Chapitre 3

programmation Python

Tests et conditions

« Ceci n'est pas un test ! »

I Des outils pour comparer

Ce sont les *opérateurs de comparaison* :

Opérateur	Signification	Remarques
<	strictement inférieur	Ordre usuel sur `int` et `float`, lexicographique sur `str`...
<=	inférieur ou égal	Idem
>	strictement supérieur	Idem
>=	supérieur ou égal	Idem
==	égal	« avoir même valeur » *Attention :* deux signes =
=!	différent	
`is`	identique	avoir même valeur *et* être le même objet
`is not`	non identique	
`in`	appartient à	avec `str`, `list` et `dict`
`not in`	n'appartient pas à	avec `str` et `list` et `dict`

Code Python (shell)

```
>>> a = 2
>>> a == 2
>>> a == 3
>>> a == 2.0
>>> a is 2.0
>>> a  != 100
>>> a > 2
>>> a >= 2
```

Code Python (shell)

```
>>> a = 'Alice'
>>> b = 'Bob'
>>> a < b
```

```
>>> 'e' in a
>>> 'e' in b
>>> liste = [1,10,100]
>>> 2 in liste
```

Ces opérateurs permettent de réaliser des tests basiques. Pour des tests plus évolués on utilisera des « mots de liaison » logiques.

II Les connecteurs logiques

- **and** permet de vérifier que 2 conditions sont *vérifiées simultanément*.
- **or** permet de vérifier qu'*au moins une* des deux conditions est vérifiée.
- **not** est un opérateur de *négation* très utile quand on veut par exemple vérifier qu'une condition est fausse.

Voici les tables de vérité des deux premiers connecteurs :

and	*True*	*False*
True	**True**	**False**
False	**False**	**False**

or	*True*	*False*
True	**True**	**True**
False	**True**	**False**

À ceci on peut ajouter que **not True** vaut **False** et vice-versa.

Code Python (shell)

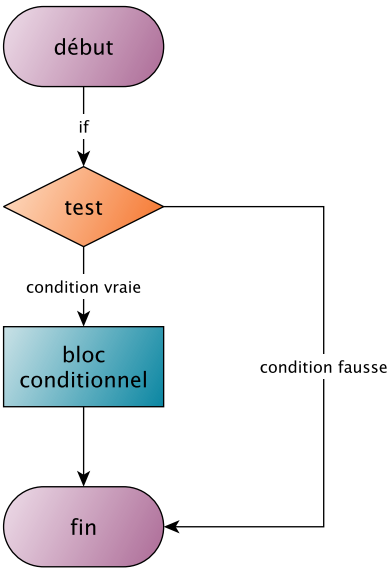
```
>>> True and False
>>> True or False
>>> not True
```

Code Python (shell)

```
>>> resultats = 12.8
>>> mention_bien = resultats >= 14 and resultats < 16
>>> print(mention_bien)
```

III if, else et elif

Voici le schéma de fonctionnement d'un test **if** :

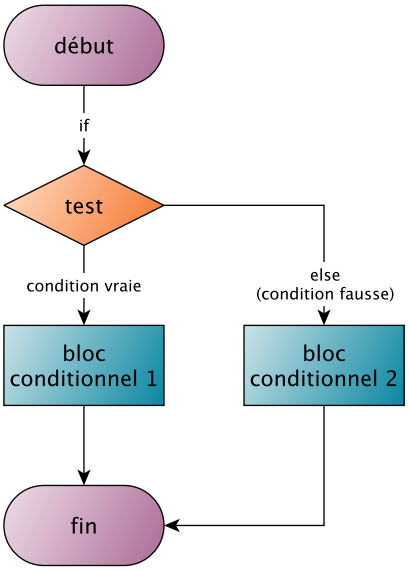


Attention : Un bloc conditionnel doit être *tabulé* par rapport à la ligne précédente : il n'y a ni `DébutSi` ni `FinSi` en PYTHON, ce sont les tabulations qui délimitent les blocs.

Code Python

```
phrase='Je vous trouve très joli'
reponse = input('Etes vous une femme ?(O/N) : ')
if reponse == 'O' :
    phrase += 'e'
phrase +='.'
print(phrase)
```

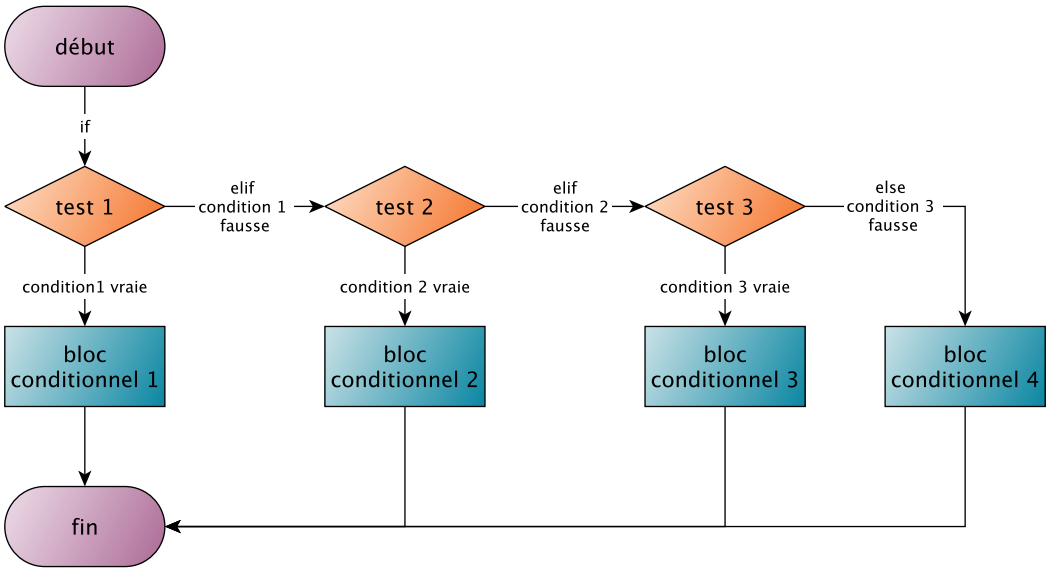
Voici le schéma de fonctionnement d'un test **if...else** :



Code Python

```
print('Bonjour')
age = int(input('Entrez votre age  : '))
if age >= 18 :
    print('Vous etes majeur')
else :
    print('Vous etes mineur.')
print('Au revoir.')
```

Voici le schéma de fonctionnement d'un test **`if...elif...`** :



Code Python

```
print('Bonjour')
prenom = input('Entrez un prénom  : ')
if prenom == 'Robert' :
    print("Robert, c'est le prénom de mon grand-père.")
elif prenom == 'Raoul' :
    print("Mon oncle s'appelle Raoul.")
elif prenom == 'Médor' :
    print("Médor, comme mon chien !")
else :
    print("Connais pas")
print('Au revoir.')
```

On peut bien sûr inclure autant de **elif** que nécessaire.

Exercices

Exercice 1

Écrire un script qui demande son âge à l'utilisateur puis qui affiche `'Bravo pour votre longévité.'` si celui-ci est supérieur à 90.

Exercice 2

Écrire un script qui demande un nombre à l'utilisateur puis affiche si ce nombre est pair ou impair.

Exercice 3

Écrire un script qui demande l'âge d'un enfant à l'utilisateur puis qui l'informe ensuite de sa catégorie :

- trop petit avant 6 ans ;
- poussin de 6 à 7 ans inclus ;
- pupille de 8 à 9 ans inclus ;
- minime de 10 à 11 ans inclus ;
- cadet à 12 ans et plus ;

Exercice 4

Écrire un script qui demande une note sur 20 à l'utilisateur puis vérifie qu'elle est bien comprise entre 0 et 20. Si c'est le cas rien ne se produit mais sinon le programme devra afficher un message tel que `'Note non valide.'`.

Exercice 5

Écrire un script qui demande un nombre à l'utilisateur puis affiche s'il est divisible par 5, par 7 par aucun ou par les deux de ces deux nombres.

Exercice 6

En reprenant l'exercice du chapitre 1 sur les numéros de sécurité sociale, écrire un script qui demande à un utilisateur son numéro de sécurité sociale, puis qui vérifie si la clé est valide ou non.

Exercice 7

Écrire un script qui résout dans **R** l'équation du second degré $ax^2 + bx + c = 0$.
On commencera par `from math import sqrt` pour utiliser la fonction `sqrt`, qui calcule la racine carrée d'un `float`.
On rappelle que lorsqu'on considère une équation du type $ax^2 + bx + c = 0$

- si $a = 0$ ce n'est pas une équation de seconde degré ;
- sinon on calcule $\Delta = b^2 - 4ac$ et
  - Si $\Delta < 0$ l'équation n'a pas de solutions dans **R** ;
  - Si $\Delta = 0$ l'équation admet pour unique solution $\frac{-b}{2a}$;
  - Si $\Delta > 0$ l'équation admet 2 solutions : $\frac{-b-\sqrt{\Delta}}{2a}$ et $\frac{-b+\sqrt{\Delta}}{2a}$.

Pour vérifier que le script fonctionne bien on pourra tester les équations suivantes :

- $2x^2 + x + 7 = 0$ (pas de solution dans **R**) ;
- $9x^2 - 6x + 1 = 0$ (une seule solution qui est $\frac{1}{3}$) ;
- $x^2 - 3x + 2$ (deux solutions qui sont 1 et 2).

Exercice 8

L'opérateur nand est défini de la manière suivante : si A et B sont deux booléens alors

`A nand B vaut not (A and B)`

Construire la table de vérité de nand en complétant :

A	B	A and B	not (A and B)
False	False		
False	True		
True	False		
True	True		

Exercice 9

Faire le QCM 02 de M. Sincère.

Corrigés des exercices

Corrigé de l'exercice 1

Code Python

```
numsecu = input("Entrer le numéro de sécurité sociale
    : ")
# numsecu est donc une variable de type str
if len(numsecu) != 15 : # si la longueur du str n'est
   pas 15 ce n'est pas un numéro valide
print("Le numéro n'a pas 15 chiffres !")
else : # sinon
numsecu = int(numsecu) # on convertit le str en int
cle = numsecu % 100 # la clé est le reste de la
   division euclidienne de numsécu par 100 : les 2
   derniers chiffres
reste = numsecu // 100   # division entière par 100 =>
    13 premiers chiffres
if cle == 97 - reste % 97 :  # si la clé est bonne
print("Le numéro est bon.") # tant mieux
else :  # sinon tant pis
print("Le numéro n'est pas bon.")
```

Corrigé de l'exercice 2

Code Python

```
age = int(input("Entrez votre age  : "))
if age < 6 :
print("Vous etes trop petit.")
elif 6 <= age <= 7 :
print("Vous jouez en Poussin.")
elif age <= 9 :
print("Vous jouez en Pupille.")
elif age <= 11 :
print("Vous jouez en Minime.")
else :
print("Vous jouez en Cadet.")
```

Corrigé de l'exercice 3

Code Python

```
from math import sqrt

a = float(input('Entrez a  : '))
b = float(input('Entrez b  : '))
c = float(input('Entrez c  : '))
if a == 0 :
print("a = 0, c'est une équation du 1er degré !")
else :
delta = b**2 - 4*a*c
print('Le discriminant vaut  : ', delta)
if delta < 0 :
print("Il n'y a pas de solutions réelles")
elif delta == 0 :
s1 = -b/(2*a)
print("Il y a une solution réelle  : s1 = ", s1)
else :
s1 = (-b-sqrt(delta))/(2*a)
s2 = (-b+sqrt(delta))/(2*a)
print("Il y a deux solutions réelles  : s1 = ", s1, "
   et s2 = ", s2)
```

Corrigé de l'exercice 4

On s'y prend progressivement :

A	B	A and B	not (A and B)
False	False	False	True
False	True	False	True
True	False	False	True
True	True	True	False

On peut si on le désire conclure avec la table suivante :

nand	*True*	*False*
True	False	True
False	True	True

Corrigé de l'exercice 5

xor	*True*	*False*
True	`False`	`True`
False	`True`	`False`

Corrigé de l'exercice 6

Cela se fait comme à l'exercice précédent.

Corrigé de l'exercice 7

Il n'y a rien à corriger.