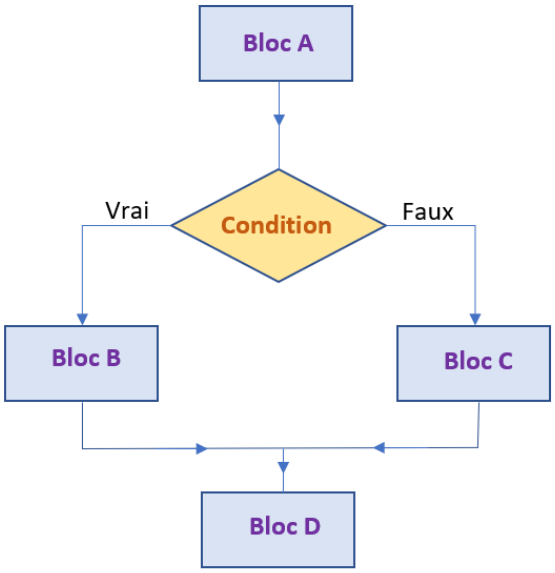
BAC Pro CIEL	Python – Module 2 Les conditions	Année 2025/2026

A. Les conditions

Un test permet de valider si une condition est vraie ou fausse. Elle permet d'adapter les instructions à effectuer en fonction du résultat. Cela permet au programme de prendre des décisions automatiquement.

La structure la plus simple consiste en trois étapes :

- **SI** une condition est vraie
- **ALORS** on effectue une série d'instruction (Bloc B dans la figure ci-dessous)
- **SINON** on effectuer une autre série d'instruction (Bloc C)



B. Les conditions en Python

Les structures conditionnelles sont utilisées pour exécuter différents blocs de code en fonction de l'évaluation de certaines conditions. En Python, elles sont implémentées à l'aide des mots-clés `if`, `elif` (abréviation de else if) et `else`.

1) La déclaration if (SI)

La déclaration `if` est le bloc conditionnel le plus simple. Le code indenté sous `if` n'est exécuté que si la condition est vraie (`True`)

La syntaxe est la suivante :

```
if condition :
    # Bloc de code à exécuter si la condition est Vraie
    #(ce bloc est indenté)
```

avec :

- ⇨ **`condition`** : C'est une expression qu'évalue à True (vrai) ou False (faux). Cela peut être une comparaison (x>10 par exemple), une opération logique (a and b par exemple), ou toute expression qui peut être interprétée comme un booléen.
- ⇨ **`:`** (deux-points) indique le début du bloc conditionnel
- ⇨ indentation : L'indentation (généralement de 4 espaces) est cruciale en Python. Tous les énoncés qui font partie du bloc if doivent avoir le même niveau d'indentation

Exemple :

```
age = 20
if age >= 18 :
    print("vous êtes majeur")
```

2) La déclaration else (SINON)

La déclaration `else` est utilisée en conjonction avec `if`. Le bloc de code sous `else` est exécuté si la condition du `if` précédent est `False` (faux).

La syntaxe est la suivante :

```
if condition :
    # Bloc de code si la condition est Vrai
else :
    # Bloc de code si la condition est Fausse
```

Exemple :

```
temperature = 15

if temperature > 25 :
    print("Alerte : Température élevée")
else :
    print("Température normale")
```

3) La déclaration elif (SINON SI)

La déclaration `elif` est utilisée pour tester des conditions supplémentaires si la condition `if` initiale (et tout `elif` précédent) est `False` (faux). Vous pouvez avoir plusieurs `elif` entre un `if` et un `else` (optionnel).

L'ordre des `elif` est important car Python évalue les conditions de haut en bas et exécute le premier bloc dont la condition est `True` (vraie), puis il sort de toute la structure conditionnelle.

La syntaxe est la suivante :

```
if Condition1 :
      # bloc de code si Condition1 est Vraie
elif Condition2 :
      # bloc de code si Condition2 est vraie
      # (et Condition1 était Fausse)
elif Condition3 :
      # bloc de code si Condition3 est Vraie
      # (et Condition1 et Condition2 étaient Fausse)
else :
      # bloc de code si aucune des conditions précédentes n'était
      # Vraie
```

Exemple :

```
note = 75
if note >= 90 :
      print("Excellente note : A")
elif note >= 80 :
      print("Très bien : B")
elif note >=70 :
      print("Bien : C")
elif note >= 60 :
      print("Passable : D")
else :
      print("Échec : F")
```

C. Opérateurs de comparaison et logiques

Les conditions utilisent souvent des opérateurs de comparaison et logiques :

- ⇨ Opérateurs de comparaison :
  - `==` : Égal à
  - `!=` : Différent de
  - `<` : Inférieur à
  - `>` : Supérieur à
  - `<=` : Inférieur ou égal à
  - `>=` : Supérieur ou égal à
- ⇨ Opérateurs logiques :
  - `and` : Vrai si les deux conditions sont Vraies
  - `or` : Vrai si au moins une des conditions est Vraie
  - `not` : Inverse la valeur de vérité d'une condition

Exemples :

```
heure = 14

jour_ferie = False

if heure >= 9 and heure <= 17 :
    print("C'est l'heure de travail.")

if heure < 9 or heure > 17 :
    print("En dehors des heures de travail.")

if not jour_ferie:
    print("C'est un jour ouvrable.")
```

```
age = 17

nationalite = "Français"

if age >= 18 and nationalite == "Français":
    print("Vous pouvez voter en France.")

elif age >= 18 and nationalite != "Français":
    print("Vous êtes majeur mais ne pouvez pas voter en France.")

else:
    print("Vous n'êtes pas encore majeur.")
```

D. Conditions imbriquées

Le bloc de code présent dans une condition peut contenir une structure conditionnelle à l'intérieur. On appelle cela les conditions imbriquées, il est encore plus critique d'être rigoureux sur l'indentation.

Exemple :

```
utilisateur_connecte = True
abonnement_premium = False
if utilisateur_connecte :
      print("Bienvenue, utilisateur !")
      if abonnement_premium :
            print("Accès à toutes les fonctionnalités")
      else :
            print("Accès limité aux fonctionnalités")
else :
      print("Veuillez vous connecter")
```

E. Erreurs fréquentes

⇨ Oublie des deux-points :

```
if age >= 18
        print("Tu es majeur")
```

```
SyntaxError: invalid syntax
```

⇨ Mauvaise indentation :

```
if age >= 18 :
print("Tu es majeur")
```

```
IndentationError: expected an indented block
```

⇨ Utilisation de = au lieu de == :

```
if age = 18 :
    print("Tu as 18 ans")
```

```
SyntaxError: invalid syntax
```

F. A retenir

if permet de tester une condition

si la condition est vraie → le bloc s'exécute

Indentation obligatoire :

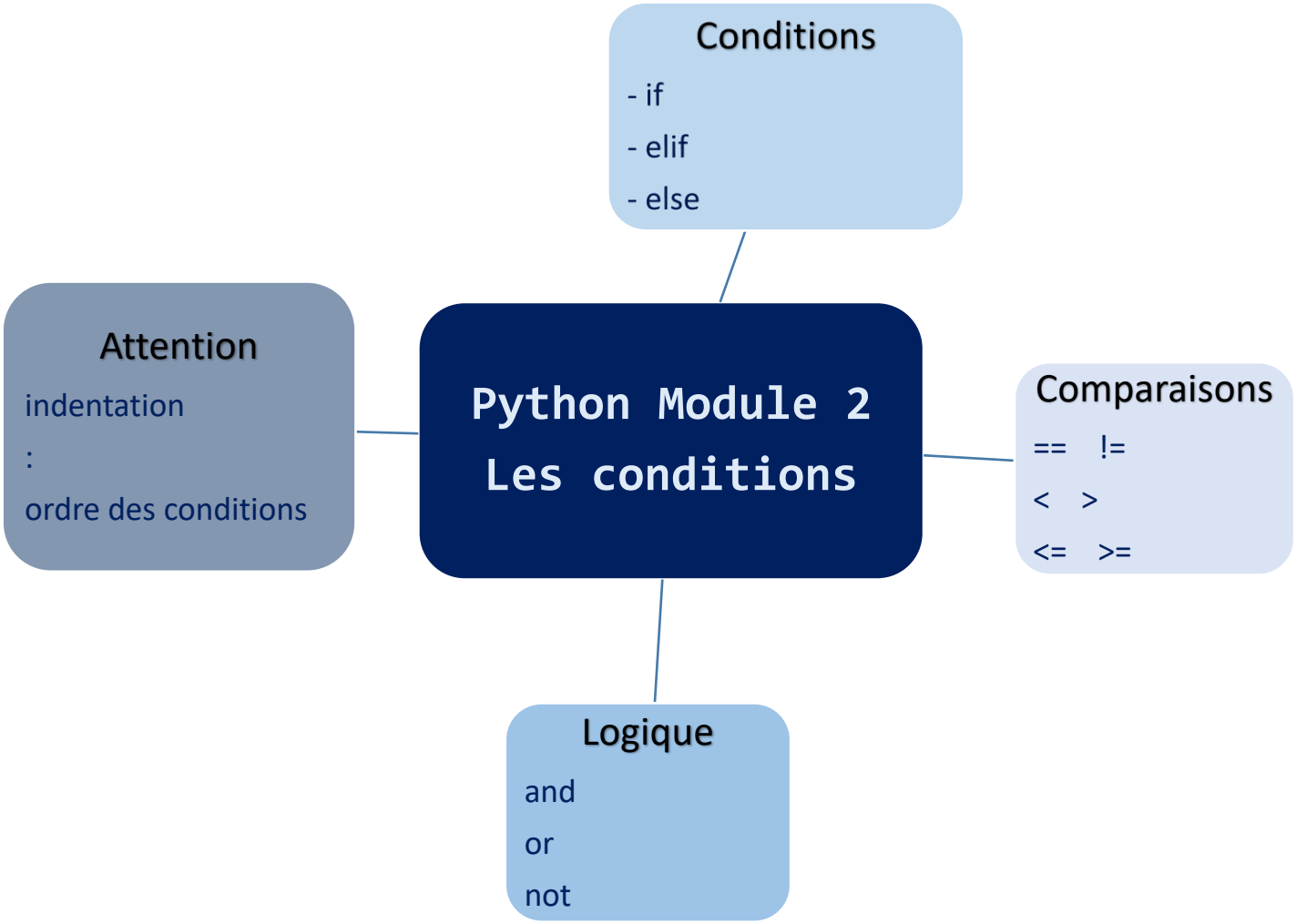
```
if condition :
    bloc instruction
    …
    if condition :
        bloc instruction
        …
    elif condition :
        bloc instruction
        ….
    else :
        bloc instruction
        …
else :
    bloc instruction
    …
```

elif permet de tester d'autre cas

else correspond au cas par défaut

Les conditions sont testées dans l'ordre.

G. Synthèse


Conditions
- if
- elif
- else
Attention
indentation
:
ordre des conditions
Python Module 2
Les conditions
Comparaisons
== !=
< >
<= >=
Logique
and
or
not