

8. [Opérations Arithmétiques](#)
 9. [Manipulation de Chaînes](#)
 10. [Structures Conditionnelles](#)
 11. [Boucles et Itérations](#)
 12. [Gestion des Fichiers Séquentiels](#)
 13. [Exercices Pratiques Partie 3](#)
-

8. Opérations Arithmétiques

8.1 Les 4 Verbes Arithmétiques de Base

COBOL offre 5 façons de faire des calculs :

ADD → Addition

SUBTRACT → Soustraction

MULTIPLY → Multiplication

DIVIDE → Division

COMPUTE → Calcul avec formule (le plus puissant)

 ADD : Addition

Format 1 : ADD ... TO ...

ADD valeur1 TO variable2

*> Équivalent à : $\text{variable2} = \text{variable2} + \text{valeur1}$

Exemples :

01 WS-COMPTEUR PIC 9(5) VALUE 10.

01 WS-TOTAL PIC 9(7)V99 VALUE 100.50.

01 WS-BONUS PIC 9(5)V99 VALUE 50.25.

PROCEDURE DIVISION.

ADD 1 TO WS-COMPTEUR.

*> WS-COMPTEUR = 11

ADD 200 TO WS-TOTAL.

*> WS-TOTAL = 300.50

ADD WS-BONUS TO WS-TOTAL.

*> WS-TOTAL = 350.75

ADD 5 TO WS-COMPTEUR.

*> WS-COMPTEUR = 16

Format 2 : ADD ... TO ... GIVING ...

ADD valeur1 TO valeur2 GIVING variable3

*> Équivalent à : $\text{variable3} = \text{valeur1} + \text{valeur2}$

*> (valeur1 et valeur2 restent inchangées)

Exemples :

01 WS-PRIX-HT PIC 9(5)V99 VALUE 100.00.

01 WS-TVA PIC 9(5)V99 VALUE 20.00.

01 WS-PRIX-TTC PIC 9(5)V99.

PROCEDURE DIVISION.

ADD WS-PRIX-HT TO WS-TVA GIVING WS-PRIX-TTC.

*> WS-PRIX-TTC = 120.00

*> WS-PRIX-HT et WS-TVA sont inchangés

DISPLAY "Prix HT : " WS-PRIX-HT. *> 100.00

DISPLAY "TVA : " WS-TVA. *> 20.00

DISPLAY "Prix TTC : " WS-PRIX-TTC. *> 120.00

Format 3 : ADD Multiple (Addition de plusieurs valeurs)

ADD valeur1 valeur2 valeur3 ... GIVING variable-résultat

*> Équivalent à : résultat = valeur1 + valeur2 + valeur3 + ...

Exemples :

01 WS-NOTE1 PIC 99V99 VALUE 15.50.

01 WS-NOTE2 PIC 99V99 VALUE 12.75.

01 WS-NOTE3 PIC 99V99 VALUE 16.00.

01 WS-NOTE4 PIC 99V99 VALUE 14.25.

01 WS-TOTAL-NOTES PIC 999V99.

PROCEDURE DIVISION.

ADD WS-NOTE1 WS-NOTE2 WS-NOTE3 WS-NOTE4

GIVING WS-TOTAL-NOTES.

*> WS-TOTAL-NOTES = 58.50

DISPLAY "Total des notes : " WS-TOTAL-NOTES.

⚠ Pièges Courants avec ADD :

* ❌ ERREUR 1 : Oublier TO

ADD 10 WS-COMPTEUR. *> ERREUR DE SYNTAXE!

* ✅ Correct :

ADD 10 TO WS-COMPTEUR.

*  ERREUR 2 : Inverser TO et GIVING

ADD WS-A GIVING WS-B TO WS-C. *> ERREUR!

*  Correct :

ADD WS-A TO WS-B GIVING WS-C.

*  ERREUR 3 : Débordement non géré

01 WS-PETIT PIC 99 VALUE 99.

ADD 1 TO WS-PETIT. *> Débordement! (100 > 99 max)

*  Correct : Utiliser ON SIZE ERROR

ADD 1 TO WS-PETIT

 ON SIZE ERROR

 DISPLAY "ERREUR : Débordement!"

END-ADD.

— SUBTRACT : Soustraction

Format 1 : SUBTRACT ... FROM ...

SUBTRACT valeur1 FROM variable2

*> Équivalent à : $\text{variable2} = \text{variable2} - \text{valeur1}$

Exemples :

01 WS-SOLDE PIC 9(7)V99 VALUE 1000.00.

01 WS-RETRAIT PIC 9(5)V99 VALUE 250.50.

PROCEDURE DIVISION.

 SUBTRACT 100 FROM WS-SOLDE.

*> WS-SOLDE = 900.00

SUBTRACT WS-RETRAIT FROM WS-SOLDE.

*> WS-SOLDE = 649.50

DISPLAY "Solde restant : " WS-SOLDE.

Format 2 : SUBTRACT ... FROM ... GIVING ...

SUBTRACT valeur1 FROM valeur2 GIVING variable3

*> Équivalent à : $\text{variable3} = \text{valeur2} - \text{valeur1}$

Exemples :

01 WS-PRIX-TTC PIC 9(5)V99 VALUE 120.00.

01 WS-TVA PIC 9(5)V99 VALUE 20.00.

01 WS-PRIX-HT PIC 9(5)V99.

PROCEDURE DIVISION.

SUBTRACT WS-TVA FROM WS-PRIX-TTC GIVING WS-PRIX-HT.

*> WS-PRIX-HT = 100.00

DISPLAY "Prix HT calculé : " WS-PRIX-HT.

Format 3 : SUBTRACT Multiple

SUBTRACT valeur1 valeur2 valeur3 FROM variable

*> Équivalent à : $\text{variable} = \text{variable} - \text{valeur1} - \text{valeur2} - \text{valeur3}$

Exemples :

01 WS-SALAIRE-BRUT PIC 9(5)V99 VALUE 3000.00.

01 WS-COTIS-SOC PIC 9(4)V99 VALUE 450.00.

01 WS-IMPOT PIC 9(4)V99 VALUE 350.00.

01 WS-MUTUELLE PIC 9(3)V99 VALUE 50.00.

PROCEDURE DIVISION.

SUBTRACT WS-COTIS-SOC WS-IMPOT WS-MUTUELLE
FROM WS-SALAIRE-BRUT.

*> WS-SALAIRE-BRUT = 2150.00 (salaire net)

DISPLAY "Salaire net : " WS-SALAIRE-BRUT.

✖ MULTIPLY : Multiplication

Format 1 : MULTIPLY ... BY ...

MULTIPLY valeur1 BY variable2

*> Équivalent à : $\text{variable2} = \text{variable2} * \text{valeur1}$

Exemples :

01 WS-PRIX-UNIT PIC 999V99 VALUE 25.50.

01 WS-QUANTITE PIC 999 VALUE 10.

PROCEDURE DIVISION.

MULTIPLY 2 BY WS-QUANTITE.

*> WS-QUANTITE = 20

MULTIPLY WS-QUANTITE BY WS-PRIX-UNIT.

*> WS-PRIX-UNIT = 510.00

Format 2 : MULTIPLY ... BY ... GIVING ...

MULTIPLY valeur1 BY valeur2 GIVING variable3

*> Équivalent à : $\text{variable3} = \text{valeur1} * \text{valeur2}$

Exemples :

01 WS-PRIX-UNIT PIC 999V99 VALUE 25.50.

01 WS-QUANTITE PIC 999 VALUE 10.

01 WS-TOTAL PIC 9(5)V99.

PROCEDURE DIVISION.

MULTIPLY WS-PRIX-UNIT BY WS-QUANTITE GIVING WS-TOTAL.

*> WS-TOTAL = 255.00

*> WS-PRIX-UNIT et WS-QUANTITE restent inchangés

DISPLAY "Total : " WS-TOTAL.

💡 Calcul de TVA - Exemple Pratique :

01 WS-PRIX-HT PIC 9(5)V99 VALUE 100.00.

01 WS-TAUX-TVA PIC V99 VALUE 0.20. *> 20%

01 WS-MONTANT-TVA PIC 9(5)V99.

01 WS-PRIX-TTC PIC 9(5)V99.

PROCEDURE DIVISION.

*--- Calculer le montant de TVA ---

MULTIPLY WS-PRIX-HT BY WS-TAUX-TVA GIVING WS-MONTANT-TVA.

*> WS-MONTANT-TVA = 20.00

*--- Calculer le prix TTC ---

ADD WS-PRIX-HT TO WS-MONTANT-TVA GIVING WS-PRIX-TTC.

*> WS-PRIX-TTC = 120.00

DISPLAY "Prix HT : " WS-PRIX-HT.

DISPLAY "TVA (20%) : " WS-MONTANT-TVA.

DISPLAY "Prix TTC : " WS-PRIX-TTC.

÷ DIVIDE : Division

Format 1 : DIVIDE ... INTO ...

DIVIDE valeur1 INTO variable2

*> Équivalent à : $\text{variable2} = \text{variable2} / \text{valeur1}$

Exemples :

01 WS-TOTAL PIC 9(5)V99 VALUE 1000.00.

01 WS-DIVISEUR PIC 99 VALUE 4.

PROCEDURE DIVISION.

DIVIDE 2 INTO WS-TOTAL.

*> WS-TOTAL = 500.00

DIVIDE WS-DIVISEUR INTO WS-TOTAL.

*> WS-TOTAL = 125.00

Format 2 : DIVIDE ... INTO ... GIVING ...

DIVIDE valeur1 INTO valeur2 GIVING variable3

*> Équivalent à : $\text{variable3} = \text{valeur2} / \text{valeur1}$

Exemples :

01 WS-TOTAL-NOTES PIC 999V99 VALUE 58.50.

01 WS-NB-NOTES PIC 99 VALUE 4.

01 WS-MOYENNE PIC 99V99.

PROCEDURE DIVISION.

DIVIDE WS-NB-NOTES INTO WS-TOTAL-NOTES GIVING WS-MOYENNE.

*> WS-MOYENNE = 14.62 (arrondi)

DISPLAY "Moyenne : " WS-MOYENNE " / 20".

Format 3 : DIVIDE ... BY ... GIVING ...

DIVIDE valeur1 BY valeur2 GIVING variable3

*> Équivalent à : $\text{variable3} = \text{valeur1} / \text{valeur2}$

Exemples :

01 WS-DISTANCE PIC 9(5) VALUE 450. *> km

01 WS-TEMPS PIC 99V9 VALUE 5.5. *> heures

01 WS-VITESSE PIC 999V9. *> km/h

PROCEDURE DIVISION.

DIVIDE WS-DISTANCE BY WS-TEMPS GIVING WS-VITESSE.

*> WS-VITESSE = 81.8 km/h

DISPLAY "Vitesse moyenne : " WS-VITESSE " km/h".

Format 4 : DIVIDE avec REMAINDER (Reste de la division)

DIVIDE valeur1 BY valeur2

GIVING quotient

REMAINDER reste

Exemples :

01 WS-TOTAL PIC 999 VALUE 100.

01 WS-DIVISEUR PIC 99 VALUE 7.

01 WS-QUOTIENT PIC 999.

01 WS-RESTE PIC 99.

PROCEDURE DIVISION.

DIVIDE WS-TOTAL BY WS-DIVISEUR

GIVING WS-QUOTIENT

REMAINDER WS-RESTE.

*> WS-QUOTIENT = 14

*> WS-RESTE = 2

*> Car $100 = (14 \times 7) + 2$

DISPLAY "100 / 7 = " WS-QUOTIENT " reste " WS-RESTE.

💡 Application Pratique : Conversion en Heures/Minutes

01 WS-MINUTES-TOTAL PIC 9(5) VALUE 725. *> 725 minutes

01 WS-HEURES PIC 999.

01 WS-MINUTES PIC 99.

PROCEDURE DIVISION.

DIVIDE WS-MINUTES-TOTAL BY 60

GIVING WS-HEURES

REMAINDER WS-MINUTES.

*> WS-HEURES = 12

*> WS-MINUTES = 5

*> Donc 725 minutes = 12h05

DISPLAY WS-MINUTES-TOTAL " minutes = "
WS-HEURES "h" WS-MINUTES.

⚠ Division par Zéro :

01 WS-DIVIDENDE PIC 999 VALUE 100.

01 WS-DIVISEUR PIC 99 VALUE 0.

01 WS-RESULTAT PIC 999.

PROCEDURE DIVISION.

* ❌ DANGER : Plantage du programme!

DIVIDE WS-DIVIDENDE BY WS-DIVISEUR GIVING WS-RESULTAT.

* ✅ CORRECT : Toujours tester avant

IF WS-DIVISEUR = ZERO

DISPLAY "ERREUR : Division par zéro impossible!"

MOVE ZERO TO WS-RESULTAT

ELSE

DIVIDE WS-DIVIDENDE BY WS-DIVISEUR GIVING WS-RESULTAT

END-IF.

* ✅ OU utiliser ON SIZE ERROR

DIVIDE WS-DIVIDENDE BY WS-DIVISEUR GIVING WS-RESULTAT

ON SIZE ERROR

DISPLAY "ERREUR : Division impossible"

MOVE ZERO TO WS-RESULTAT

END-DIVIDE.

 COMPUTE : Le Plus Puissant

COMPUTE permet d'écrire des formules complètes !

COMPUTE variable = expression-mathématique

Opérateurs Disponibles :

Opérateur	Signification	Priorité	Exemple
()	Parenthèses	1 (plus haute)	(A + B)
**	Puissance	2	A ** 2
*	Multiplication	3	A * B
/	Division	3	A / B
+	Addition	4	A + B
-	Soustraction	4 (plus basse)	A - B

Exemples Simples :

01 WS-A PIC 999V99 VALUE 10.

01 WS-B PIC 999V99 VALUE 5.

01 WS-C PIC 999V99 VALUE 2.

01 WS-RESULTAT PIC 9(5)V99.

PROCEDURE DIVISION.

*--- Addition simple ---

COMPUTE WS-RESULTAT = WS-A + WS-B.

*> WS-RESULTAT = 15.00

*--- Formule complexe ---

COMPUTE WS-RESULTAT = WS-A + WS-B * WS-C.

*> WS-RESULTAT = 20.00 (10 + 5*2 = 10 + 10)

*--- Avec parenthèses ---

COMPUTE WS-RESULTAT = (WS-A + WS-B) * WS-C.

*> WS-RESULTAT = 30.00 ((10 + 5) * 2 = 15 * 2)

*--- Puissance ---

COMPUTE WS-RESULTAT = WS-A ** 2.

*> WS-RESULTAT = 100.00 (10²)

Exemple Pratique : Calcul d'Intérêts Composés

* Formule : Montant Final = Capital × (1 + Taux)^{Années}

01 WS-CAPITAL-INITIAL PIC 9(7)V99 VALUE 10000.00.

01 WS-TAUX-ANNUEL PIC V9999 VALUE 0.0350. *> 3.5%

01 WS-NB-ANNEES PIC 99 VALUE 5.

01 WS-MONTANT-FINAL PIC 9(9)V99.

01 WS-INTERETS PIC 9(7)V99.

PROCEDURE DIVISION.

COMPUTE WS-MONTANT-FINAL =

WS-CAPITAL-INITIAL * (1 + WS-TAUX-ANNUEL) ** WS-NB-ANNEES.

*> WS-MONTANT-FINAL = 11876.86

COMPUTE WS-INTERETS = WS-MONTANT-FINAL - WS-CAPITAL-INITIAL.

*> WS-INTERETS = 1876.86

DISPLAY "Capital initial : " WS-CAPITAL-INITIAL " €".

DISPLAY "Taux annuel : 3.5%".

DISPLAY "Durée : " WS-NB-ANNEES " ans".

DISPLAY "Montant final : " WS-MONTANT-FINAL " €".

DISPLAY "Intérêts gagnés : " WS-INTERETS " €".

 Exemples Avancés avec COMPUTE :

1. Calcul de Moyenne avec Pondération :

01 WS-NOTE-MATHS PIC 99V99 VALUE 15.50.

01 WS-NOTE-FRANCAIS PIC 99V99 VALUE 14.00.

01 WS-NOTE-ANGLAIS PIC 99V99 VALUE 16.50.

01 WS-COEF-MATHS PIC 9 VALUE 3.

01 WS-COEF-FRANCAIS PIC 9 VALUE 2.

01 WS-COEF-ANGLAIS PIC 9 VALUE 2.

01 WS-MOYENNE PIC 99V99.

PROCEDURE DIVISION.

COMPUTE WS-MOYENNE =

(WS-NOTE-MATHS * WS-COEF-MATHS +

WS-NOTE-FRANCAIS * WS-COEF-FRANCAIS +

WS-NOTE-ANGLAIS * WS-COEF-ANGLAIS) /

(WS-COEF-MATHS + WS-COEF-FRANCAIS + WS-COEF-ANGLAIS).

*> WS-MOYENNE = 15.28

DISPLAY "Moyenne pondérée : " WS-MOYENNE " / 20".

2. Calcul de TVA (méthode rapide) :

01 WS-PRIX-HT PIC 9(5)V99 VALUE 100.00.

01 WS-TAUX-TVA PIC V99 VALUE 0.20. * > 20%

01 WS-PRIX-TTC PIC 9(5)V99.

PROCEDURE DIVISION.

COMPUTE WS-PRIX-TTC = WS-PRIX-HT * (1 + WS-TAUX-TVA).

* > WS-PRIX-TTC = 120.00

3. Calcul de Remise :

01 WS-PRIX-INITIAL PIC 9(5)V99 VALUE 250.00.

01 WS-REMISE PIC V99 VALUE 0.15. * > 15% de remise

01 WS-PRIX-FINAL PIC 9(5)V99.

PROCEDURE DIVISION.

COMPUTE WS-PRIX-FINAL = WS-PRIX-INITIAL * (1 - WS-REMISE).

* > WS-PRIX-FINAL = 212.50

DISPLAY "Prix initial : " WS-PRIX-INITIAL " €".

DISPLAY "Remise 15% : -" (WS-PRIX-INITIAL * WS-REMISE) " €".

DISPLAY "Prix final : " WS-PRIX-FINAL " €".

8.2 Gestion des Erreurs Arithmétiques

ON SIZE ERROR

Protège contre les débordements :

COMPUTE variable = expression

ON SIZE ERROR

instructions-si-erreur

NOT ON SIZE ERROR

instructions-si-ok

END-COMPUTE.

Exemple Complet :

01 WS-PETIT-NOMBRE PIC 99 VALUE 95.

01 WS-RESULTAT PIC 99.

PROCEDURE DIVISION.

COMPUTE WS-RESULTAT = WS-PETIT-NOMBRE + 10

ON SIZE ERROR

DISPLAY "ERREUR : Résultat trop grand (> 99)!"

DISPLAY "Calcul : " WS-PETIT-NOMBRE " + 10 = 105"

MOVE 99 TO WS-RESULTAT *> Valeur max

NOT ON SIZE ERROR

DISPLAY "Calcul OK : " WS-RESULTAT

END-COMPUTE.

DISPLAY "Résultat final : " WS-RESULTAT.

Avec tous les verbes arithmétiques :

*--- ADD avec protection ---

ADD WS-A TO WS-B

ON SIZE ERROR

DISPLAY "Débordement dans ADD"

END-ADD.

*--- SUBTRACT avec protection ---

SUBTRACT WS-A FROM WS-B

ON SIZE ERROR

DISPLAY "Débordement dans SUBTRACT"

END-SUBTRACT.

*--- MULTIPLY avec protection ---

MULTIPLY WS-A BY WS-B GIVING WS-C

ON SIZE ERROR

DISPLAY "Débordement dans MULTIPLY"

END-MULTIPLY.

*--- DIVIDE avec protection (division par zéro) ---

DIVIDE WS-A BY WS-B GIVING WS-C

ON SIZE ERROR

DISPLAY "Erreur division (zéro ou débordement)"

MOVE ZERO TO WS-C

END-DIVIDE.

 **ROUNDED** : Arrondi

Arrondit le résultat au nombre de décimales de la variable :

COMPUTE variable **ROUNDED** = expression

Exemples :

01 WS-DIVIDENDE PIC 999 VALUE 100.

01 WS-DIVISEUR PIC 99 VALUE 3.

01 WS-RESULTAT PIC 999V99.

01 WS-RESULTAT-ARR PIC 999V99.

PROCEDURE DIVISION.

*--- Sans ROUNDED ---

COMPUTE WS-RESULTAT = WS-DIVIDENDE / WS-DIVISEUR.

*> WS-RESULTAT = 33.33 (troncature)

*--- Avec ROUNDED ---

COMPUTE WS-RESULTAT-ARR ROUNDED =

WS-DIVIDENDE / WS-DIVISEUR.

*> WS-RESULTAT-ARR = 33.33 (arrondi mathématique)

DISPLAY "Sans arrondi : " WS-RESULTAT.

DISPLAY "Avec arrondi : " WS-RESULTAT-ARR.

💡 Règles d'Arrondi COBOL :

Si la première décimale abandonnée est :

- 0-4 : Arrondi inférieur (33.334 → 33.33)

- 5-9 : Arrondi supérieur (33.335 → 33.34)

Exemple complet :

01 WS-VALEUR PIC 999V999.

01 WS-ARRONDI PIC 999V99.

MOVE 123.456 TO WS-VALEUR.

COMPUTE WS-ARRONDI ROUNDED = WS-VALEUR.

*> Résultat : 123.46 (0.456 → 0.46 car 6 > 5)

MOVE 123.454 TO WS-VALEUR.

COMPUTE WS-ARRONDI ROUNDED = WS-VALEUR.

*> Résultat : 123.45 (0.454 → 0.45 car 4 < 5)

8.3 Fonctions Intrinsèques

COBOL offre des fonctions mathématiques prédéfinies :

Fonctions Mathématiques

FUNCTION ABS(nombre) → Valeur absolue

FUNCTION MAX(n1, n2, ...) → Maximum

FUNCTION MIN(n1, n2, ...) → Minimum

FUNCTION MOD(n1, n2) → Modulo (reste)

FUNCTION REM(n1, n2) → Remainder (reste)

FUNCTION INTEGER(nombre) → Partie entière

FUNCTION INTEGER-PART(nombre) → Partie entière

FUNCTION SQRT(nombre) → Racine carrée

FUNCTION SIN(angle) → Sinus (radians)

FUNCTION COS(angle) → Cosinus (radians)

FUNCTION TAN(angle) → Tangente (radians)

FUNCTION LOG(nombre) → Logarithme naturel

FUNCTION LOG10(nombre) → Logarithme base 10

FUNCTION EXP(nombre) → Exponentielle (e^x)

Exemples Pratiques :

01 WS-NOMBRE PIC S999V99 VALUE -45.67.

01 WS-A PIC 999V99 VALUE 10.50.

01 WS-B PIC 999V99 VALUE 7.25.

01 WS-C PIC 999V99 VALUE 15.00.

01 WS-RESULTAT PIC 9(5)V99.

PROCEDURE DIVISION.

*--- Valeur absolue ---

COMPUTE WS-RESULTAT = FUNCTION ABS(WS-NOMBRE).

*> WS-RESULTAT = 45.67

*--- Maximum de plusieurs valeurs ---

COMPUTE WS-RESULTAT = FUNCTION MAX(WS-A, WS-B, WS-C).

*> WS-RESULTAT = 15.00

*--- Minimum ---

COMPUTE WS-RESULTAT = FUNCTION MIN(WS-A, WS-B, WS-C).

*> WS-RESULTAT = 7.25

*--- Modulo (reste de division) ---

COMPUTE WS-RESULTAT = FUNCTION MOD(100, 7).

*> WS-RESULTAT = 2 (car $100 = 14 \times 7 + 2$)

*--- Racine carrée ---

COMPUTE WS-RESULTAT = FUNCTION SQRT(144).

*> WS-RESULTAT = 12.00

*--- Partie entière ---

COMPUTE WS-RESULTAT = FUNCTION INTEGER(45.67).

*> WS-RESULTAT = 45.00

 Application : Calcul de Prêt Bancaire

* Calcul de Mensualité de Prêt

* Formule : $M = C \times (t / (1 - (1 + t)^{-n}))$

* Où : M = mensualité, C = capital, t = taux/12, n = nb mois

01 WS-CAPITAL PIC 9(7)V99 VALUE 200000.00.

01 WS-TAUX-ANNUEL PIC V9999 VALUE 0.0350. *> 3.5%

01 WS-NB-ANNEES PIC 99 VALUE 20.

01 WS-TAUX-MENSUEL PIC V9(6).

01 WS-NB-MOIS PIC 9(5).

01 WS-MENSUALITE PIC 9(5)V99.

01 WS-COUT-TOTAL PIC 9(9)V99.

01 WS-COUT-CREDIT PIC 9(7)V99.

PROCEDURE DIVISION.

*--- Calculer taux mensuel ---

COMPUTE WS-TAUX-MENSUEL = WS-TAUX-ANNUEL / 12.

*--- Calculer nombre de mois ---

COMPUTE WS-NB-MOIS = WS-NB-ANNEES * 12.

*--- Calculer mensualité ---

COMPUTE WS-MENSUALITE ROUNDED =

WS-CAPITAL * (WS-TAUX-MENSUEL /

(1 - (1 + WS-TAUX-MENSUEL) ** (-WS-NB-MOIS))).

*--- Calculer coût total ---

COMPUTE WS-COUT-TOTAL = WS-MENSUALITE * WS-NB-MOIS.

*--- Calculer coût du crédit ---

COMPUTE WS-COUT-CREDIT = WS-COUT-TOTAL - WS-CAPITAL.

*--- Afficher ---

DISPLAY "=====

DISPLAY " SIMULATION DE PRÊT IMMOBILIER "

DISPLAY "=====

DISPLAY "Capital emprunté : " WS-CAPITAL " €".

DISPLAY "Taux annuel : 3.50 %".

DISPLAY "Durée : " WS-NB-ANNEES " ans".

DISPLAY "-----

DISPLAY "Mensualité : " WS-MENSUALITE " €".

DISPLAY "Coût total : " WS-COUT-TOTAL " €".

DISPLAY "Coût du crédit : " WS-COUT-CREDIT " €".

DISPLAY "=====

Sortie :

=====

SIMULATION DE PRÊT IMMOBILIER

=====

Capital emprunté : 200000.00 €

Taux annuel : 3.50 %

Durée : 20 ans

Mensualité : 1159.58 €

Coût total : 278299.20 €

Coût du crédit : 78299.20 €

=====

9. Manipulation de Chaînes

9.1 MOVE et Règles d'Alignement

MOVE Simple

MOVE source TO destination

Règles d'Alignement selon le Type :

*=== NUMÉRIQUE : Alignement à DROITE ===

01 WS-SOURCE PIC 999 VALUE 123.

01 WS-DEST PIC 9(5).

MOVE WS-SOURCE TO WS-DEST.

*> Résultat : 00123 (zéros à gauche)

*=== ALPHANUMÉRIQUE : Alignement à GAUCHE ===

01 WS-SOURCE PIC X(5) VALUE "HELLO".

01 WS-DEST PIC X(10).

MOVE WS-SOURCE TO WS-DEST.

*> Résultat : "HELLO " (espaces à droite)

*=== Si destination plus petite : TRONCATURE ===

01 WS-SOURCE PIC X(10) VALUE "BONJOUR123".

01 WS-DEST PIC X(5).

MOVE WS-SOURCE TO WS-DEST.

*> Résultat : "BONJO" (troncature à droite pour alphanum)

💡 Exemples Détaillés :

DATA DIVISION.

WORKING-STORAGE SECTION.

*--- Variables numériques ---

01 WS-NUM-SOURCE PIC 999 VALUE 42.

01 WS-NUM-DEST-GRAND PIC 9(6).

01 WS-NUM-DEST-PETIT PIC 99.

*--- Variables alphanumériques ---

01 WS-ALPHA-SOURCE PIC X(10) VALUE "COBOL".

01 WS-ALPHA-DEST-GRAND PIC X(15).

01 WS-ALPHA-DEST-PETIT PIC X(3).

PROCEDURE DIVISION.

*--- Numérique : Alignement droite ---

MOVE WS-NUM-SOURCE TO WS-NUM-DEST-GRAND.

*> Résultat : 000042

DISPLAY "Num grand : [" WS-NUM-DEST-GRAND "]".

MOVE WS-NUM-SOURCE TO WS-NUM-DEST-PETIT.

*> Résultat : 42 (OK, rentre)

DISPLAY "Num petit : [" WS-NUM-DEST-PETIT "]".

*--- Si source trop grande (troncature à gauche pour numérique) ---

```
MOVE 12345 TO WS-NUM-DEST-PETIT.
```

*> Résultat : 45 (troncature des chiffres de gauche!)

```
DISPLAY "Num tronqué : [" WS-NUM-DEST-PETIT "].
```

*--- Alphanumérique : Alignement gauche ---

```
MOVE WS-ALPHA-SOURCE TO WS-ALPHA-DEST-GRAND.
```

*> Résultat : "COBOL "

```
DISPLAY "Alpha grand : [" WS-ALPHA-DEST-GRAND "].
```

```
MOVE WS-ALPHA-SOURCE TO WS-ALPHA-DEST-PETIT.
```

*> Résultat : "COB" (troncature à droite)

```
DISPLAY "Alpha petit : [" WS-ALPHA-DEST-PETIT "].
```

MOVE CORRESPONDING (CORR)

Copie automatique entre structures ayant des champs de même nom :

```
MOVE CORRESPONDING structure-source TO structure-dest
```

*> Ou la forme abrégée :

```
MOVE CORR structure-source TO structure-dest
```

Exemple :

```
01 EMPLOYE-FICHER.
```

```
05 EMP-F-ID      PIC 9(6).
```

```
05 EMP-F-NOM     PIC X(30).
```

```
05 EMP-F-PRENOM  PIC X(20).
```

```
05 EMP-F-SALAIRE PIC 9(7)V99.
```

```
05 EMP-F-DATE-EMB PIC X(10).
```

01 EMPLOYE-ECRAN.

05 EMP-E-ID PIC 9(6).

05 EMP-E-NOM PIC X(30).

05 EMP-E-PRENOM PIC X(20).

05 EMP-E-ADRESSE PIC X(50). *> Pas dans fichier

05 EMP-E-TELEPHONE PIC X(15). *> Pas dans fichier

PROCEDURE DIVISION.

*--- Copie manuelle (longue) ---

MOVE EMP-F-ID TO EMP-E-ID.

MOVE EMP-F-NOM TO EMP-E-NOM.

MOVE EMP-F-PRENOM TO EMP-E-PRENOM.

*> Fastidieux!

*--- MOVE CORR (automatique) ---

MOVE CORR EMPLOYE-FICHER TO EMPLOYE-ECRAN.

*> Copie automatiquement ID, NOM, PRENOM

*> (les noms doivent correspondre exactement)

⚠ Attention avec MOVE CORR :

✅ Copie SI :

- Même nom de champ (exact, case-insensitive)

- Types compatibles


❌ Ne copie PAS :

- FILLER

- Champs de noms différents
- Niveaux 66, 77, 88

Exemple :

01 SOURCE.

05 SALAIRE PIC 9(5)V99.  Copié

05 MONTANT PIC 9(5)V99.  Nom différent dans dest

05 FILLER PIC X(10).  FILLER jamais copié

01 DEST.

05 SALAIRE PIC 9(7)V99.  Reçoit la valeur

05 TOTAL PIC 9(5)V99.  Nom différent

9.2 STRING : Concaténation

Assembler plusieurs chaînes en une seule :

STRING source1 source2 source3 ...

DELIMITED BY délimiteur

INTO destination

[WITH POINTER pointeur]

[ON OVERFLOW instructions]

END-STRING

Exemples de Base :

01 WS-PRENOM PIC X(20) VALUE "Jean".

01 WS-NOM PIC X(30) VALUE "Dupont".

01 WS-NOM-COMPLET PIC X(50).

01 WS-ESPACE PIC X VALUE SPACE.

PROCEDURE DIVISION.

*--- Concaténer avec SIZE (tout le contenu) ---

STRING

WS-PRENOM DELIMITED BY SIZE

WS-ESPACE DELIMITED BY SIZE

WS-NOM DELIMITED BY SIZE

INTO WS-NOM-COMPLET

END-STRING.

*> Résultat : "Jean Dupont"

DISPLAY "Nom complet : " WS-NOM-COMPLET.

DELIMITED BY : Les Options

DELIMITED BY SIZE

→ Copie tout le champ (jusqu'à la fin)

DELIMITED BY SPACES

→ Copie jusqu'au premier espace

DELIMITED BY "caractère"

→ Copie jusqu'au caractère spécifié

DELIMITED BY variable

→ Copie jusqu'à la valeur de la variable

Exemples Détaillés :

01 WS-TEXTE1 PIC X(20) VALUE "Hello World ".

01 WS-TEXTE2 PIC X(20) VALUE "COBOL Programming".

01 WS-RESULTAT PIC X(50).

PROCEDURE DIVISION.

*--- DELIMITED BY SIZE : Tout copier ---

STRING

WS-TEXTE1 DELIMITED BY SIZE

WS-TEXTE2 DELIMITED BY SIZE

INTO WS-RESULTAT

END-STRING.

*> Résultat : "Hello World COBOL Programming"

*--- DELIMITED BY SPACES : Jusqu'au premier espace ---

STRING

WS-TEXTE1 DELIMITED BY SPACES

" - " DELIMITED BY SIZE

WS-TEXTE2 DELIMITED BY SPACES

INTO WS-RESULTAT

END-STRING.

*> Résultat : "Hello - COBOL"

💡 Construire une Adresse Complète :

01 ADR-NUMERO PIC 9(3) VALUE 45.

01 ADR-RUE PIC X(40) VALUE "rue de la Paix".

01 ADR-CP PIC 9(5) VALUE 75001.

01 ADR-VILLE PIC X(30) VALUE "PARIS".

01 ADR-COMPLETE PIC X(100).

PROCEDURE DIVISION.

STRING

ADR-NUMERO DELIMITED BY SIZE

" " DELIMITED BY SIZE

ADR-RUE DELIMITED BY SIZE

", " DELIMITED BY SIZE

ADR-CP DELIMITED BY SIZE

" " DELIMITED BY SIZE

ADR-VILLE DELIMITED BY SIZE

INTO ADR-COMPLETE

END-STRING.

DISPLAY "Adresse : " ADR-COMPLETE.

*> Résultat : "45 rue de la Paix, 75001 PARIS"

WITH POINTER : Contrôle de Position

01 WS-PRENOM PIC X(10) VALUE "Jean".

01 WS-NOM PIC X(15) VALUE "Dupont".

01 WS-RESULTAT PIC X(50).

01 WS-POSITION PIC 99 VALUE 1.

PROCEDURE DIVISION.

STRING

"M. " DELIMITED BY SIZE

WS-PRENOM DELIMITED BY SPACES

" " DELIMITED BY SIZE

WS-NOM DELIMITED BY SPACES

```
    INTO WS-RESULTAT
    WITH POINTER WS-POSITION
END-STRING.
```

```
    DISPLAY "Résultat : " WS-RESULTAT.
```

```
    DISPLAY "Position finale : " WS-POSITION.
```

*> Position = position après le dernier caractère copié + 1

ON OVERFLOW : Gestion du Débordement

```
01 WS-SOURCE1    PIC X(40) VALUE "Texte très long...".
01 WS-SOURCE2    PIC X(40) VALUE "Encore plus long...".
01 WS-DEST       PIC X(50). *> Trop petit!
```

PROCEDURE DIVISION.

STRING

WS-SOURCE1 DELIMITED BY SIZE

WS-SOURCE2 DELIMITED BY SIZE

INTO WS-DEST

ON OVERFLOW

DISPLAY "ATTENTION : Texte tronqué!"

END-STRING.

9.3 UNSTRING : Découpage

Découper une chaîne en plusieurs parties :

UNSTRING source

DELIMITED BY délimiteur1 [OR délimiteur2 ...]

INTO dest1 [DELIMITER IN delim1] [COUNT IN count1]

dest2 [DELIMITER IN delim2] [COUNT IN count2]

...

[WITH POINTER pointeur]

[TALLYING IN compteur]

[ON OVERFLOW instructions]

END-UNSTRING

Exemple Simple : Découper un Nom Complet

```
01 WS-NOM-COMPLET PIC X(50) VALUE "Jean Pierre Dupont".
```

```
01 WS-PRENOM1     PIC X(20).
```

```
01 WS-PRENOM2     PIC X(20).
```

```
01 WS-NOM         PIC X(20).
```

PROCEDURE DIVISION.

UNSTRING WS-NOM-COMPLET

DELIMITED BY SPACES

INTO WS-PRENOM1 WS-PRENOM2 WS-NOM

END-UNSTRING.

```
DISPLAY "Prénom 1 : " WS-PRENOM1. *> "Jean"
```

```
DISPLAY "Prénom 2 : " WS-PRENOM2. *> "Pierre"
```

```
DISPLAY "Nom      : " WS-NOM.    *> "Dupont"
```

💡 Parser une Adresse Email :

```
01 WS-EMAIL       PIC X(50) VALUE "jean.dupont@exemple.fr".
```

```
01 WS-USER        PIC X(30).
```

```
01 WS-DOMAINE     PIC X(30).
```

```
01 WS-SEPARATEUR  PIC X.
```


PROCEDURE DIVISION.

UNSTRING WS-EMAIL

DELIMITED BY "@"

INTO WS-USER

DELIMITER IN WS-SEPARATEUR

WS-DOMAIN

END-UNSTRING.

DISPLAY "Utilisateur : " WS-USER. *> "jean.dupont"

DISPLAY "Séparateur : " WS-SEPARATEUR. *> "@"

DISPLAY "Domaine : " WS-DOMAIN. *> "exemple.fr"

 Parser une Date (JJ/MM/AAAA) :

01 WS-DATE-TEXTE PIC X(10) VALUE "15/11/2024".

01 WS-JOUR PIC 99.

01 WS-MOIS PIC 99.

01 WS-ANNEE PIC 9(4).

PROCEDURE DIVISION.

UNSTRING WS-DATE-TEXTE

DELIMITED BY "/"

INTO WS-JOUR WS-MOIS WS-ANNEE

END-UNSTRING.

DISPLAY "Jour : " WS-JOUR. *> 15

DISPLAY "Mois : " WS-MOIS. *> 11

DISPLAY "Année : " WS-ANNEE. *> 2024

 Parser un CSV (plusieurs délimiteurs) :

```
01 WS-CSV-LINE    PIC X(100)
    VALUE "Dupont;Jean;45;Informatique;75000".
01 WS-NOM         PIC X(30).
01 WS-PRENOM      PIC X(30).
01 WS-AGE         PIC 99.
01 WS-SERVICE     PIC X(30).
01 WS-SALAIRE     PIC 9(7).
01 WS-NB-CHAMPS   PIC 9 VALUE ZERO.
```

PROCEDURE DIVISION.

UNSTRING WS-CSV-LINE

DELIMITED BY "; " OR " , " * > Accepte ; ou ,

INTO WS-NOM

WS-PRENOM

WS-AGE

WS-SERVICE

WS-SALAIRE

TALLYING IN WS-NB-CHAMPS

END-UNSTRING.

DISPLAY "Nom : " WS-NOM.

DISPLAY "Prénom : " WS-PRENOM.

DISPLAY "Âge : " WS-AGE.

DISPLAY "Service : " WS-SERVICE.

DISPLAY "Salaire : " WS-SALAIRE.

DISPLAY "Champs traités : " WS-NB-CHAMPS. *> 5

9.4 INSPECT : Recherche et Remplacement

INSPECT permet de :

- Compter des caractères
- Remplacer des caractères
- Convertir des caractères

 INSPECT TALLYING (Compter)

INSPECT chaîne

TALLYING compteur FOR [ALL|LEADING|CHARACTERS]

caractère-ou-chaîne

[BEFORE|AFTER INITIAL délimiteur]

Exemples :

```
01 WS-TEXTE      PIC X(50) VALUE "HELLO WORLD HELLO".
```

```
01 WS-COMPTEUR   PIC 99 VALUE ZERO.
```

PROCEDURE DIVISION.

*--- Compter toutes les occurrences de 'L' ---

INSPECT WS-TEXTE

TALLYING WS-COMPTEUR FOR ALL "L".

*> WS-COMPTEUR = 5

DISPLAY "Nombre de 'L' : " WS-COMPTEUR.

*--- Compter 'HELLO' ---

MOVE ZERO TO WS-COMPTEUR.

INSPECT WS-TEXTE

TALLYING WS-COMPTEUR FOR ALL "HELLO".

*> WS-COMPTEUR = 2

DISPLAY "Nombre de 'HELLO' : " WS-COMPTEUR.

*--- Compter les espaces ---

MOVE ZERO TO WS-COMPTEUR.

INSPECT WS-TEXTE

TALLYING WS-COMPTEUR FOR ALL SPACES.

*> WS-COMPTEUR = 2

DISPLAY "Nombre d'espaces : " WS-COMPTEUR.

LEADING : Compter en début uniquement

01 WS-NOMBRE PIC X(10) VALUE "00012345 ".

01 WS-ZEROS PIC 99 VALUE ZERO.

01 WS-ESPACES PIC 99 VALUE ZERO.

PROCEDURE DIVISION.

*--- Compter les zéros initiaux ---

INSPECT WS-NOMBRE

TALLYING WS-ZEROS FOR LEADING "0".

*> WS-ZEROS = 3

*--- Compter les espaces finaux (depuis la fin) ---

INSPECT WS-NOMBRE

TALLYING WS-ESPACES FOR LEADING SPACES

AFTER INITIAL "5".

*> WS-ESPACES = 2

 INSPECT REPLACING (Remplacer)

INSPECT chaîne

REPLACING [ALL|LEADING|FIRST]

caractère1 BY caractère2

[BEFORE|AFTER INITIAL délimiteur]

Exemples :

01 WS-TEXTE PIC X(50).

PROCEDURE DIVISION.

*--- Remplacer tous les espaces par des tirets ---

MOVE "HELLO WORLD TEST" TO WS-TEXTE.

INSPECT WS-TEXTE REPLACING ALL SPACES BY "-".

*> Résultat : "HELLO-WORLD-TEST"

DISPLAY WS-TEXTE.

*--- Remplacer les zéros initiaux par des espaces ---

MOVE "00012345" TO WS-TEXTE.

INSPECT WS-TEXTE REPLACING LEADING "0" BY SPACE.

*> Résultat : " 12345"

DISPLAY "[" WS-TEXTE "]".

*--- Remplacer la première occurrence ---

MOVE "HELLO HELLO HELLO" TO WS-TEXTE.

INSPECT WS-TEXTE REPLACING FIRST "HELLO" BY "BONJR".

*> Résultat : "BONJR HELLO HELLO"

DISPLAY WS-TEXTE.

💡 Masquer un Numéro de Carte Bancaire :

01 WS-CARTE PIC X(19) VALUE "1234-5678-9012-3456".

01 WS-CARTE-MASQUE PIC X(19).

PROCEDURE DIVISION.

 MOVE WS-CARTE TO WS-CARTE-MASQUE.

*--- Masquer les 12 premiers chiffres ---

 INSPECT WS-CARTE-MASQUE

 REPLACING ALL "1" BY "*"

 ALL "2" BY "*"

 ALL "3" BY "*"

 ALL "4" BY "*"

 ALL "5" BY "*"

 ALL "6" BY "*"

 ALL "7" BY "*"

 ALL "8" BY "*"

 ALL "9" BY "*"

 ALL "0" BY "*"

 BEFORE INITIAL "-3456".

 DISPLAY "Original : " WS-CARTE.

 DISPLAY "Masqué : " WS-CARTE-MASQUE.

*> Résultat : "****_****_****-3456"

 INSPECT CONVERTING (Convertir)

Convertir plusieurs caractères en une seule instruction :

INSPECT chaîne

CONVERTING "chaîne-source" TO "chaîne-destination"

Exemples :

```
01 WS-TEXTE      PIC X(50).
```

PROCEDURE DIVISION.

*--- Convertir minuscules en majuscules (simple) ---

```
MOVE "hello world" TO WS-TEXTE.
```

```
INSPECT WS-TEXTE
```

```
    CONVERTING "abcdefghijklmnopqrstuvwxy"
```

```
        TO "ABCDEFGHIJKLMNOPQRSTUVWXYZ".
```

*> Résultat : "HELLO WORLD"

```
DISPLAY WS-TEXTE.
```

*--- ROT13 (chiffrement basique) ---

```
MOVE "HELLO" TO WS-TEXTE.
```

```
INSPECT WS-TEXTE
```

```
    CONVERTING "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

```
        TO "NOPQRSTUVWXYZABCDEFGHIJKLM".
```

*> Résultat : "URYYB"

```
DISPLAY "ROT13 : " WS-TEXTE.
```

⚠ Les deux chaînes doivent avoir la MÊME LONGUEUR

✅ Correct :

```
INSPECT WS-TEXTE CONVERTING "abc" TO "ABC".
```

❌ Erreur :

```
INSPECT WS-TEXTE CONVERTING "abc" TO "AB". *> Longueurs différentes!
```

9.5 Fonctions de Chaînes

 **LENGTH** : Longueur

FUNCTION LENGTH(chaîne) → Retourne la longueur

Exemples :

```
01 WS-TEXTE      PIC X(50) VALUE "HELLO".
```

```
01 WS-LONGUEUR   PIC 99.
```


PROCEDURE DIVISION.

```
    COMPUTE WS-LONGUEUR = FUNCTION LENGTH(WS-TEXTE).
```

*> WS-LONGUEUR = 50 (taille du champ, pas du contenu!)

```
    COMPUTE WS-LONGUEUR = FUNCTION LENGTH("HELLO").
```

*> WS-LONGUEUR = 5 (taille de la chaîne littérale)

 Pour avoir la longueur "utile" (sans espaces de fin) :

```
01 WS-TEXTE      PIC X(50) VALUE "HELLO".
```

```
01 WS-TEXTE-TRIM PIC X(50).
```

```
01 WS-LONGUEUR-UTILE PIC 99.
```

PROCEDURE DIVISION.

```
    MOVE FUNCTION TRIM(WS-TEXTE) TO WS-TEXTE-TRIM.
```

```
    COMPUTE WS-LONGUEUR-UTILE =
```

```
        FUNCTION LENGTH(FUNCTION TRIM(WS-TEXTE)).
```

*> WS-LONGUEUR-UTILE = 5

 **Extraction de Sous-Chaînes**

FUNCTION SUBSTRING(chaîne, position, longueur)

*> Ou référence modification :

chaîne(position:longueur)

Exemples :

```
01 WS-TEXTE      PIC X(20) VALUE "HELLO WORLD".
```

```
01 WS-EXTRAIT    PIC X(10).
```

PROCEDURE DIVISION.

*--- Méthode 1 : FUNCTION ---

```
MOVE FUNCTION SUBSTRING(WS-TEXTE, 1, 5) TO WS-EXTRAIT.
```

```
*> WS-EXTRAIT = "HELLO"
```

*--- Méthode 2 : Référence Modification (préférée) ---

```
MOVE WS-TEXTE(1:5) TO WS-EXTRAIT.
```

```
*> WS-EXTRAIT = "HELLO"
```

```
MOVE WS-TEXTE(7:5) TO WS-EXTRAIT.
```

```
*> WS-EXTRAIT = "WORLD"
```

*--- Affichage direct ---

```
DISPLAY "Mot 1 : " WS-TEXTE(1:5).  *> "HELLO"
```

```
DISPLAY "Mot 2 : " WS-TEXTE(7:5).  *> "WORLD"
```

💡 Parser une Date :

```
01 WS-DATE      PIC X(10) VALUE "2024-11-15".
```

```
01 WS-ANNEE     PIC 9(4).
```

```
01 WS-MOIS      PIC 99.
```

```
01 WS-JOUR      PIC 99.
```

PROCEDURE DIVISION.

MOVE WS-DATE(1:4) TO WS-ANNEE. * > 2024

MOVE WS-DATE(6:2) TO WS-MOIS. * > 11

MOVE WS-DATE(9:2) TO WS-JOUR. * > 15

DISPLAY "Année : " WS-ANNEE.

DISPLAY "Mois : " WS-MOIS.

DISPLAY "Jour : " WS-JOUR.

 TRIM : Supprimer les Espaces

FUNCTION TRIM(chaîne [LEADING|TRAILING])

Exemples :

01 WS-TEXTE PIC X(20) VALUE " HELLO ".

01 WS-RESULTAT PIC X(20).

PROCEDURE DIVISION.

*--- Supprimer espaces des deux côtés (défaut) ---

MOVE FUNCTION TRIM(WS-TEXTE) TO WS-RESULTAT.

* > WS-RESULTAT = "HELLO"

*--- Supprimer espaces de début uniquement ---

MOVE FUNCTION TRIM(WS-TEXTE LEADING) TO WS-RESULTAT.

* > WS-RESULTAT = "HELLO "

*--- Supprimer espaces de fin uniquement ---

MOVE FUNCTION TRIM(WS-TEXTE TRAILING) TO WS-RESULTAT.

* > WS-RESULTAT = " HELLO"

UPPER-CASE / LOWER-CASE

FUNCTION UPPER-CASE(chaine) → Majuscules

FUNCTION LOWER-CASE(chaine) → Minuscules

Exemples :

```
01 WS-TEXTE      PIC X(30) VALUE "Hello World".
```

```
01 WS-MAJUSCULES PIC X(30).
```

```
01 WS-MINUSCULES PIC X(30).
```

PROCEDURE DIVISION.

```
MOVE FUNCTION UPPER-CASE(WS-TEXTE) TO WS-MAJUSCULES.
```

```
*> WS-MAJUSCULES = "HELLO WORLD"
```

```
MOVE FUNCTION LOWER-CASE(WS-TEXTE) TO WS-MINUSCULES.
```

```
*> WS-MINUSCULES = "hello world"
```

```
DISPLAY "Original   : " WS-TEXTE.
```

```
DISPLAY "Majuscules : " WS-MAJUSCULES.
```

```
DISPLAY "Minuscules : " WS-MINUSCULES.
```

REVERSE : Inverser

FUNCTION REVERSE(chaine)

Exemples :

```
01 WS-TEXTE      PIC X(20) VALUE "COBOL".
```

```
01 WS-INVERSE     PIC X(20).
```

PROCEDURE DIVISION.

```
MOVE FUNCTION REVERSE(WS-TEXTE) TO WS-INVERSE.
```







*> WS-INVERSE = "LOBOC"

DISPLAY "Original : " WS-TEXTE.

DISPLAY "Inversé : " WS-INVERSE.

(La suite continue avec les structures conditionnelles, les boucles, et des exemples complets...)

 Points Couverts dans cette section :

-  Opérations arithmétiques complètes (ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPUTE)
-  Fonctions mathématiques (ABS, MAX, MIN, SQRT, etc.)
-  Gestion d'erreurs (ON SIZE ERROR, ROUNDED)
-  Manipulation de chaînes (STRING, UNSTRING, INSPECT)
-  Fonctions de chaînes (TRIM, LENGTH, UPPER-CASE, etc.)
-  Exemples pratiques complets (prêt bancaire, parsing CSV, etc.)

 Continue vers la suite avec les structures de contrôle et les fichiers !