initiales en majuscules séparées par des points.

```
terminal
Entrez votre nom complet :
› hammadi elaagerbi
Initiales : H.A
```

Challenge — 3 mots :

```
terminal
Entrez votre nom complet :
› youssef ben hsan
Initiales : Y.B.H
```

□□ Exercice 6 — Extraire l'extension d'un fichier

□ **Objectif :** Demander un nom de fichier. Afficher uniquement son extension. Utiliser **find(".")** pour trouver la position du point.

```
terminal
Nom de fichier :
› document.pdf
```

```
Extension : pdf
```

Autre exemple :

```
terminal
Nom de fichier :
› cours_python.docx
Extension : docx
```

□ 4. Parcours lettre par lettre avec while

On peut parcourir une chaîne caractère par caractère avec un indice et while — utile quand on veut construire un résultat lettre à lettre ou tester chaque caractère.

```python
ch = input('Phrase : ')
i  = 0
while i < len(ch) :
    # traiter ch[i]
    i = i + 1
```

□□ Exercice 7 — Compter majuscules et minuscules (ASCII)

□ **Objectif :** Demander une phrase. Afficher combien de lettres sont majuscules et combien sont minuscules. Utiliser **ord()** pour tester chaque caractère.

□ ord('A')=65, ord('Z')=90 → majuscule si 65 ≤ ord(c) ≤ 90
ord('a')=97, ord('z')=122 → minuscule si 97 ≤ ord(c) ≤ 122

```
terminal
Entrez une phrase :
› Bienvenue À Python !
Majuscules : 3
Minuscules : 13
```

□□ Exercice 8 — Somme des chiffres d'un nombre

□ **Objectif :** L'utilisateur entre un nombre (ex : 438). Calculer la somme de ses chiffres en parcourant le nombre caractère par caractère avec while.

terminal

```
Entrez un nombre :
› 438
4 + 3 + 8 = 15
Somme des chiffres : 15
```

□□ Exercice 9 — Remplacer les voyelles par □

□ **Objectif :** Demander une phrase. Construire et afficher la même phrase avec toutes les voyelles remplacées par □. Parcourir lettre par lettre avec while et construire le résultat par concaténation.

terminal

```
Entrez une phrase :
› Je suis étudiant
Résultat : J✶ s✶✶s ✶t✶d✶✶nt
```

□ Jeu — Le Mot Mystère

Le programme choisit un mot secret. L'utilisateur voit seulement des tirets. Il propose une lettre à la fois. Si elle est dans le mot → elle apparaît à la bonne place. Il gagne quand il découvre toutes les lettres.

terminal

```
Mot secret choisi : python
État : _ _ _ _ _ _
Propose une lettre :
› p
État : p _ _ _ _ _
Propose une lettre :
› y
État : p y _ _ _ _
...
Bravo ! Tu as trouvé : python
```

□ Récapitulatif — Chaînes de Caractères

□ Créer & Mesurer

Créer une chaîne	`mot = "Bonjour"`
Longueur	`len(ch)`
Accéder lettre i	`ch[i] (commence à 0)`
Dernière lettre	`ch[len(ch)-1]`

□□ Découper & Modifier

Extraire [debut:fin]	`ch[debut:fin] ← fin exclu`
Supprimer de d à f	`ch[:d] + ch[f:]`
Majuscules	`ch.upper()`
Inverser (boucle)	`for i in range(len(ch)-1, -1, -1):`
Coller des chaînes	`ch1 + ch2`
Texte + nombre	`ch + str(nombre)`

□ Tester & Chercher

Trouver position	`ch.find('mot') → -1 si absent`
Tester présence	`ch.find('x') != -1 → présent`
C'est un chiffre ?	`ch.isdecimal() → True / False`
Convertir str→int	`int(ch)`
Convertir int→str	`str(x)`
Code ASCII	`ord('A') → 65 chr(65) → 'A'`

□ Parcourir avec while

Initialiser	`i = 0`
Condition	`while i < len(ch) :`
Accéder lettre	`ch[i]`
Avancer	`i = i + 1`
Construire résultat	`res = res + ch[i]`