CM 1 - Bases

Programmation Python – Master 1 MAS

Romain Tavenard

2026

romain.tavenard@univ-rennes2.fr

DANS CE COURS

- Environnement de développement : VS Code
- 1 TD (/ projet) = 1 Script Python
- Structure de votre code

```
# 1. Les imports

# 2. Les fonctions

# 3. Les tests
```

- Polycopié associé : https://rtavenar.github.io/poly_python/

UNIVERSITÉ RENNES 2

VARIABLES & STRUCTURES DE CONTRÔLE

1 ○ ○

Pour définir une variable en Python :

```
chaine_exemple = "une valeur"
entier_exemple = 123
autre_entier = entier_exemple ** 2  # Élévation à la puissance
encore_un_autre = entier_exemple % 2  # Opérateur "modulo"
un_flottant_ici = entier_exemple / 2  # Division à virgule
un_entier_la = entier_exemple // 2  # Division entière
```

Le type dépend de la valeur affectée.

Il vous est demandé d'utiliser des noms de variable explicites.

STRUCTURES CONDITIONNELLES

UNIVERSITÉ RENNES 2

Syntaxe du `if ... elif ... else`

```
if condition:
  # Code si Vrai
elif autre_condition:
  # Code si Vrai
else:
  # Code sinon
```

Attention à l'indentation !

Les clauses `elif` et `else` sont optionnelles.

`condition` et `autre_condition` sont des expressions booléennes.

EXPRESSIONS BOOLÉENNES

Exemples d'expressions booléennes :

- `x > 10`
- `y <= 8`
- `z == "a"`
- `a != "a"`
- `(a > 10) or (b == 5)`
- `(a > 10) or c`
  où `c` est une variable de type `bool`

EXERCICE

UNIVERSITÉ RENNES 2

Écrivez une expression conditionnelle, qui à partir d'une température d'eau stockée dans une variable `c` *affiche si l'eau à cette température est à l'état liquide, solide ou gazeux.*

Boucles for

Syntaxe du **for**

```
for i in range(10):  # i allant de 0 à 9
  # Code à répéter pour toutes les valeurs de i

for i in range(5, 20):  # i allant de 5 à 19
  # Code à répéter pour toutes les valeurs de i
```

Attention à l'indentation !

En Python, la boucle **for** permet d'itérer sur une séquence de valeurs.

Boucles while

Syntaxe du **while**

```
while condition:
  # Code à répéter tant que condition vaut Vrai
```

Attention à l'indentation !

`condition` est une expression booléenne (comme pour le **if**).

Quelle boucle choisir ?

Pour choisir entre `for` et `while`

- Si nombre d'itérations fixé ⇒ `for`
- Si on veut répéter le même traitement pour toutes les valeurs d'une séquence (`range`, liste, dictionnaire) ⇒ `for`
- Sinon ⇒ `while`

FONCTIONS

2

Définir une fonction en Python (1/2)

Une fonction a (peut avoir) des paramètres et une valeur de retour :

```python
def affiche_pair(n):
  for i in range(n):
    if i % 2 == 0:
      print(i)
```

- Nom de la fonction : `affiche_pair`
- Paramètre(s) / Argument(s) : `n` (1 seul paramètre / argument)
- Valeur de retour : Pas de valeur de retour ici (pas de `return`)

Définir une fonction en Python (2/2)

Une fonction a (peut avoir) des paramètres et une valeur de retour :

```python
def est_pair(i):
  return (i % 2 == 0)
```

- Nom de la fonction : `est_pair`
- Paramètre(s) / Argument(s) : `i` (1 seul paramètre / argument)
- Valeur de retour : valeur booléenne indiquant si `i` est pair

APPEL DE FONCTION EXISTANTE

UNIVERSITÉ RENNES 2

Deux façons d'appeler une fonction en Python :

1. Si elle retourne une valeur que l'on souhaite utiliser :

```
x = est_pair(5)  # x est de type bool

# ou bien
print(est_pair(5))

# ou encore
if est_pair(5):
  print("Youhou")
```

2. Si elle ne retourne rien ou que sa valeur de retour ne nous intéresse pas :

```
affiche_pair(5)
```

EXERCICE

« Écrivez une fonction en Python qui affiche tous les termes plus petits que 1000 de la suite (u_n) définie comme :

$$\begin{aligned} u_0 &= 2 \\ \forall n \geq 1,\ u_n &= u_{n-1}^2 \end{aligned}$$

»

Les paramètres optionnels

Une fonction peut avoir des paramètres optionnels :

```python
def est_multiple(i, p=2):
  return (i % p == 0)

print(est_multiple(9, 3))  # Affiche True
print(est_multiple(9))     # Affiche False
```

Les paramètres optionnels sont toujours placés après les paramètres non optionnels.

UNIVERSITÉ RENNES 2

CHAÎNES DE CARACTÈRES

3

LE TYPE STR

Il existe en Python de nombreuses fonctions pour manipuler simplement les chaînes de caractères :

```python
x = "une chaîne de caractères "
y = "et une autre"
print(len(x))  # Affiche 25
print(x + y)   # Affiche une chaîne de caractères et une autre
print(y * 3)   # Affiche et une autreet une autreet une autre

age = 25
print(f"L'âge est de {age} ans.")  # Affiche L'âge est de 25 ans.
```

Les méthodes de la classe str

Code	Action
`ch.count(sub)`	Compte les occurrences
`ch.endswith(suffix)`	Teste la fin de la chaîne
`ch.startswith(prefix)`	Teste le début de la chaîne
`ch.find(sub)`	Indice du début de la première occurrence
`ch.rfind(sub)`	Indice du début de la dernière occurrence
`ch.lower()`	`ch` en minuscules
`ch.upper()`	`ch` en majuscules
`ch.replace(old, new)`	Copie de `ch` dans laquelle toutes les occurrences de `old` ont été remplacées par `new`
`ch.split(sep=None)`	Découpe `ch` à chaque occurrence de `sep`

EXERCICE

UNIVERSITÉ RENNES 2

Écrivez une fonction qui prenne en argument deux chaînes de caractères `s` *et* `prefix` *et retourne le nombre de mots de la chaîne* `s` *qui débutent par la chaîne* `prefix`*.*