

Épisode II : Structures conditionnelles

EXERCICE 1

Taper les instructions suivantes et prenez le temps d'expliquer précisément l'évolution des variables pendant l'exécution pas à pas de ces instructions.

```
i = 9
j = 0
b = i < j      # b ne contient donc pas un entier, mais True ou False
b = i != 9
b = i == 9
bi = i % 2 == 0 or i % 3 == 0
bj = j % 2 == 0 or j % 3 == 0
b = not (bi and bj)
```

EXERCICE 2

Parmi les expressions suivantes, quelles sont celles qui valent `True` *si et seulement si*^a les trois variables `a`, `b`, `c` ont des valeurs toutes différentes ? Rayer les expressions incorrectes.

- `a != b and b != c`
- `a != b and b != c and a != c`
- `a != b or b != c`
- `a != b or b != c or a != c`

a. si et seulement si signifie que l'un est vrai si l'autre est vrai, et que l'un est faux si l'autre est faux.

EXERCICE 3

Écrire une expression qui vaut `True` si `x` appartient à $[0, 5[$ et `False` dans le cas contraire.

EXERCICE 4

Parmi les expressions suivantes, quelles sont celles qui valent `True` si l'entier `n` est pair et `False` dans le cas contraire ? Rayer les mauvaises réponses.

- `n == 0 % 2`
- `1 != 2 % n`
- `n // 2 == 0`
- `n % 2 != 1`
- `0 != n % 2`
- `n % 2 == 0`

Écrire une fonction `pair(n)` qui teste si `n` est pair ; la valeur calculée doit être *booléenne*, c'est-à-dire égale à `True` ou `False`. Testez-la.

EXERCICE 5

Écrire un script python permettant de calculer l'intersection de deux intervalles $[a, b]$ et $[c, d]$.

EXERCICE 6

Donner les valeurs des variables x et y après exécution de l'exemple suivant pour x valant 1, puis pour x valant 8.

```
if x % 2 == 0:
    y = x // 2
else:
    y = x + 1
x = x + 1
```

Même question pour chacun des codes suivants :

```
if x % 2 == 0:
    y = x // 2
    y = x + 1
x = x + 1
```

```
if x % 2 == 0:
    y = x // 2
else:
    y = x + 1
    x = x + 1
```

```
if x % 2 == 0:
    y = x // 2
else:
    y = x + 1
x = x + 1
```

EXERCICE 7

Écrire une fonction `compare(a, b)` qui retourne -1 si $a < b$, 0 si $a = b$, et 1 si $a > b$. Testez-la. Donner plusieurs versions de cette fonction.

EXERCICE 8

Écrire une fonction `max2(x, y)` qui calcule et retourne le maximum de deux nombres x et y . Attention : bien appeler cette fonction `max2`, car la fonction `max` est prédéfinie en python.

EXERCICE 9

Écrire une fonction `max3(x, y, z)` qui calcule et retourne le maximum de trois nombres x, y, z . Donner plusieurs versions de cette fonction dont une utilise la fonction `max2`.