

## Instructions conditionnelles

### I. Les booléens

◇ Le type booléen (*bool*) comporte deux valeurs : **True** et **False** qui représentent *le résultat d'un test*. Les booléens sont utilisés, au cours d'un programme, pour prendre des décisions et agir sur le déroulement du programme.

◇ On peut obtenir des booléen avec les relations de comparaison  $<$ ,  $>$ ,  $<=$ ,  $>=$ ,  $==$  (égal) et  $!=$  (différent).

⚠ Pour tester l'égalité de  $x$  et  $y + 1$ , il faut bien utiliser  $==$  et surtout pas  $=$ , en effet écrire  $x=y+1$  ne teste pas si  $x$  et  $y + 1$  sont égaux mais affecte à la variable  $x$  la valeur  $y + 1$ .

◇ On peut aussi utiliser les opérations logiques **and**, **or**, **not**.

*Remarque.* On peut définir des variables de type booléen (comme pour n'importe quel autre type) mais on le fait rarement en pratique. □

```
>>> x=1
>>> y=2
>>> x<y
True
>>> y<=x
False
>>> y==x+1
True
>>> y!=x+1
False
>>> x<y<=x+1
True
>>> (x<2) and (not (y==3))
True
>>> not (x==1)
False
```

### II. Instruction conditionnelles

◇ Une *instruction conditionnelle* sépare le déroulement du programme en deux suivant une certaine condition. Une instruction conditionnelle est composée de 3 éléments :

- Une *condition*  $c$  ;
- Une *première liste d'instructions*, exécutées si  $c$  est vraie ;
- Une *deuxième liste d'instructions*, exécutées si  $c$  est fausse.

◇ Dans un algorithme, on écrit :

---

```
si c
alors
    Liste d'instructions 1
sinon
    Liste d'instructions 2
fin si
```

---

*Exemple Python.*

```
x=1
if x<0:
    print "Négatif"
else:
    print "Positif"
```

Positif

□

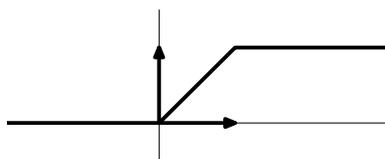
◇ Avec PYTHON :

- La partie **else** : peut être omise ;
- Il y a également une construction **if/elif/else** (voir exemple suivant) ;

△ Les différentes listes d'instructions sont repérées par leur indentation.

*Exemple Python.* Définir la fonction  $f$  telle que :

$$\begin{aligned} \forall x \in \mathbb{R}, f(x) &= 0 \text{ si } x \leq 0 \\ &= x \text{ si } 0 < x < 1 \\ &= 1 \text{ sinon} \end{aligned}$$



```
def f(x):
    if x<=0:
        return 0
    elif 0<x<1:
        return x
    else:
        return 1
print "f(-1)=", f(-1)
```

f(-1)= 0

```
print "f(0.5)=", f(0.5)
```

f(0.5)= 0.5

*Exemple Python.* Écrire une fonction qui calcule et affiche les  $n$  premiers termes de la suite de Syracuse ( $u_n$ ) définie par  $u_0 \in \mathbb{N}$  et :

$$\begin{aligned} \forall n \in \mathbb{N}, u_{n+1} &= \frac{u_n}{2} \text{ si } u_n \text{ est pair} \\ &= 3u_n + 1 \text{ si } u_n \text{ est impair} \end{aligned} \quad \square$$

```
def syracuse(u0, n):
    u=u0
    for k in range(n):
        print u
        if u%2==0:
            u=u/2
        else:
            u=3*u+1
```

Affichage des 60 premiers termes de la suite lorsque  $u_0 = 37$  :

```
syracuse(37, 60)
```

```
37 112 56 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1 4 2 1 4 2 1 4
2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1 4 2
```