

Autotest n°1 – correction

EXERCICE 1 : Revoir les tests 1 et 2.

EXERCICE 2 : Revoir la feuille d'exercices Python n°1.

EXERCICE 3 : Revoir la feuille d'exercices d'algorithmique n°1.

EXERCICE 4 : Convertir en base 10 les nombres suivants, en montrant la technique utilisée.

1) 1100101_2

2) 1001110_2

3) 1312_4

4) $A9_{16}$

Solution :

64	32	16	8	4	2	1
1	1	0	0	1	0	1
1	0	0	1	1	1	0

1) $1100101_2 = 64 + 32 + 4 + 1 = 101_{10}$

2) $1001110_2 = 64 + 8 + 4 + 2 = 78_{10}$

3) $1312_4 = 1 \times 64 + 3 \times 16 + 1 \times 4 + 2 \times 1 = 64 + 48 + 4 + 2 = 118_{10}$

4) $A9_{16} = 10 \times 16 + 9 = 169_{10}$

EXERCICE 5 : Convertir dans la base demandée les nombres suivants, en montrant la technique utilisée.

1) 200 en base 2

2) 179 en base 2

3) 197 en base 4

4) 127 en base 16

Solution :

1) $200 = 128 + 72 = 128 + 64 + 8 = 11001000_2$

2) $179 = 128 + 51 = 128 + 32 + 19 = 128 + 32 + 16 + 2 + 1 = 10110011_2$

$197 : 4 = 49$ reste 1
 $49 : 4 = 12$ reste 1
 $12 : 4 = 3$ reste 0
 $3 : 4 = 0$ reste 3
 3 0 1 1

$127 : 16 = 7$ reste 15
 $7 : 16 = 0$ reste 7
 7 F

EXERCICE 6 : Un entier est représenté en hexadécimal avec 3 chiffres. Sans connaître sa valeur, combien de bits faut-il au minimum pour être sûr de pouvoir représenter ce nombre en binaire ?

Solution : Le plus grand nombre qu'on peut écrire avec 3 chiffres en hexadécimal est FFF_{16} . Or $FFF_{16} = 15 \times 16^2 + 15 \times 16 + 15 = 4095_{10}$.

En binaire, $4095_{10} = 111111111111_2$.

Il faut donc 12 bits pour être sûr de pouvoir écrire le nombre.

EXERCICE 7 : Établir les tables de vérité des expressions booléennes suivantes :

1) $((\text{non } a) \text{ ou } b) \text{ et } (a \text{ ou } (\text{non } b))$

Solution :

a	b	$\text{non } a$	$\text{non } b$	$(\text{non } a) \text{ ou } b$	$a \text{ ou } (\text{non } b)$	$((\text{non } a) \text{ ou } b) \text{ et } (a \text{ ou } (\text{non } b))$
0	0	1	1	1	1	1
0	1	1	0	1	0	0
1	0	0	1	0	1	0
1	1	0	0	1	1	1

2) a ou non((non a) ou (non b))

Solution :

a	b	non a	non b	(non a) ou (non b)	non((non a) ou (non b))	a ou non((non a) ou (non b))
0	0	1	1	1	0	0
0	1	1	0	1	0	0
1	0	0	1	1	0	1
1	1	0	0	0	1	1

EXERCICE 8 : On considère la fonction anonyme, présentée ci-contre, qui prend 2 booléens et renvoie un booléen. Déterminer les valeurs des expressions suivantes.

```
def anonyme(a, b):
    if a:
        return b
    else:
        return True
```

```
>>> anonyme(False, False)
True
>>> anonyme(False, True)
True
```

```
>>> anonyme(True, False)
False
>>> anonyme(True, True)
True
```

EXERCICE 9 : On considère la fonction ci-contre. Déterminer toutes les exécutions qui sont correctes.

```
def affiche(n):
    for i in range(n):
        print(i)
```

```
>>> affiche(3)
0
1
2
3
```

```
>>> affiche(4)
0
1
2
3
```

```
>>> affiche(4)
1
2
3
4
```

```
>>> affiche(4)
4
4
4
4
```

```
>>> affiche(0)
0
```

```
>>> affiche(1)
0
```

```
>>> affiche(1)
1
```

```
>>> affiche(0)
>>> # rien
```

EXERCICE 10 : Pour chacune des questions suivantes, déterminer la bonne réponse.

1) Choisir une expression booléenne pour la variable S qui satisfait la table de vérité suivante.

a) **A ou (non B)**

b) (non A) ou B

c) (non A) ou (non B)

d) non (A ou B)

A	B	S
0	0	1
0	1	0
1	0	1
1	1	1

2) Soient P et Q deux formules logiques telles que P est vraie et Q est fausse. Quelle est la valeur de l'expression (P et Q) ou ((non P) ou Q)?

a) vraie

b) **fausse**

c) ni vraie, ni fausse

d) vraie et fausse en même temps

3) Quelles valeurs des booléens a , b et c permettent de rendre vraie l'expression suivante : ((non a) et b) et ((non b) ou c)

a) $a = 1$, $b = 1$ et $c = 1$

b) $a = 1$, $b = 0$ et $c = 0$

c) **$a = 0$, $b = 1$ et $c = 1$**

d) $a = 0$, $b = 0$ et $c = 0$

EXERCICE 11 : On considère la fonction ci-dessous.

```
def mystere(n):
```

```
    i = 0
```

```
    s = 0
```

```
    while s < n:
```

```
        i = i + 1
```

```
        s = s + i
```

```
    return s
```

i	s	s < n
0	0	True
1	1	True
2	3	True
3	6	True
4	10	True
5	15	True
6	21	False

1) Compléter le tableau ci-contre pour l'exécution de `mystere(20)` et entourer la valeur renvoyée à la fin. Chaque ligne, à part la première, correspond à l'état de la mémoire après l'instruction `s = s + i`

2) Quelle est la valeur renvoyée par `mystere(100)` ?

Solution : On remarque que `s` est la somme des entiers de 1 à `i`. Lorsque `i = 14`, on obtient `s = 105` qui est donc la valeur obtenue.

EXERCICE 12 : On veut écrire une fonction `remplace` qui prend une chaîne de caractères `texte` et deux symboles `s1` et `s2`, et qui renvoie une copie de `texte` où `s1` a été remplacé par `s2`.

```
>>> replace('bonjour', 'o', 'X')  
'bXnjXur'
```

```
>>> replace('bonjour', 'M', 'i')  
'bonjour'
```

1) Quel est le résultat de `replace("radiateur", "a", "o")` ?

Solution : On obtient "rodiateur".

2) Quel est le résultat de `replace("radiateur", "o", "a")` ?

Solution : On obtient "radiateur", car il n'y a pas de "o".

3) Quel est le résultat de `replace("", "a", "b")` ?

Solution : On obtient "", c'est-à-dire le texte vide.

4) Compléter le code de la fonction ci-dessous :

```
def replace(texte, s1, s2):
```

```
    res = ""
```

```
    for lettre in texte:
```

```
        if lettre == s1:
```

```
            res = res + s2
```

```
        else:
```

```
            res = res + lettre
```

```
    return res
```