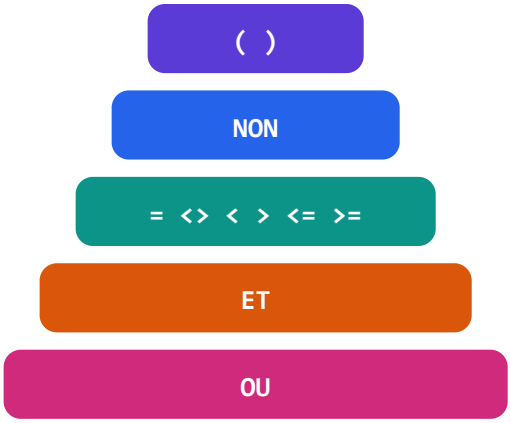
CHAPITRE 4

La priorité des opérateurs

L'ordre de calcul et les parenthèses



AU PROGRAMME

1. Pourquoi une priorité ?
2. La hiérarchie des opérations
3. Le piège du ET et du OU
4. Grouper avec NON et dans les SINONSI
5. Méthode : écrire une condition complexe

Nom : ..... **Classe :** .....

AVANT DE COMMENCER

Ce que tu vas apprendre

Une condition peut mélanger des comparaisons, des ET, des OU et des NON. L'ordinateur ne lit pas la phrase comme toi : il suit un ordre strict, la priorité des opérateurs. Ce chapitre te montre cet ordre et surtout comment le maîtriser avec des parenthèses.

1. **Citer l'ordre de priorité : parenthèses, NON, comparaisons, ET, OU.**
2. **Calculer une expression arithmétique en respectant la priorité.**
3. **Expliquer pourquoi le ET est évalué avant le OU, et le piège qui en découle.**
4. **Placer les parenthèses pour qu'une condition dise ce que l'on veut.**
5. **Appliquer NON à tout un groupe de conditions.**
6. **Traduire une règle en français en condition complète.**

PARTIE 1

Pourquoi une priorité ?

Tu sais déjà qu'en mathématiques, `2 + 3 * 4` vaut 14 et non 20 : la multiplication passe avant l'addition. L'ordinateur suit ce genre de règle pour **tout** : les calculs, les comparaisons et les opérateurs logiques.

Quand une condition comporte plusieurs opérateurs, la machine ne peut pas tout évaluer en même temps. Elle suit un **ordre fixé à l'avance** : c'est la priorité des opérateurs.

DÉFINITION

La priorité des opérateurs

L'ordre dans lequel l'ordinateur évalue les morceaux d'une expression. Les **parenthèses** permettent d'imposer un autre ordre.

D'abord les calculs

Expression	Ordre de calcul	Résultat
`2 + 3 * 4`	`3 * 4` d'abord, puis `2 + 12`	14
`(2 + 3) * 4`	la parenthèse d'abord : `5 * 4`	20
`10 - 4 - 3`	de gauche à droite : `6 - 3`	3
`7 + 9 mod 4`	`mod` comme `*` et `/` : `9 mod 4` d'abord	8

À RETENIR

Les calculs avant les comparaisons

- `*`, `/` et `mod` passent avant `+` et `-` ; à égalité, on va de gauche à droite.
- Dans `SI prix * 2 > budget ALORS`, le calcul est fait **avant** la comparaison.

PARTIE 2

La hiérarchie des opérations

Dans une condition, l'algorithme évalue toujours dans cet ordre, du plus prioritaire au moins prioritaire :

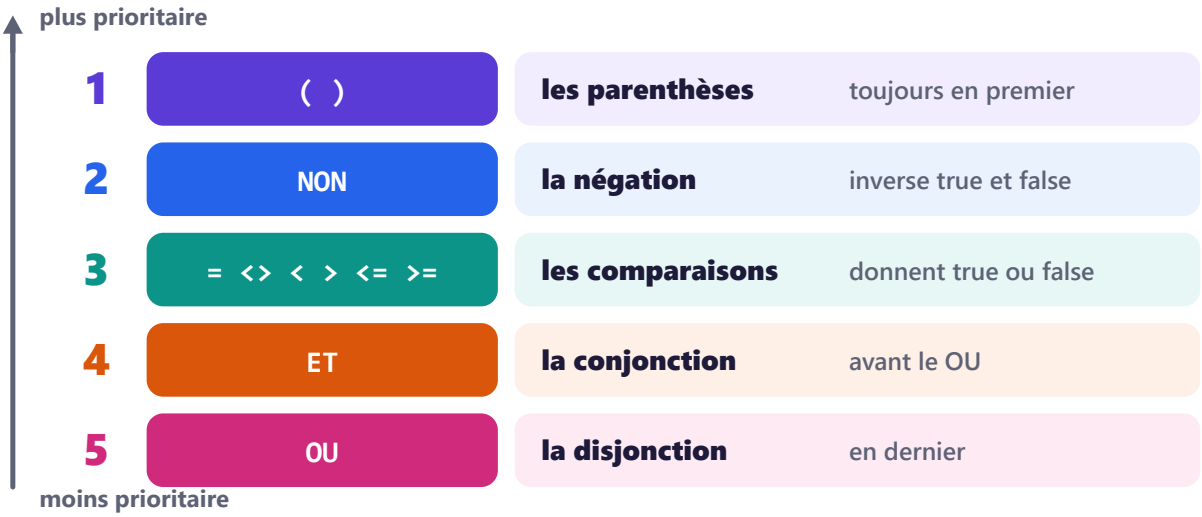


Figure 4.1 — Les cinq niveaux de priorité d'une condition.

Ordre	Opérateur	Ce qu'il fait
1	`( )`	Groupement : ce qui est entre parenthèses est évalué en premier.
2	`NON`	Négation : inverse true et false.
3	`=` `<>` `<` `>` `<=` `>=`	Comparaisons : transforment deux valeurs en true ou false.
4	`ET`	Conjonction : toujours avant le OU.
5	`OU`	Disjonction : évalué en dernier.

EXEMPLE

Dans `age < 12 ET ville = "Waremme"`, la machine calcule d'abord `age < 12` (true ou false) et `ville = "Waremme"` (true ou false), puis seulement le `ET` qui les relie.

ATTENTION

NON exige des parenthèses

Le `NON` passe **avant** les comparaisons. `NON age > 18` voudrait dire « (NON age) > 18 », ce qui n'a aucun sens : VS Code refuse ce code. Écris toujours `NON(age > 18)`.

POUR ALLER PLUS LOIN

Le `XOR` ne figure pas dans la hiérarchie du cours. En C#, il passe après les comparaisons et avant le ET. Le plus sûr : mettre des parenthèses dès qu'un XOR côtoie un ET ou un OU.

PARTIE 3

Le piège du ET et du OU

La confusion entre la priorité du ET et celle du OU cause l'immense majorité des erreurs de logique.

EXEMPLE

Le billet de cinéma

Le cinéma de Waremme donne une réduction si la personne est **jeune ou âgée** (moins de 12 ans ou plus de 65 ans) **et** si elle habite **à Waremme**.

PSEUDOCODE La version fausse

```
SI age < 12 OU age > 65 ET ville = "Waremme" ALORS
```

Comme le `ET` passe avant le `OU`, la machine lit : `age < 12 OU (age > 65 ET ville = "Waremme")`. Un enfant de 10 ans qui habite à Huy obtient la réduction : `age < 12` est vrai, et `true OU …` est toujours vrai.

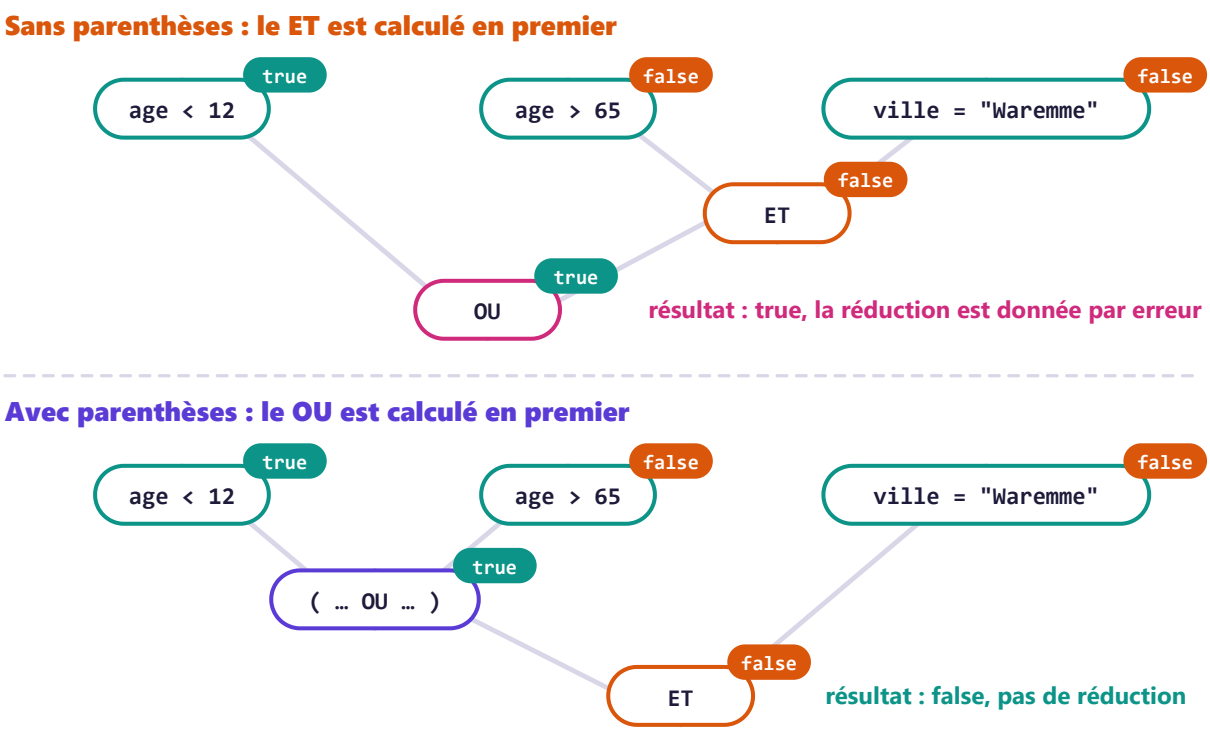


Figure 4.2 — La même condition, sans et avec parenthèses, pour un enfant de 10 ans qui habite à Huy.

PSEUDOCODE La version juste

```
SI (age < 12 OU age > 65) ET ville = "Waremme" ALORS
```

Âge et ville	Sans parenthèses	Avec parenthèses	Attendu
10 ans, Huy	true	false	pas de réduction
10 ans, Waremme	true	true	réduction
70 ans, Huy	false	false	pas de réduction
40 ans, Waremme	false	false	pas de réduction

À RETENIR

La règle d'or

- Dès qu'une condition mélange `ET` et `OU`, mets des **parenthèses** autour du `OU`.
- Une parenthèse en trop ne fait jamais de mal ; une parenthèse oubliée peut tout changer.

PARTIE 4

Grouper avec NON et dans les SINONSI

Le NON devant tout un groupe

Un mot de passe est refusé s'il vaut "password" ou "12345678". Pour dire « le mot de passe n'est **ni** l'un **ni** l'autre », on place le `NON` devant tout le groupe :

PSEUDOCODE

```
variable mdp en chaine de caracteres
DEBUT
    ECRIRE "Choisis un mot de passe :"
    LIRE mdp
    // valide s'il n'est ni "password" ni "12345678"
    SI NON(mdp = "password" OU mdp = "12345678") ALORS
        ECRIRE "Mot de passe accepté."
    SINON
        ECRIRE "Mot de passe trop connu !"
    FINSI
FIN
```

À L'ÉCRAN

```
Choisis un mot de passe :
12345678
Mot de passe trop connu !
```

DÉFINITION

La loi de De Morgan

`NON(A OU B)` est la même chose que `NON(A) ET NON(B)`. Et `NON(A ET B)` est la même chose que `NON(A) OU NON(B)`. Quand le NON entre dans la parenthèse, le OU devient ET et le ET devient OU.

PSEUDOCODE La même condition, sans NON

```
SI mdp <> "password" ET mdp <> "12345678" ALORS
```

Grouper dans un SINONSI

On recommande le film « Inception » aux spectateurs d'au moins 16 ans qui aiment l'action **ou** le drame :

```pseudocode
SI age >= 16 ET (genre = "action" OU genre = "drame") ALORS
    ECRIRE "Nous vous recommandons Inception."
SINONSI age >= 16 ALORS
    ECRIRE "Nous vous recommandons Interstellar."
SINON
    ECRIRE "Nous vous recommandons Vice-Versa."
FINSI
```

Sans les parenthèses, la machine calculerait `(age >= 16 ET genre = "action") OU genre = "drame"` : un enfant de 10 ans qui aime les drames recevrait la recommandation.

PARTIE 5

Méthode : écrire une condition complexe

Cinq étapes

1. **Découpe la règle**
   Repère chaque petite affirmation : « il est en 4TT », « il a 14 en maths »…

2. **Une comparaison par morceau**
   Traduis chaque affirmation : `annee = "4TT"`, `note_maths >= 14`.

3. **Relie les morceaux**
   « et » devient `ET`, « ou » devient `OU`, « ni… ni… » devient `NON(… OU …)`.

4. **Mets les parenthèses**
   Autour de chaque groupe `OU` qui est relié à un `ET`.

5. **Teste**
   Choisis quelques cas (dont des cas limites) et vérifie le résultat dans un tableau.

EXEMPLE

Exemple : l'option Robotique

Un élève peut s'inscrire à l'option Robotique s'il est **en 4TT** et s'il a obtenu **au moins 14** en mathématiques **ou** en sciences.

PSEUDOCODE

```
variable annee en chaine de caracteres
variables note_maths, note_sciences en entier
DEBUT
    ECRIRE "Ton année ?"
    LIRE annee
    ECRIRE "Ta note de mathématiques ?"
    LIRE note_maths
    ECRIRE "Ta note de sciences ?"
    LIRE note_sciences
    SI annee = "4TT" ET (note_maths >= 14 OU note_sciences >= 14) ALORS
        ECRIRE "Inscrit à Robotique."
    SINON
        ECRIRE "Non éligible à Robotique."
    FINSI
FIN
```

À L'ÉCRAN

```
Ton année ?
4TT
Ta note de mathématiques ?
12
Ta note de sciences ?
15
Inscrit à Robotique.
```

Sans la parenthèse, le `ET` lierait l'année aux seules mathématiques : un élève de 3TT avec 15 en sciences serait inscrit.

BILAN DU CHAPITRE 4

L'essentiel à retenir

1. La **priorité** est l'ordre dans lequel la machine évalue une expression ; les **parenthèses** imposent un autre ordre.
2. Dans les calculs, `*`, `/` et `mod` passent avant `+` et `-` ; les calculs passent avant les comparaisons.
3. Dans une condition : **( )**, puis **NON**, puis les **comparaisons**, puis **ET**, et enfin **OU**.
4. Le **ET passe avant le OU** : `A OU B ET C` se lit `A OU (B ET C)`. Mets des parenthèses autour du OU.
5. `NON` s'écrit devant des parenthèses : `NON(age > 18)`.
6. **De Morgan** : `NON(A OU B)` = `NON(A) ET NON(B)` ; `NON(A ET B)` = `NON(A) OU NON(B)`.
7. Pour une règle complexe : découpe, traduis chaque morceau, relie, groupe, puis **teste** des cas.

Le lexique

Priorité des opérateurs
L'ordre d'évaluation des morceaux d'une expression.

Groupement
Utiliser des parenthèses pour imposer l'ordre d'évaluation.

Négation
L'opérateur `NON`, qui inverse true et false.

Conjonction
L'opérateur `ET` : vrai si les deux conditions sont vraies.

Disjonction
L'opérateur `OU` : vrai si au moins une condition est vraie.

Loi de De Morgan
Règle pour faire entrer un NON dans une parenthèse : OU devient ET, ET devient OU.

Cas limite
Une valeur à la frontière d'une condition (18 pour « 18 ans et plus ») : à toujours tester.

Je fais le point

Coche une case par ligne : où en es-tu avant l'interro ?

Je suis capable de…	Oui	Presque	Pas encore
Calculer `2 + 3 * 4` et `(2 + 3) * 4` sans me tromper.	☐	☐	☐
Citer l'ordre : parenthèses, NON, comparaisons, ET, OU.	☐	☐	☐
Expliquer l'erreur du billet de cinéma.	☐	☐	☐
Ajouter les parenthèses qui manquent à une condition.	☐	☐	☐
Écrire `NON(…)` devant un groupe, ou l'équivalent sans NON.	☐	☐	☐
Traduire une règle en français en condition complète.	☐	☐	☐

CHAPITRE 4 · EXERCICES

Teste tes connaissances

avant l'interro

COMMENT T'Y PRENDRE

1. **Relis la synthèse** : l'essentiel, le lexique et « Je fais le point ».
2. **Fais les exercices sans le cours**, comme à l'interro.
3. **Corrige-toi** avec la correction vue en classe ou sur le site.
4. **Reviens sur tes erreurs** : relis la partie du cours concernée.

LES 5 EXERCICES

1. **L'ordre de priorité**
2. **Évalue les conditions**
3. **Parenthèses et NON**
4. **Ajoute les parenthèses**
5. **La bonne condition**

Exercice 1 L'ordre de priorité

▸ **Coche la bonne réponse.**

1. Que vaut `5 + 2 * 3` ?

☐ 21 ☐ 11 ☐ 13

2. Que vaut `(5 + 2) * 3` ?

☐ 21 ☐ 11 ☐ 13

3. Dans une condition sans parenthèses, qu'est-ce qui est évalué en dernier ?

☐ le ET ☐ le OU ☐ le NON ☐ les comparaisons

4. `true OU false ET false` vaut...

☐ true ☐ false

5. Comment écrire « l'âge n'est pas supérieur à 18 » ?

☐ `NON age > 18` ☐ `NON(age > 18)` ☐ `age NON > 18`

Exercice 2 Évalue les conditions

▸ **Les variables valent a = 3 et b = 9. Écris true ou false.**

Condition	Résultat
`b > 5 OU a > 5 ET a > b`	
`(b > 5 OU a > 5) ET a > b`	
`NON(a = 3 OU b = 3)`	
`NON(a = 3) ET NON(b = 3)`	
`a + b * 2 > 20`	

Exercice 3 Parenthèses et NON

- **Coche vrai ou faux. Si c'est faux, corrige la phrase sur la ligne.**

Affirmation	Vrai	Faux
Sans parenthèses, le ET est évalué avant le OU.	☐	☐
`NON(A OU B)` est équivalent à `NON(A) OU NON(B)`.	☐	☐
Ajouter des parenthèses autour d'une seule comparaison change le résultat.	☐	☐
Les calculs sont faits avant les comparaisons.	☐	☐

Exercice 4 Ajoute les parenthèses

- **Pour chaque règle, écris la condition correcte.**

1. Une boisson est gratuite pour les clients **fidèles** qui commandent un **menu ou un dessert**. Recopie la condition avec les parenthèses :
`SI fidele ET menu OU dessert ALORS`

2. Un parc refuse l'attraction si la personne mesure **moins de 120 cm** ou a **moins de 8 ans**, sauf si elle est **accompagnée**. Écris la condition d'accès refusé avec `taille`, `age` et le booléen `accompagne`.

Exercice 5 La bonne condition

▸ **Écris l'algorithme complet : déclarations, DEBUT, instructions, FIN.**

Une salle de sport est ouverte du lundi au vendredi (jours 1 à 5) de 8 h à 20 h, et le samedi (jour 6) de 9 h à 13 h. Demande le jour (de 1 à 7) et l'heure (de 0 à 23), puis affiche « Ouvert » ou « Fermé ».

EXEMPLE D'ÉCRAN

```
Jour (1 = lundi … 7 = dimanche) ?
6
Heure (0 à 23) ?
14
Fermé
```