

Séance 1 : VARIABLES, FONCTIONS ET CONDITIONS

L1 – Université Côte d’Azur

Exercice 1 — Variables et instructions

Exécutez manuellement le code ci-dessous et donnez, pour chacune des lignes exécutées, la valeur des variables en mémoire. On commencera à partir de la ligne 5.

```

1 def double(x):
2     a = 10*x
3     return a//5
4
5 a = 2
6 b = 10
7 if a>b:
8     c = a/b
9 else:
10    c = b/a
11 a = double(a)+1
12 a == c
13 d = b % (a - c)

```

Exercice 2 — Fonctions simples

1. Écrivez la fonction `max(a, b)` qui renvoie le plus grand élément entre a et b. Écrivez trois tests avec `assert`
2. Écrivez la fonction `max4(a, b, c, d)` qui renvoie le plus grand élément entre les quatres arguments en utilisant la fonction `max`.
3. Même question que la précédente mais sans utiliser la fonction `max`.
4. On définit le signe d'un nombre comme étant positif si ce nombre est supérieur ou égal à zéro et négatif s'il est strictement inférieur à 0. Écrivez une fonction `même_signe(a, b)` qui renvoie un booléen indiquant si les deux nombres a et b ont le même signe.
5. Écrivez cinq tests avec `assert` pour la fonction `même_signe`

Exercice 3 — Courbes de fonctions

Tracez la courbe des fonctions ci-dessous.

```

1 def f(x) :
2     if x <= 0 or x >= 1 :
3         return 1
4     else :
5         return -1
6
7

```

```

1 def g(x) :
2     if x <= 0 :
3         return 1
4     elif x <= 1 :
5         return 0
6     else :
7         return 1

```

```

1 def h(x) :
2     if x < 0 :
3         return -x
4     return x

```

```

1 def i(x) :
2     return x
3     if x < 0 :
4         return -x

```

Exercice 4 — Le plus grand en valeur absolue

On souhaite écrire une fonction `max_abs(x,y)` qui **renvoie** le nombre le plus grand en valeur absolue. Par exemple on aura `max_abs(2,-3) == -3`. Si les deux nombres ont la même valeur absolue mais pas le même signe, la fonction renverra celui qui est positif. Par exemple, `max_abs(3,-3) == 3`.

1. Écrivez trois tests avec `assert`.

```
1 assert max_abs(3,-5)==-5
2 assert max_abs(-3,5)==5
3 assert max_abs(-4,4)==4
```

2. Écrivez le fonction `max_abs(x,y)`

```
1 def max_abs(x,y) :
2     if abs(x) > abs(y) :
3         return x
4     elif abs(x) == abs(y) :
5         if x > 0 :
6             return x
7         else : # abs(x)==abs(y) et x<=y
8             return y
9     else : # abs(x)<abs(y)
10        return y
```

Autre solution :

```
1 def max_abs2(x,y) :
2     if abs(x) > abs(y) :
3         return x
4     elif abs(x) == abs(y) and x > 0 :
5         return x
6     else :
7         return y
```

3. Modifiez la fonction précédente en une fonction `afficher_max_abs(x,y,msg)` sans résultat qui **affiche** le message `msg` suivi du plus grand en valeur absolue. Par exemple, dans la console, on aura :

```
1 >>> afficher_max_abs(1,-3,'le plus grand en valeur absolue est')
2 le plus grand en valeur absolue est -3
```

```
1 def afficher_max_abs(x,y,msg) :
2     if abs(x) > abs(y) or (abs(x) == abs(y) and x > 0) :
3         print(msg , x)
4     else :
5         print(msg , y)
```

4. Écrivez la fonction `afficher_max_abs` en une seule ligne en appelant la fonction `max_abs`.

```
1 def afficher_max_abs(x,y,msg) :
2     print(msg, max_abs(x,y))
```

Exercice 5 — Convertir l'heure

1. Écrivez une fonction `hms(n)` prenant un entier positif n représentant un nombre de secondes. L'effet de cette fonction sans résultat est l'affichage d'une ligne exprimant la conversion de n secondes en heures-minutes-secondes.

```
1 >>> hms(4567)
2 4567 ---> 1 heure(s) 16 minute(s) 7 seconde(s)
```

```

1 def hms(n) :
2     secondes = n%60
3     minutes_dont_heures = n//60
4     minutes = minutes_dont_heures%60
5     heures = minutes_dont_heures//60
6     print(n, '---->', heures, 'heure(s)', minutes, 'minute(s)', secondes, 'seconde(s)')

```

2. Modifier le programme pour gérer le « s » du pluriel.

```

1 >>> hms(4567)
2 4567 ---> 1 heure 16 minutes 7 secondes
3 >>> hms(4140)
4 4140 ---> 1 heure 9 minutes 0 seconde

```

```

1 # L'astuce est d'ajouter une fonction auxiliaire.
2 def pluriel(mot,n):
3     if n<=1:
4         return mot
5     else:
6         return mot+"s"
7
8 def hms(n):
9     secondes = n%60
10    minutes_dont_heures = n//60
11    minutes = minutes_dont_heures%60
12    heures = minutes_dont_heures//60
13
14    h = pluriel("heure", heures)
15    m = pluriel("minute", minutes)
16    s = pluriel("seconde", secondes)
17    print(n, '---->', heures, h, minutes, m, secondes, s)

```

Exercice 6 — L'espion

On définit la fonction suivante :

```

1 def spy() :
2     print('My name is')
3     # Bond, James Bond
4     return 0 + 0 + 7

```

Qu'affiche la console si l'on saisit les expressions suivantes ? Expliquez.

- 1) spy()
- 2) spy
- 3) spy() + spy()
- 4) max(spy() , spy())
- 5) spy() == 7 or spy() == 'My name is'
- 6) print(spy())
- 7) print(print(spy()))

La première ligne est affichée, la seconde est retournée (c'est le résultat)

```
1 >>> spy()  
2 My name is  
3 7
```

Une fonction non évaluée a une valeur : son nom et son adresse en mémoire

```
1 >>> spy  
2 <function spy at 0x7fa27355d8a0>
```

La fonction est appelée deux fois, elle affiche deux fois « My name is », puis le calcul renvoie 14.

```
1 >>> spy() + spy()  
2 My name is  
3 My name is  
4 14
```

La fonction est appelée deux fois, elle affiche deux fois « My name is », puis le calcul renvoie 7.

```
1 >>> max(spy() , spy())  
2 My name is  
3 My name is  
4 7
```

La fonction n'est appelée qu'une fois (évaluation paresseuse Cours 1 partie V). Le calcul n'affiche qu'une fois « My name is » puis renvoie *True*.

```
1 >>> spy() == 7 or spy() == 'My name is'  
2 My name is  
3 True
```

En apparence comme au 6.1, mais c'est le *print* qui déclenche l'affichage du 7, alors qu'à la question 1 c'était le shell qui avait affiché le résultat de l'expression. Rien n'est renvoyé (techniquement si : *None* est renvoyé, mais dans ce cas le shell n'affiche rien).

```
1 >>> print(spy())  
2 My name is  
3 7
```

La fonction *print* renvoie *None*; le shell n'affiche pas le resultat, mais *print(None)* affiche *None*

```
1 >>> print(print(spy()))  
2 My name is  
3 7  
4 None
```