

STRUCTURES CONDITIONNELLES

L'instruction `if` permet de vérifier une condition avant d'exécuter un bloc d'instructions.

```
if <condition> :  
    <instruction 1>  
    <instruction 2>  
    ...  
    <instruction n>  
<suite> #ne fait pas partie du bloc
```

L'expression `condition` renvoie un booléen, et les instructions 1,...,n, ne sont exécutées que si ce booléen vaut `True`.

La suite du programme, non indentée, sera exécutée quelle que soit l'évaluation de la condition. C'est l'indentation qui permet de définir l'appartenance des instructions à la structure conditionnelle.

On peut par ailleurs préciser un bloc d'instructions à exécuter lorsque la condition renvoie `False` :

```
if <condition> :  
    <instruction 1>  
    <instruction 2>  
    ...  
    <instruction n>  
else :  
    <instruction 1bis>  
    ...  
<suite>
```

Comme précédemment, l'indentation des instructions définit le bloc auquel elles appartiennent.

Enfin, lorsqu'il faut distinguer plus de deux cas, plutôt que d'écrire :

```
if <condition1> :  
    <bloc d'instructions 1>  
else :  
    if <condition2> :  
        <bloc d'instructions 2>  
    else :  
        if <condition3> :  
            <bloc d'instructions 3>  
        else :  
            <bloc d'instructions 4>  
<suite>
```

on pourra écrire :

```
if <condition1> :  
    <bloc d'instructions 1>  
elif <condition2> :  
    <bloc d'instructions 2>  
elif <condition3> :  
    <bloc d'instructions 3>  
else :  
    <bloc d'instructions 4>  
<suite>
```

Exercices

EXERCICE 1

Quelles sont les valeurs des variables x et y après les instructions suivantes :

```

if x == 5:
    x = x + 1
    if y == 6 :
        x = 10
    y = 2*y
else:
    x = y - 1
y = y + 1

```

1. dans le cas où leurs valeurs initiales sont $x = 5$ et $y = 6$?
2. dans le cas où leurs valeurs initiales sont $x = 2$ et $y = 6$?
3. dans le cas où leurs valeurs initiales sont $x = 5$ et $y = 4$?
4. dans le cas où leurs valeurs initiales sont $x = 8$ et $y = 1$?

EXERCICE 2

1. Écrire un programme qui demande à l'utilisateur trois nombres entiers a , b , c et qui affiche le triplet contenant ces trois nombres dans l'ordre croissant ; on n'utilisera pas les fonctions `min` et `max`.
2. Écrire un programme qui demande à l'utilisateur trois nombres entiers a , b , c et qui indique si un triangle de côtés de longueurs respectives a , b et c est ou non rectangle.
3. (*) Écrire un programme qui demande à l'utilisateur trois nombres entiers a , b , c et qui indique la nature du triangle de côtés de longueurs respectives a , b et c .

EXERCICE 3

Le mois de février comporte 28 jours, sauf les années bissextiles où il en comporte 29.

Une année est bissextile s'il s'agit d'un multiple de 4 qui n'est pas un multiple de 100 ; de plus, les multiples de 400 sont aussi bissextiles (bien que multiples de 100).

1. Écrire un programme qui demande à l'utilisateur une année a et qui indique si l'année est ou non bissextile.
2. (*) Écrire un programme qui demande à l'utilisateur trois nombres entiers j , m , a et qui indique si le triplet (j, m, a) représente une date valide.
3. (**) On convient de numéroté les jours de la semaine de 0 à 6 (0 pour dimanche, 1 pour lundi, etc.).

En notant $\lfloor x \rfloor$ la partie entière de x , le numéro du jour de la semaine correspondant à une date donnée (j, m, a) s'obtient en prenant le reste de la division par 7 de l'entier n défini par :

$$n = j + \lfloor 2,6m - 0,2 \rfloor + d + \left\lfloor \frac{d}{4} \right\rfloor + \left\lfloor \frac{c}{4} \right\rfloor + 5c$$

où :

$$p = \begin{cases} m + 10 & \text{si } m \leq 2 \\ m - 2 & \text{sinon} \end{cases} \quad b = \begin{cases} a - 1 & \text{si } m \leq 2 \\ a & \text{sinon} \end{cases} \quad c = \left\lfloor \frac{b}{100} \right\rfloor \quad d = b - 100c$$

(Formule de Zeller)

Écrire un programme qui demande à l'utilisateur trois nombres entiers j , m , a et qui, si le triplet (j, m, a) représente une date valide, indique le jour de la semaine correspondant à cette date.