1 La structure conditionnelle

En algorithmique, il est assez fréquent d'avoir des instructions qui ne sont évaluées que sous certaines conditions (par exemple une racine carrée sera calculée que si on a un nombre positif). On utilisera en Python la structure suivante :

```
if (conditions) :
    instructions
```

L'indentation est primordiale. En effet, c'est grâce à celle-ci que Python délimite les structures.

Tester par exemple le script suivant. À l'exécution, on essaiera le cas où on donne un nombre positif et le cas où on donne un nombre négatif.

```
n=eval(input('Donnez un nombre '))
if (n>0) :
    print('Ce nombre est positif')
print('Cette phrase apparaît toujours')
```

Quand Python teste `(n>0)`, il récupère soit `True` si la condition est vérifiée, soit `False` si elle ne l'est pas.

Pour créer une condition, on utilise en général les opérateurs suivants :

Syntaxe Python	$x < y$	$x > y$	$x == y$	$x! = y$	$x <= y$	$x >= y$
Signification	$x < y$	$x > y$	$x = y$	$x \neq y$	$x \leq y$	$x \geq y$

Attention : il faut bien distinguer le symbole d'affectation `=` du symbole de comparaison `==`.

On peut associer plusieurs conditions à l'aide de `or` (ou), `and` (et) ,`not` (non).

Dans les exemples suivants, prévoir si Python va répondre `True` ou `False` puis tester pour vérifier.

```
>>> ( 4 > 0)
>>> (3 <= 8) and (3 > 0)
>>> (10 < 0) or (10 == 5)
>>> (10 < 0) or (10 != 5)
>>> not((8**2==64) and (8>=2))
>>> not((8**2==64) and (8<2))
```

2 Disjonctions de cas

En algorithmique, il est assez fréquent que l'on ait à enchaîner des disjonctions de cas. Par exemple, l'existence de solutions réelles de l'équation $x^2 = a$ dépend du signe de a. Ainsi, on utilisera en Python la structure suivante :

```
if (conditions) :
    instructions
else :
    instructions
```

Que fait le script suivant ?

```
n=eval(input('Donnez un nombre '))
if (n>0) :
    print('Ce nombre est positif')
else:
print('Ce nombre est négatif')
```

Dans les cas où il y a plus de deux choix possibles, on peut utiliser une version plus élaborée :

```
if (conditions) :
    instructions
elif (conditions) :
    instructions
```

```
        ⋮
elif (conditions) :
    instructions
else :
    instructions
```

3 Exercices

Exercice 1

Écrire un script qui demande à l'utilisateur d'entrer un nombre et affiche si le nombre est supérieur à 100.

Exercice 2

Écrire un script qui demande à l'utilisateur d'entrer un nombre et compare le nombre est supérieur à 100.

Exercice 3

Créer un script `entier` qui demande à l'utilisateur de donner un nombre réel et affiche la phrase `Ce nombre est un entier` si le nombre est entier et la phrase `Ce nombre n'est pas un entier` sinon.

conseil : penser à utiliser l'instruction `floor`

Exercice 4

Créer un script `devinette` qui génère un nombre `a` au hasard entre 0 et 10 puis demande à l'utilisateur de donner un nombre entier `b` entre 0 et 10. Si `b` et `a` sont égaux, on affiche `Vous avez gagné`, sinon on affiche `Vous avez perdu`.

Exercice 5

Créer un script qui demande à l'utilisateur de saisir un nombre entier naturel non nul et indique si ce nombre est divisible par 2, divisible par 3, divisible par 5, divisible par 10.

Bien sûr un nombre peut être divisible par plusieurs entiers...