Variables booléennes et structures conditionnelles

Une **variable booléenne** (`bool` en python) peut prendre deux valeurs : `True` ou `False`. (vrai ou faux)

Ex :
```
b = True
print(type(b))        # Affiche 'bool'
```

Les **comparaisons** sont des expressions qui sont traduites à l'exécution en une valeur booléenne.

Opération	Opérateur	Exemples
Égal	`==`	`5 == 5 # True` `5 == 6 # False`
Différent	`!=`	`5 != 5 # False` `5 != 6 # True`
Strictement supérieur	`>`	`6 > 5  # True` `5 > 6  # False`
Strictement inférieur	`<`	`6 < 5  # False` `5 < 6  # True`
Supérieur ou égal	`>=`	`5 >= 5 # True` `5 >= 6 # False`
Inférieur ou égal	`<=`	`5 <= 6 # True` `6 <= 5 # False`

Ex :
```
a = 6
b = (a % 2) == 0      # a est-elle paire ?
c = (a + 4) != 10     # la somme de a et 4 est-elle différente de 10 ?
print(b, c)           # True False
```

Les **structures conditionnelles** permettent l'exécution ou non d'instructions si des conditions sont vraies ou fausses. Ils existent en 3 formes :

En pseudo-code	Exemple en Python

SI condition ALORS
... instructions si la condition est vraie ...
	`n = ... # à compléter`
	`if n >= 0 :`
	`  print("Le nombre est positif.")`

SI condition ALORS
... instructions si la condition est vraie ...
SINON
... instructions si la condition est fausse ...
	`n = ... # à compléter`
	`if n >= 0 :`
	`  print("Le nombre est positif.")`
	`else :`
	`  print("Le nombre est négatif.")`

SI condition_1 ALORS
... instructions si la condition_1 est vraie ...
SINON SI condition_2 ALORS
... instructions si la condition_2 est vraie ...
SINON SI condition_...
...
SINON
... instructions si aucune condition n'est vraie...
	`n = ... # à compléter`
	`if n > 0 :`
	`  print("Le nombre est positif.")`
	`elif n < 0:`
	`  print("Le nombre est négatif.")`
	`elif n == 0:`
	`  print("Le nombre est nul.")`
	`else :`
	`  print("!Erreur de MATHS!")`

Il est important de noter que les instructions à exécuter dans la condition sont **<u>indentées</u>** : elles sont décalées vers la droite à l'aide de la touche tabulation.

```
if n >= 0 :
    print("Code exécuté si n >= 0")
print("Code exécuté tout le temps")
```

On peut combiner des expressions ou des variables booléennes avec les **opérateurs logiques ET** (`and`) et **OU** (`or`).
On peut obtenir le contraire d'une variable ou expression booléenne avec l'**opérateur logique NON** (`not`).

Ci-dessous, les **tables de vérités** des opérateurs logiques ET, OU et NON.

a	b	a and b
True	True	True
True	False	False
False	True	False
False	False	False

a	b	a or b
True	True	True
True	False	True
False	True	True
False	False	False

a	Not a
False	True
True	False

Ex :
```
a = 3.5
b = a < 5 and a >= 0            # True
c = a < 0 or a == 3.5           # True
d = not (a < 5 and a >= 0)      # False
e = not (a < 0 or a == 3.5)     # False
f = not True                    # False
g = not False                   # True
```

Comme en mathématiques, les opérateurs booléens ont des **priorités opératoires**. Le NON est prioritaire - suivi du ET puis enfin du OU. Pour éviter tout problème, il est recommandé d'utiliser des parenthèses.

Dans le cas des conjonctions (si les dernières opérations a effectuer sont des ET logique), dès lors qu'une première condition est fausse, les autres conditions ne sont pas vérifiées puisque le résultats sera faux de toute façon. On appelle cela une **évaluation paresseuse**.

Ex :

```
if 4 > 5 and 0/0 == 0:     # 4 > 5 est faux, donc 0/0 == 0 n'est pas exécuté
    print("Boom")          # et heureusement, on ne peut pas diviser par 0.
else:
    print("Rien de cassé")
```

Attention, une variable crée dans les instructions sous une condition est accessible en dehors de la condition. (source)

Ex :
```
if True:
    zzz = 999
print(zzz) # affiche 999
```

Exercice 1 : Prédict, run, investigate

Donner la valeur des expressions booléennes suivantes sans exécuter le code, puis exécuter pour vérifier.

`2 >= 3`
`5 == 5`
`4 > 5`
`2.5 <= 0.25`
`6 != 2`
	`25/5 + 2 == 7`
	`6*3 == 36/2`
	`not True`
	`not False`
	`True and False`
		`True or False`
		`True and True`
		`True or True`
		`False or (True and False)`
		`not (True and False)`

Exercice 2 : Prédict, run, investigate

Sans exécuter, qu'affichent les codes suivants ? Vérifier en exécutant le code.

```
# Code 1
a = 3.6
b = 5.4
if a > b :
    print("Le plus grand nombre est", a)
else:
    print("Le plus grand nombre est", b)
```

```
# Code 2
n = 7
if n % 2 == 0 :
    print(n, "est pair.")
```

```
# Code 3
n = -4
if n > 0 or n < -4 :
    print("Cool.")
elif n < -3 and n > -10:
    print("Super.")
else :
    print("Pas ouf.")
print("Lunaire.")
```

```
# Code 4
a, b = 1, -2    # Double initialisation
if a < 0:
    if 5 < b:
        print("p1")
    else:
        print("p2")
else :
    print("p3")
    if 4 > b:
        print("p4")
```

Exercice 3 : Modify

1) En modifiant le code 2 de l'exercice 2, écrire un programme qui affiche si un nombre est pair ou impaire.

2) En modifiant le code 3 de l'exercice 2, écrire un programme qui affiche qu'il fait chaud si le nombre est supérieur à 30, froid si le nombre est inférieur à 10 et tiède sinon.

3) En modifiant le code 1 de l'exercice 2, écrire un programme pour afficher le maximum entre 3 nombres.

Exercice 4 : Make

1) Écrire un programme qui indique si la somme 0.1 + 0.1 + 0.1 est égale à 0.3 .

2) Écrire un programme qui demande un nombre et indique s'il est divisible par 5 et 11 ou non.

3) Écrire un programme qui demande deux nombres puis vérifie que leur somme n'est pas égale à 21.

4) Écrire un programme qui demande à l'utilisateur une année et affiche si elle est bissextile ou non. Les années sont bissextiles si elles sont multiples de quatre, mais pas si elles sont multiples de cent, à l'exception des années multiples de quatre cents qui, elles, sont également bissextiles.

5) Écrire un programme qui demande de deviner le nombre mystère et indique si l'on a gagné. On pourra utiliser les lignes de codes suivantes :

```
import random
n = random.randint(0,1)
```

6) Écrire un programme qui demande à l'utilisateur son revenue imposable et calcule le montant de ses impôts pour une personne célibataire. (Aide : https://www.service-public.fr/particuliers/vosdroits/F1419)