

### La structure if

Elle va permettre d'expliquer ce qu'il faut faire si une condition est vérifiée.

```
if note>15 :  
    print("Très Bien !")
```

Dans la première ligne on pourrait traduire le "if" par "si" et l'indispensable ":" en fin de ligne par "faire" ce qui donnerait "Si note>15 faire".

L'ordinateur comprend alors qu'il doit faire ce qui suit seulement si la note est supérieure à 15. Plus précisément, il ne va pas faire tout ce qui suit mais seulement ce qui est écrit en décalé (on dit que le texte est indenté). C'est pour cela que print n'est pas écrit au même niveau que if. Pour indenter, on utilise la touche tabulation. Pour résumer notre exemple, si la note obtenue est par exemple 19, le programme affichera Très Bien !. Si on obtient la note 15 il ne fera rien.

### La structure if avec else

Étoffons notre exemple en rajoutant ce qu'il doit afficher si la note n'est pas supérieure à 15.

```
if note>15 :  
    print("Très Bien !")  
else :  
    print("Insuffisant !")
```

Comme pour tout à l'heure, on peut traduire "else" par "sinon" et le ":" par "faire". L'ordinateur comprend alors que si la condition note>15 n'est pas vérifiée alors il exécutera ce qui est indenté en dessous de else.

Pour résumer notre exemple, si la note obtenue est par exemple 19, le programme affichera Très Bien !. Si on obtient la note 15 ou 3, il affichera Insuffisant !.

### La structure if... elif ... else

On comprend bien sur notre exemple qu'il faudrait rajouter davantage de conditions selon la valeur de la note pour afficher des commentaires plus pertinents. Pour cela, on va utiliser l'instruction intermédiaire elif qui est une contraction de else et if c'est à dire qu'on peut le traduire par "sinon si". Il va nous permettre d'enchaîner les distinctions de cas. On peut en mettre autant que l'on veut.

```
if note>16 :  
    print("Très Bien !")  
elif note>12 :  
    print("Bien")  
elif note>10 :  
    print("Moyen")  
elif note>20 :  
    print("Tu es un génie !!!")  
else :  
    print("Insuffisant !")
```

Remarque : Pour le premier elif, il n'est pas nécessaire de vérifier que la note est en dessous de 16 car si ce n'était pas le cas il aurait affiché "Très Bien !" et serait passé directement à la suite du else sans regarder les autres conditions. C'est pour cette même raison que "Tu es un génie" ne pourra jamais s'afficher car si la note est par exemple 22, on a note>16 et donc il s'affichera "Très Bien !" et c'est tout.

Pour résumer notre exemple : Quand le programme s'exécute et tombe sur un if il va chercher dans la série de conditions la première qui est validée et exécuter la partie du programme qui correspond puis va sauter directement après le bloc de conditions sans examiner les autres.

Concluons cette partie avec des erreurs classiques à éviter :

- Après if et elif il y a toujours une condition.
- Après else, il n'y a jamais de condition (étant donné que signifie sinon donc correspond à tous les cas restants, pas besoin donc de les préciser même si par chance on les connaît)
- Après un if, elif et else il ne faut pas oublier de mettre en fin de ligne les deux points :

### Les conditions

Après un if ou un elif, on a dit qu'il fallait mettre une condition. Une condition est tout simplement quelque chose qui peut être vrai ou faux. Voici quelques exemples de conditions qu'on complétera au besoin.

#### True et False

Ils signifient vrai et faux donc ce sont les conditions les plus évidentes. On les appelle les booléens. Il peut paraître absurde de mettre un booléen directement après un if (puisque si on met True, le code correspondant sera toujours exécuté donc le if ne sert à rien et inversement pour le False) mais dans la pratique, le booléen sera dans une variable qui elle pourra être modifiée et donc selon la valeur, l'action à effectuer ne sera pas la même.

#### Les comparaisons

- Une façon d'obtenir un booléen est en comparant deux objets comme on a fait avec notre note dans la partie précédente. Voici les notations pour pouvoir comparer :
- `==` : pour tester l'égalité. Il faut doubler le `=` pour le différencier de l'affectation des variables.
- `!=` : différent de.
- `<` : strictement inférieur.
- `<=` : inférieur ou égal.
- `>` : strictement supérieur.
- `>=` : supérieur ou égal.

#### Par exemple :

`1+1 == 2` est vrai.  
`2*3 == 3` est faux.  
`2+3 != 4` est vrai.  
`14 >= 15` est faux.

Les inégalités ne fonctionnent pas que pour les nombres. Par exemple on peut comparer des mots ou des listes de nombres avec l'ordre lexicographique (celui du dictionnaire).

`"elephant" < "souris"` est vrai  
`"français" < "mathématiques"` est vrai  
`(1,5) < (2,3)` est vrai

#### Les connecteurs logiques not, and et or

Ils permettent de créer une condition à partir d'autres conditions. Il en existe d'autres mais ce sont les principaux. Si on les traduit, ça donnerait "contraire de" pour not, "et" pour and et "ou" pour or. Par exemple :  
`not x < 3` est identique à `x >= 3`. `x < 3 and x > 1` est identique à `1 < x < 3` car x doit vérifier les deux conditions en même temps. `x < 3 or x > 5` est identique... à elle-même car il n'y a pas de façon plus concise de l'écrire. Elle sera vraie si une des deux conditions est vérifiée.