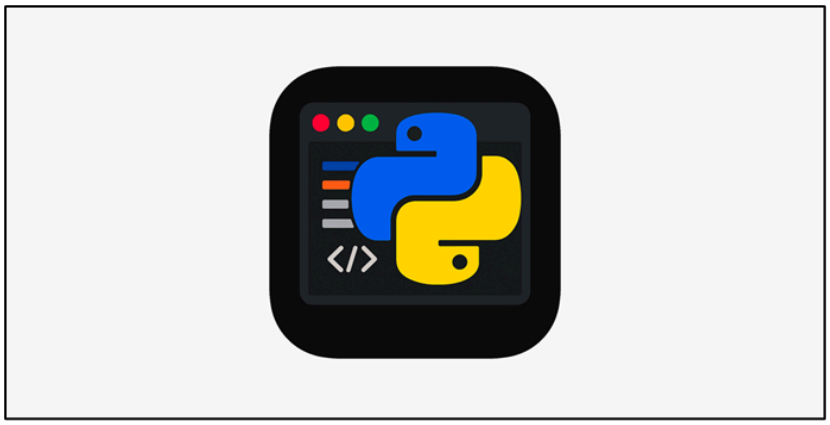


Learn Python

THE FUN WAY

<u>À la fin du cours, tu seras capable de :</u>

- Résoudre des problèmes scolaires qui nécessitent de la logique et de la méthode
- Lire et écrire du code compréhensible et efficace
- Transformer une idée en programme fonctionnel

“Programming isn’t about what you know; it’s about what you can figure out.”

Réalisé par Youssef Ben Hsan

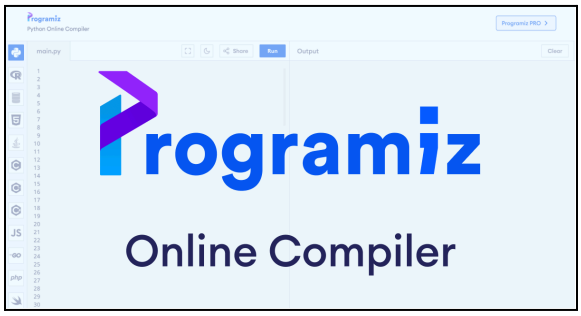
📞 26 159 209

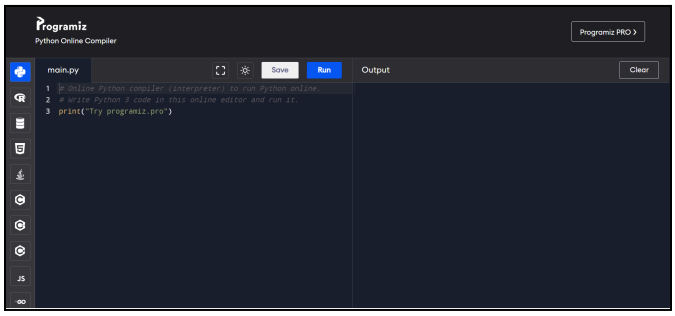
Notions de base à aborder :

- Qu'est-ce que le Python ?

- Comment écrire et exécuter du code ?

on va utiliser un des éditeur en ligne pour le moment (online)





Quelques activités de découverte :

le premier programme: `print()`

```
main.py
1   print("Bonjour le monde !")
```

=======>>

```
Bonjour le monde !
```

pour insérer un commentaire :

–#– pour entrer un commentaire

expliquant votre code

```
main.py
1   #votre commentaire
```

Essayez d'écrire ce text :

👉 utilisez \n (new line) pour rentrer à la ligne

👉utilisez \t tab pour tabulation

text |-------> text

```
Python is easy,
        I'm sure you will master it
                so quickly
```

Petit Projet : les variables

Complétez le programme sur votre poste qu'il affiche votre âge :

```
1  annee_naissance= 20…
2  annee_courante=20…
3  age=.....-.....
4  print("On est en: ",age)
```

Une Question Profonde

Comment la manière dont nous nommerons nos variables peut-elle influencer la compréhension et la maintenance de notre code ?

Ecrivez un programme avec input( ) qui affiche :

👉 utilisez input pour lire une valeur saisie par l'utilisateur au clavier.

comment?

```
1  mot = input("entrez une mot: ")
2  print(mot)
```

=====>

```
entrez un  mot: Hola
Hola
```

👉Écrivez un programme qui demande à l'utilisateur de saisir son prénom et son nom, puis les affiche comme montré dans l'aperçu :

```
entrez votre prenom : Youssef
entrez votre nom : Ben Hsan
Bonjour, Youssef Ben Hsan
```

🎯 Objectif : Identifier le type de chaque valeur

Utilise print(type(...)) pour afficher le type de chaque variable ci-dessous.

```
1  a=42
2  print( type( a ) )
```

====>

```
<class 'int'>
```

=>le type de la variable a (contenant 42) est int (= integer) , un nombre sans virgule.

games = 10

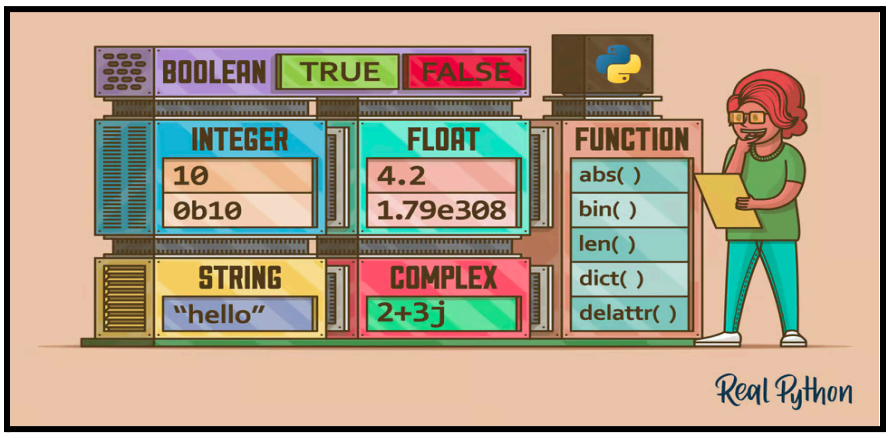
score = 99.5

username = "Ali"

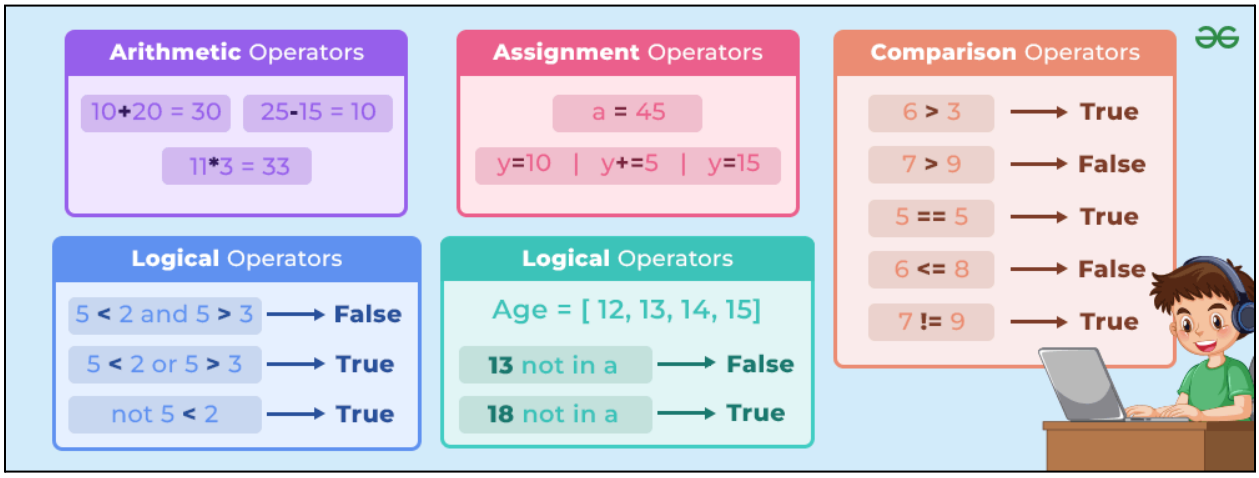
is_winner = False

temperature = 36.6

superhero = "Spider-Man"



Les opérateurs en Python



🎓 Questions à Choix Multiples (QCM)

1) Quelle est la manière correcte de créer une variable en Python ?

- A) var x = 10
- B) x := 10
- C) x = 10
- D) 10 = x

2) **Quel sera le résultat du code suivant ?**

```
x = "Hello”

print(x + " World!")
```

- **A) Hello World!**
- **B) HelloWorld!**
- **C) Hello + World!**
- **D) Erreur**

3) **Quel symbole est utilisé pour les commentaires en Python ?**

- **A) #**
- **B) //**
- **C) /***
- **D) --**

4) **Quel est le type de donnée de la valeur 3.14 en Python ?**

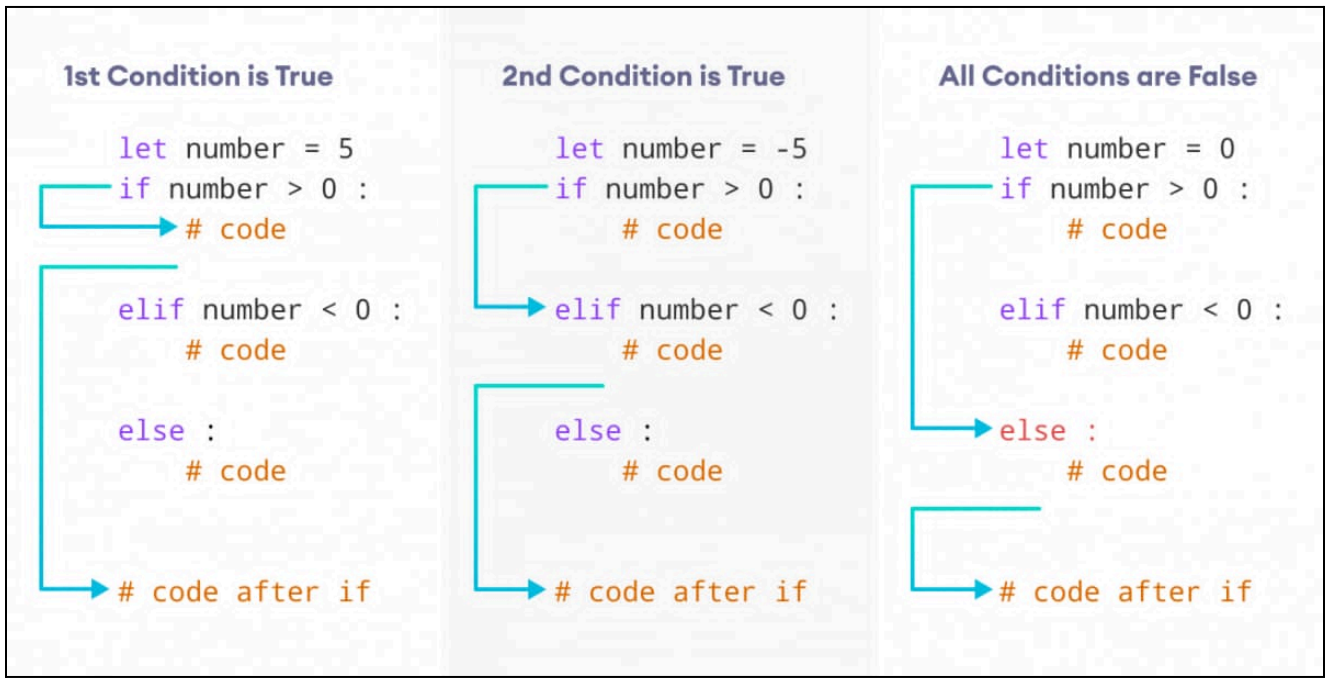
- **A) int**
- **B) str**
- **C) float**
- **D) bool**

🔍 Association de Vocabulaire

Associe chaque terme à sa définition correcte :

Terme	Définition
1. Variable	F) Une ligne ignorée par l'interpréteur, servant à expliquer le code.
2. Syntaxe	E) Une classification comme entier, flottant ou chaîne de caractères.
3. Chaîne de caractères	D) Une séquence de caractères utilisée pour du texte.
4. Commentaire	B) Un conteneur servant à stocker des valeurs.
5. Type de donnée	C) L'ensemble des règles qui définissent les combinaisons valides dans un code.

Comprendre les conditions if, elif, else



🧪 Exercice 1:

saisie une note (entre 0 et 20), puis :

- `"Excellent"` si la note ≥ 16
- `"Bien"` si la note ≥ 12
- `"Passable"` si la note ≥ 10
- `"Échec"` **sinon**

🧪 Exercice 2:

saisie l'âge d'un utilisateur, puis :

- **Affiche `"Mineur"` si l'âge est inférieur à 18**
- **Affiche `"Majeur"` si l'âge est entre 18 et 60**
- **Affiche `"Sénior"` si l'âge est plus grand que 60**

le type String

1 Compter le nombre de lettres dans une chaîne

🧮 Utilise len() pour afficher le nombre de caractères dans une phrase.

```
phrase = "Bonjour tout le monde !"
print("La phrase contient", _____, "caractères.")
```

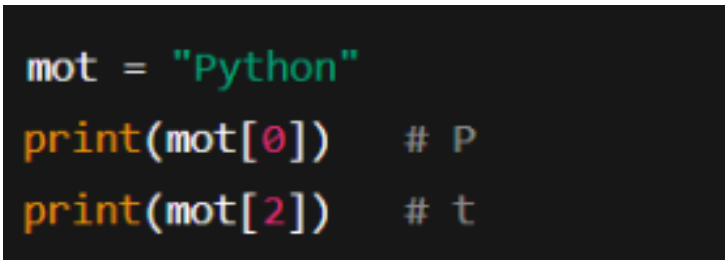
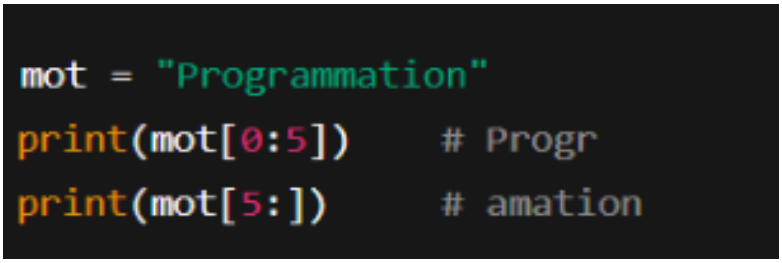
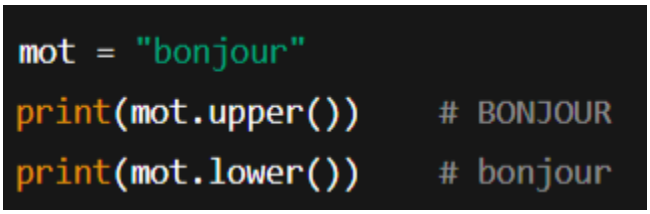
2 Extraire les 3 premières lettres et les 2 dernières

🧪 Utilise le slicing. [début : fin (pas inclu)] :

```
mot = "programmation"

debut = mot[ : ___]
fin = mot[___ : ]
print("Début :", debut)
print("Fin :", fin)
```

opérations sur les chaînes

	Operation
1-Accéder à une lettre par son indice:	
2- Extraire une partie de chaîne (slicing):	 ⚠️ Attention : le dernier caractère est exclu
3-Tout en majuscules ou minuscules:	
4-tester si un mot est majuscule ou minuscule	👉méthode:isupper() 👉méthode:islower( )

5-Remplacer une partie de texte:	`texte = "Je suis fatigué"` `print(texte.replace("fatigué", "en forme"))`

6-Tester si un mot est présent dans une chaîne	`mot = "python"`
	`phrase = "J'aime apprendre Python"`
	
	`# Affiche directement Vrai ou Faux`
	`print(mot.lower() in phrase.lower())`

7-Vérifier si c'est une lettre ou un chiffre	`char = "a"`
	`print(char.isalpha())  # True`
	
	`num = "8"`
	`print(num.isdigit())  # True`

8-Inverser une chaîne de caractères	`texte = "bonjour"`
	`inversé = texte[::-1]`
	`print(inversé)`

9-compter combien de fois un caractère	`texte = "programmation"`
	`nb = texte.count("m")`
	`print("La lettre 'm' apparaît", nb, "fois.")`

10-Retourne la première position de la chaîne ch1 dans la chaîne ch2	🔸 **Exemple 1 : mot trouvé** ➡️ ``` texte = "Bonjour tout le monde" position = texte.find("tout") print(position) ``` **8** 🔸 **Exemple 2 : mot non trouvé** ``` texte = "Bonjour le monde" position = texte.find("chat") print(position) ``` ➡️ **-1**

concaténation:

🧩 **Qu'est-ce que la concaténation ?**

👉**La concaténation, c'est coller plusieurs morceaux de texte ensemble.**

```python
prenom = "Ali"
message = "Bonjour " + prenom
print(message)
```

➡️**bonjour Ali**

⚠️ **Attention :**

👉Tu ne peux pas **concaténer** un nombre avec du texte directement :

```python
age = 18
print("Tu as " + age)   # ❌ Erreur !
```

🔧 **Il faut convertir le nombre en chaîne (`str`) :**

```python
age = 18
print("Tu as " + str(age))  # ✅
```

+Des exercices d'applications+

🧪 exercice1: chaîne palindrome

🎯 **Objectif :**

Demander à l'utilisateur d'écrire une mot, puis vérifier si la phrase est un **palindrome**

🧪**exemple:**

À l'endroit : `r - a - d - a - r`

À l'envers : `r - a - d - a - r`

donc palindrome✅

🧪 exercice2: Vérifier si une lettre est dans un mot :

📍 **L'utilisateur entre une lettre → dire si elle est présente dans `"informatique"`.**

🧪 exercice3:Compter les voyelles

Question :

Demande à l'utilisateur une phrase et affiche combien de voyelles (a, e, i, o, u, y) elle contient.

Exemple d'exécution :

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

🧪 exercice4:Censure un mot interdit

Question :

Demande à l'utilisateur une phrase et remplace tous les mots "bête" (ou variantes avec majuscules) par "*".**

Exemple d'exécution :

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

🧪 exercice5:Extraire les initiales

Question :

Demande le nom complet de l'utilisateur (prénom + nom) et affiche ses initiales en majuscules.

Challenge 1 : un nom composé de 2 mots :

Exemple d'exécution :

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

Challenge 2 : un nom composé de 3 mots:

Exemple d'exécution :

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

🧪 exercice6:Extraire extension :

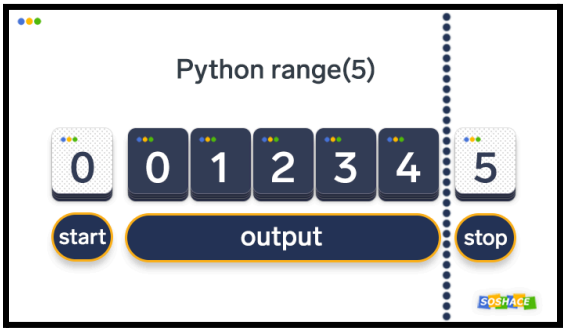
- Demande à l'utilisateur d'entrer un nom de fichier
- Affiche l'extension du fichier

🧪 Exemple :

Nom de fichier saisi : abc.java

Sortie : java

print(“Votre nom”) 50 fois



👉 **La boucle for est utilisée pour répéter un bloc de code un certain nombre de fois .**

➡️ **On va voir comment ça marche avec des exercices simples !**

🧪 **exercice1:**

🎯**Objectif :** Afficher les nombres de 1 à 10 à l’aide d’une boucle `for`.

```
# À compléter
for i in _____:
    print(_____)
```

🧪 **exercice2:**

🎯**Objectif :** Calculer la somme des nombres

➕ Calcule la somme des nombres de 1 à 100.

💡 **Résultat attendu :** `5050`

```
# Étape 1 : Initialiser la somme
somme = _____

# Étape 2 : Parcourir les nombres de 1 à 100
for i in range(_____, _____):

    # Étape 3 : Ajouter chaque nombre à la somme
    _____ += _____

# Étape 4 : Afficher le résultat
print("Somme =", _____)
```

🧪 exercice3:

🎯**Objectif :Afficher les lettres d'un mot**

🔤 Affiche chaque lettre du mot "Python" sur une ligne différente.

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

🧪 exercice4:

🎯**Objectif : Créer une pyramide d'étoiles**

⭐ Affiche une pyramide comme ceci :

```
*
**
***
****
*****
```

```
for i in range(1, 6):
    print("*" * _____)
```

+Puis à l'envers!

🧠 Application 1 : Compte à rebours pour le lancement 🚀

Question : 🔥

Écris un programme qui affiche les nombres de 10 à 1,

puis affiche "Décollage !".

Exemple d'exécution :

```
10
9
8
...
1
Décollage !
```

🧠 Application 2 : Inverser une phrase 🔁

Question : 🔥🔥

Demande à l'utilisateur une phrase, et affiche-la **lettre par lettre à l'envers**.

Exemple d'exécution :

```
Entrez une phrase : Bonjour
r
u
o
j
n
o
B
```

🧠 Application 3:

🎯Objectif : **Compter une lettre dans une phrase**

L'utilisateur entre une phrase et une lettre. Affiche combien de fois la lettre apparaît (en ignorant les majuscules/minuscules).

```
> Phrase : Python est populaire
> Lettre : p
La lettre 'p' apparaît 2 fois.
```

🧠 Application 4 : Triangle inversé 🔻

🎯 **Objectif** : Utiliser les boucles `for` pour afficher une forme inversée d'un triangle.

📝 **Consigne** : 🔥🔥🔥

Écris un programme qui demande à l'utilisateur un nombre n et qui affiche un triangle inversé d'étoiles avec n lignes.

📌 Exemple pour `n = 4` :

```
markdown

****
***
**
*
```

🧠Application 5: Calcul de moyenne des notes 📊

Question :🔥🔥🔥🔥

1) Demande à l'utilisateur combien de notes il veut entrer.

2) récupère chaque note, affiche-les une à une et à la fin affiche la moyenne.

Exemple d'exécution :

```
Combien de notes ? 3
Note 1 : 14
Note 2 : 17
Note 3 : 15
Moyenne : 15.33
```

🧠Application 6 : Tableau de multiplication magique ✨

🎯 **Objectif** : Utiliser des **boucles imbriquées** pour afficher une table de multiplication.

📝 **Consigne** :🔥🔥🔥🔥🔥

Écris un programme qui demande à l'utilisateur un nombre n, puis affiche un **tableau de multiplication** de **1 à n**.

📌 Exemple pour `n = 4` :

```
1   2   3   4
2   4   6   8
3   6   9   12
4   8   12  16
```

Complétez :

```
n = int(input("Entrez un nombre : "))

for i in range(1, _____ + 1):
    for j in range(1, _____ + 1):
        print(_____, end="\t")
    print()
```

➕Maintenant **Colorier la diagonale principale (i == j) avec un symbole spécial :**

🔎 **Exemple pour** `n = 5` **:**

```
💥  2   3   4   5
2   💥  6   8   10
3   6   💥  12  15
4   8   12  💥  20
5   10  15  20  💥
```

🆕 Nouveau concept

🔐 Mission Sécurité : Trouve le bon mot de passe !

Problématique : Comment faire pour que l'utilisateur répète sa tentative jusqu'à ce qu'il entre le mot de passe correct ?

(on ne peut pas utiliser for car on ne sait pas le nombre de fois exacte à répéter la demande d'écrire le mot de passe)

Solution ?  :

while loop✅


```python
mot_de_passe = ""

# Le bon mot de passe est : python123
while mot_de_passe != "_____":
    mot_de_passe = input("Entrez le mot de passe : ")
    if mot_de_passe != "_____":
        print("Mot de passe incorrect ❌")

print("Accès autorisé ✅")
```

🧠La boucle `while` – On répète tant que la condition est vraie

Tu te souviens du type de données bool ?

Une condition comme [x < 5] retourne True ou False.

➡️ Tant que c'est True, Python continue à exécuter le bloc.

```
while [condition]:
        ...
        le bloc
        (ex: print("Hello"))
        ...
```

Passant au pratique:

🧪 exercice1:

🎯Objectif :Compter de 1 à 10

avec la boucle while

```
i = 1
while _____:
    print(i)
    _____
```

🧪 exercice2:

🎯Objectif :**Saisie jusqu'à un nombre pair** **modulo %**

—>Redemande un nombre tant que l'utilisateur entre un **nombre impair**.

🧪 exercice3:

🎯Objectif : **Trouver la première puissance de 2 supérieure à 1000**

📐explication : 2*2 = 4 <1000 continue

2*2*2 = 8 <1000 continue

.

.

2*2*2*2….*2 =1024 >1000 stop

🔷 Exercice : Devinette magique (GAME 😀)

Le programme choisit un nombre mystère entre 1 et 20.
L'utilisateur a un nombre illimité d'essais, jusqu'à ce qu'il trouve le bon nombre.
À chaque mauvaise réponse, le programme donne un indice :

- "Trop petit" si le nombre est inférieur
- "Trop grand" s'il est supérieur

Lorsque l'utilisateur trouve, le programme affiche :

- "Bravo ! Tu as trouvé en X essais !"

```
Un nombre mystère entre 1 et 20 a été choisi...
Devine : 10
Trop grand !
Devine : 5
Trop petit !
Devine : 7
Bravo ! Tu as trouvé en 3 essais !
```

🧪 exercice5:

🎯Objectif : **Compter majuscules et minuscules Ascii**

Donne une phrase. Affiche combien il y a de **lettres majuscules** et **minuscules.**

```
> Phrase : Bienvenue À Python !
Majuscules : 3
Minuscules : 13
```

📐explication mot **Ascii** : **ASCII** est un **code** qui permet à l'ordinateur de **représenter des caractères** (lettres, chiffres, symboles) sous forme de **nombres.**

👉méthode:ord( )

👉méthode:chr( )

```
print(ord('A'))   # Affiche 65
print(chr(65))    # Affiche 'A'
```

🧪 exercice6:Somme des chiffres d'un nombre

🎯Objectif :**Somme des chiffres d'un nombre**

L'utilisateur entre un nombre (ex: "438"). Calcule la **somme de ses chiffres.**

```
> Nombre : 438
Somme des chiffres : 15
```

🧪 exercice7:

🎯Objectif : **Remplacer les voyelles par * dans une phrase**

Affiche la phrase, mais toutes les **voyelles sont remplacées par** *.

```
> Phrase : Je suis étudiant
Sortie : J* s**s *t*d**nt
```

-LISTS-

📦 Les Listes en Python : Le Coffre à Jouets Magique !

Imagine que tu as un coffre à jouets 🧸 où tu peux mettre n'importe quoi :

des voitures, des bonbons et même un autre coffre !

Ce coffre magique, c'est une liste en Python.

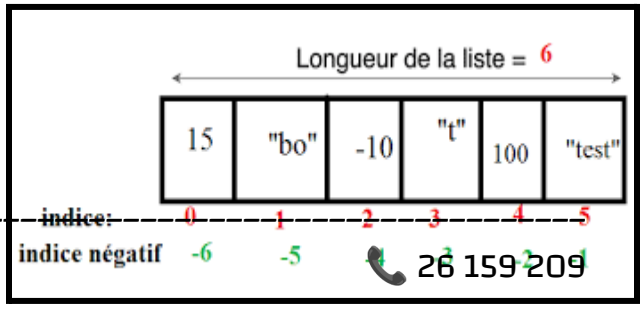
Tu peux y ranger plein de choses dans un ordre précis,

et les ressortir quand tu veux.

*Très similaire à une chaîne de caractères, mais au lieu de ***caractères***, on a des informations de types différents stockées dans un ordre précis.*

📦 Une liste peut contenir :

- des nombres : `[1, 2, 3]`



- des textes : `["a", "b", "c"]`
- des booléens : `[True, False]`
- ou même mélanger tout : `[1, "salut", True]`

différentes opérations sur les listes :

	Operation
1. Créer une liste	`fruits = ["pomme", "banane", "orange"]`
2. Accéder à un élément	`print(fruits[0])  # Affiche "pomme"`
3. Modifier un élément	`fruits[1] = "kiwi"` `print(fruits)  # ["pomme", "kiwi", "orange"]`
4. Ajouter un élément à la fin	`fruits.append("fraise")` `print(fruits)  # ["pomme", "kiwi", "orange", "fraise"]`
5. Insérer un élément à une position précise	`fruits.insert(1, "ananas")` `print(fruits)  # ["pomme", "ananas", "kiwi", "orange", "fraise"]`

6. Supprimer un élément par sa valeur	`fruits.remove("kiwi")` `print(fruits)  # ["pomme", "ananas", "orange", "fraise"]`
7. Supprimer un élément par son index	`del fruits[0]` `print(fruits)  # ["ananas", "orange", "fraise"]`
8. Vider une liste	`fruits.clear()` `print(fruits)  # []`

9. Parcourir une liste	`nombres = [1, 2, 3]`
	`for n in nombres:`
	`    print(n)`

10. Trier une liste	`notes = [14, 9, 18, 12]`
	`notes.sort()`
	`print(notes)  # [9, 12, 14, 18]`

11. Concaténer deux listes	`liste1 = [1, 2]`
	`liste2 = [3, 4]`
	`resultat = liste1 + liste2`
	`print(resultat)  # [1, 2, 3, 4]`

12. Vérifier la présence d'un élément	`if "banane" in fruits:` `    print("Oui, il y a une banane.")`

1. Créer une liste de fruits et les afficher un par un

Crée une liste `fruits` contenant 3 fruits, puis affiche chaque fruit sur une ligne.

```
pomme
banane
kiwi
```

2. Accéder au premier et au dernier fruit

Affiche le premier et le dernier élément de la liste `fruits`.

```
pomme
kiwi
```

3. Modifier un élément de la liste

Change le deuxième fruit en "orange" et affiche toute la liste.

```
['pomme', 'orange', 'kiwi']
```

4. Ajouter et supprimer des fruits

Ajoute "fraise" à la liste et supprime "pomme" s'il est présent.

```
['orange', 'kiwi', 'fraise']
```

5. Vider une liste de nombres

Vide complètement la liste `[1, 2, 3, 4, 5]` et affiche-la.

```
[]
```

6. Afficher le carré de chaque nombre

Parcours la liste `[2, 4, 6, 8]` et affiche le carré de chaque élément.

```
4
16
36
64
```

7. Somme des nombres pairs

Calcule la somme des nombres **pairs** dans `[3, 7, 10, 15, 20]`.

```
30
```

8. Vérifier la présence d'un mot

Demande un mot à l'utilisateur. Indique s'il est dans `["python", "java", "c++", "ruby"]`.

```
Langage trouvé !
```

(si l'utilisateur tape "Java")

9. Créer toutes les paires possibles

Affiche toutes les paires possibles entre noms dans `["Alice", "Bob", "Charlie"]`.

```
[('Alice', 'Bob'), ('Alice', 'Charlie'), ('Bob', 'Charlie')]
```

10. Fusionner deux listes et supprimer les doublons

Fusionne `["chat", "chien"]` et `["chien", "lapin"]` sans doublons.

```
['chat', 'chien', 'lapin']
```

11. Compter les répétitions d'un mot

Compte le nombre de fois que "bonjour" apparaît dans `"bonjour bonjour salut bonjour"`.

```
3
```

12. Multiplier chaque nombre par 3

Crée une nouvelle liste avec les éléments de `[1, 2, 3, 4, 5]` multipliés par 3.

```
[3, 6, 9, 12, 15]
```

13. Garder les mots qui commencent par une voyelle

À partir de `["arbre", "maison", "vélo", "avion", "bateau"]`, garde les mots commençant par une voyelle.

```
['arbre', 'avion']
```

14. Inverser une liste

Inverse `[1, 2, 3, 4]` de deux façons : avec `.reverse()` puis avec une boucle.

```
[4, 3, 2, 1]
```

15. Afficher une matrice et sa diagonale

Crée une matrice 3×3 avec les chiffres de 1 à 9, puis affiche la diagonale.

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
1
5
9
```

16. Supprimer les doublons tout en gardant l'ordre

Supprime les doublons de `[5, 3, 5, 2, 3, 1, 4]`.

```
[5, 3, 2, 1, 4]
```

17. Mettre une majuscule à chaque mot

Pour chaque phrase dans `["le chat dort", "python est cool"]`, affiche la version avec chaque mot en majuscule.

```
Le Chat Dort
Python Est Cool
```

18. Analyser un texte et afficher les mots les plus fréquents

À partir d'un texte, supprime la ponctuation, compte la fréquence des mots et affiche les 3 plus fréquents.

```
[('bonjour', 3), ('le', 2), ('chat', 1)]
```

19. Grille 5×5 avec diagonale à 1

Crée une grille 5×5 avec des `0`, mais remplace les éléments de la diagonale par `1`.

```
[1, 0, 0, 0, 0]
[0, 1, 0, 0, 0]
[0, 0, 1, 0, 0]
[0, 0, 0, 1, 0]
[0, 0, 0, 0, 1]
```

20. Mini-jeu : Deviner les lettres d'un mot

Un mot aléatoire est choisi, l'utilisateur a 5 essais pour deviner ses lettres une par une.

```
Devine une lettre : e
Mot : _ _ _ _ e
Devine une lettre : m
Mot : _ _ m m e
...
Mot final : pomme
```