Instructions conditionnelles

Pour que les programmes soient intéressants, il est nécessaire qu'ils puissent effectuer des opérations différentes en fonction des données qu'ils reçoivent. Pour pouvoir programmer cela, le programmeur dispose d'une instruction : l'instruction if, ou instruction conditionnelle.

Pour écrire une instruction conditionnelle simple on rédige ainsi :

```
>>> if condition1 :
>>>     première ligne du bloc d'instructions 1
>>>     … …
>>>     dernière ligne du bloc d'instructions 1
>>> suite du programme
```

Dans la première ligne, le if permet au compilateur d'identifier le début d'une instruction conditionnelle. Ensuite il évalue la condition1. Si cette condition a la valeur vraie, le compilateur effectue le bloc d'instructions 1. Sinon, il ignore le bloc d'instructions 1 et passe directement à la suite du programme.

Comment le compilateur sait-il que le bloc d'instruction débute et se termine ?

Dans beaucoup de langage, il existe un mot clé équivalent au if qui débute l'instruction (par exemple then) et une qui termine l'instruction (par exemple end). En Python, le début du bloc d'instructions est marqué par les deux points, puis tout ce qui est indenté de la même manière que la première ligne du bloc d'instruction 1 est considéré par le compilateur comme faisant partie de ce bloc.

Il faudra donc être particulièrement vigilant sur la présentation du programme !

Ex : Que font les programmes suivants ?

```
Prog1 :
>>> a=3
>>> if a<2 :
>>>     a=2
>>>     print(a)
```

```
Prog2 :
>>> a=1
>>> if a<2 :
>>>     a=2
>>>     print(a)
```

```
Prog3 :
>>> a=1
>>> if a<2 :
>>>     a=2
>>> print('a')
```

Pour écrire une instruction conditionnelle avec une alternative, on rédige ainsi :

```
>>> if condition1 :
>>>     première ligne du bloc d'instructions 1
>>>     … …
>>>     dernière ligne du bloc d'instructions 1
>>> else :
>>>     première ligne du bloc d’instructions 2
>>>     … …
>>>     dernière ligne du bloc d'instructions 2
>>> suite du programme
```

Dans ce cas, le compilateur évalue la condition1 puis effectue le bloc d'instructions 1 si la

condition1 est vraie et le bloc d'instruction 2 si la condition1 est fausse. Là encore, l'indentation des instructions permet au compilateur d'identifier le début et la fin des blocs.

Enfin, on peut cumuler plusieurs conditions à vérifier en utilisant elif (quand il y a plus de deux situations possibles) :

```
>>> if condition1 :
>>>     première ligne du bloc d'instructions 1
>>>     … …
>>>     dernière ligne du bloc d'instructions 1
>>> elif condition2 :
>>>     première ligne du bloc d’instructions 2
>>>     … …
>>>     dernière ligne du bloc d'instructions 2
>>> else :
>>>     première ligne du bloc d’instructions 3
>>>     … …
>>>     dernière ligne du bloc d'instructions 3
>>> suite du programme
```

On peut également imbriquer des instructions conditionnelles. Il faut dans ce cas être encore plus vigilant sur les indentations puisqu'il faut faire apparaître un niveau d'indentation pour chaque if :

```
>>> if condition1 :
>>>      if condition2 :
>>>           bloc d'instructions 1
>>>      else :
>>>           bloc d'instructions 2
>>> else :
>>>     bloc d'instructions 3
```

Attention : indentez en utilisant les espaces plutôt que les tabulations.

Exercices :

1- Écrire un script qui demande à l'utilisateur un nombre et renvoie sa valeur absolue.

2- Écrire un script qui demande à l'utilisateur de rentrer deux nombres puis qui renvoie le maximum des deux.

3- Écrire un script qui demande à l'utilisateur de rentrer trois nombres puis qui renvoie le maximum des trois.

4- Écrire un script qui demande à l'utilisateur de rentrer les trois coefficients d'un polynôme du second degré et qui renvoie :la ou les racines quand elles existent, ou un message d'erreur si le polynôme n'est pas du second degré ou s'il n'y a pas de racines réelles.

5- Programmer une clé de détermination des feuillus.