

# Info113 - TD2

L'objectif du second TD est de se familiariser avec la notion de structure conditionnelle, tout en réutilisant les notions de variables et de fonctions abordées à la fin du TD précédent. Ceci nous permettra d'écrire des programmes *Python* ayant une exécution non linéaire. Les concepts vus dans ce TD sont :

- les variables globales / locales,
- les fonctions,
- les expressions booléennes,
- les structures conditionnelles (`if ... then ...`).

## RETOUR SUR LE TD PRÉCÉDENT

**Exercice 1.** Commencez par définir deux variables `a` et `b` contenant des valeurs arbitraires. Trouvez ensuite un moyen d'invertir leurs valeurs, c'est à dire d'écrire la valeur initiale de `a` dans `b`, et la valeur initiale de `b` dans `a`.

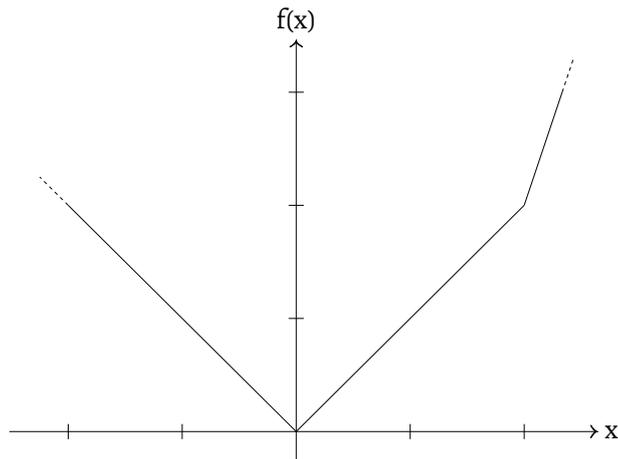
**Exercice 2.** On définit dans l'interpréteur les variables suivantes :

```
x = 6                a = len(z) - 10
y = x * 10           b = "ab" * 10
z = "Python"         c = 2 ** 10
```

Essayez de deviner la valeur des expressions suivantes, puis vérifiez votre résultat en les entrant dans l'interpréteur. On supposera que les fonctions `abs`, `moyenne2` et `moyenne3` définies lors du TD précédent ont été entrées dans l'interpréteur au préalable.

```
moyenne3(3, abs(2 ** 4 - 12), moyenne2(1, len(z) // 2))
(3 ** 2 + 4 ** 2) ** 0.5
len(z * 10) // 5
x * 2 / (y / abs(a))
c // 2 + c // 4 + c // 8
(z + b * 2)[0] + "i"
((y + 2 * c) / 4) // 2
abs(abs(abs(a) - 2) - 10) + moyenne2(abs(-1 * c), c // 4)
```

**Exercice 3.** On s'intéresse à la fonction  $f$  définie par  $f(x) = -x$  si  $x \leq 0$ ,  $f(x) = x$  si  $0 < x < 2$ ,  $f(x) = 3x - 4$  si  $2 \leq x$ . On peut également l'exprimer à l'aide de la valeur absolue :  $f(x) = |x| + |x - 2| + x - 2$ . Définissez la fonction *Python*  $f$  correspondante, après avoir défini la valeur absolue de la même manière qu'au TD précédent.



## MANIPULER DES EXPRESSIONS BOOLÉENNES

**Exercice 4.** Définissez une fonction `est_multiple_de` qui prend en argument deux entiers  $m$  et  $n$ , et qui retourne `True` si  $m$  est un multiple de  $n$ , et `False` sinon.

*Note :* en *Python*, l'opérateur `%` désigne le modulo (le reste de la division euclidienne).

**Exercice 5.** Définissez une variable  $x$  contenant un entier (une année par exemple), puis en utilisant la fonction `est_multiple_de` de l'exercice précédent, écrivez des expressions booléennes qui :

1. vérifie si  $x$  est un multiple de 4,
2. vérifie si  $x$  est un multiple de 4, mais pas un multiple de 100,
3. vérifie si  $x$  est un multiple de 4, mais pas un multiple de 100, sauf si c'est un multiple de 400.

**Exercice 6.** Décrivez en français le test effectué par l'expression suivante sur la variable  $x$  :

```
(x < 100 and x % 10 == 0) or (
    x >= 250 and x % 5 == 0 and (x % 3 == 0 or x % 2 == 0))
```

Que vaut cette expression si  $x$  est égal à 24, 255, 90, -18, 1242 et 820 ?

**Exercice 7.** Traduisez en une expression *Python* la phrase suivante: « La valeur de  $x$  est soit inférieure à zéro, soit paire et strictement plus grande que 42, soit un multiple de 3 qui n'est ni un multiple de 5, ni un multiple de 2. »

**Exercice 8.** Pour chaque combinaison de valeurs booléennes possibles des variables  $a$  et  $b$ , donnez la valeur des expressions suivantes:

```
not (a or b)                (not a) or (not b)
(not a) and (not b)        not (a and b)
```

Pouvez vous en déduire deux équivalences sur les expressions booléennes ?

**Exercice 9.** Définissez une fonction `est_ok` qui prend en argument une chaîne de caractères, et retourne `True` si la chaîne contient le mot « OK », dont chaque caractère peut être soit une majuscule, soit une minuscule. Recommencez avec le mot « Python », puis avec le mot « Llanfairpwllgwyngyllgogerychwyrndrobwlllantysiliogogoch ».

## VARIABLES LOCALES, VARIABLES GLOBALES

**Exercice 10.** Définissez une variable globale `pi`, contenant une valeur approchée de la constante  $\pi \simeq 3.141592$ , puis utilisez-la pour définir deux fonctions `perimetre_disque` et `surface_disque` avec un argument contenant le rayon. Pourquoi est-il préférable de définir une variable globale dans ce cas ?

**Exercice 11.** On entre dans l'interpréteur les lignes suivantes:

```
x = 12
def f(x):
    return x + 2
def g(y):
    y = y + 7
    return x + y
def h(z):
    y = 2
    x = y + (z + 1) ** 2
    z = -2
    return (x + y) * z
```

Quelles sont les valeurs des expressions suivantes ?

```
f(-14)                g(-4)
f(x)                  h(x)
g(x - 7)              h(0)
```

Quelle est la valeur de la variable  $x$  avant et après l'exécution de chacune de ces expressions ?

## LES STRUCTURES CONDITIONNELLES « IF ... ELSE ... »

**Exercice 12.** Réécrivez la fonction de valeur absolue `abs` en utilisant une structure conditionnelle. Laquelle des deux versions est-elle préférable ?

**Exercice 13.** Utilisez la première définition de la fonction `f` donnée dans l'exercice 3 pour redéfinir la fonction `f` en utilisant une structure conditionnelle.

**Exercice 14.** Définissez une fonction `mention` qui prend en argument une note entre 0 et 20, et retourne une chaîne de caractères contenant la mention correspondante (passable, assez bien, bien, très bien).

**Exercice 15.** Définissez une fonction `est_bissextile`, qui prend en argument une année, et retourne un booléen indiquant `True` si l'année est bissextile, et `False` sinon. Vous pourrez vous inspirer de vos réponses aux exercices 4 et 5.

*Note :* une année est bissextile si elle est un multiple de 4, mais pas un multiple de 100, ou si elle est un multiple de 400.

**Exercice 16.** Définissez ensuite une fonction `nb_jours` qui prend une année en argument, et retourne un entier contenant le nombre de jours dans cette année.

*Note :* une année bissextile comporte 366 jours.

**Exercice 17.** Définissez une fonction `est_bissextile2`, qui prend en compte le calendrier julien, c'est à dire qu'avant 1582, seules les années multiples de 4 sont bissextiles.

**Exercice 18.** Définissez une fonction `max2`, puis des fonctions `max3`, `max4` et `max5`, qui prennent en argument deux, trois, quatre ou cinq nombres, et retournent le maximum.

**Exercice 19.** Utilisez des structures conditionnelles pour résoudre à nouveau l'exercice 9.