



**Curso Profissional: Programador/a de Informática**  
**PSD – 10.º ano: UFCD 0807 - Programação em Cobol – fundamentos**

FICHA DE TRABALHO N.º 3

**Ano letivo 21/22**

## Declarações/instruções

### Comando/verbo INICIALIZE

Este comando proporciona o ajustamento de categorias de dados a valores predeterminados, como por exemplo, ajustar todos os itens alfanuméricos em espaços em branco.

#### SINTAXE:

INITIALIZE [VARIÁVEL] REPLACING [ALPHABETIC, ALPHANUMERIC, NUMERIC, ALPHANUMERIC-EDITED, NUMERIC-EDITED, DATA] BY [IDENTIFICADOR OU VALOR]

#### Exemplo1:

<p>WORKING-STORAGE SECTION. 01 CLIENTE.</p> <p>05 CODIGO PIC 9(004). 05 NOME PIC X(030). 05 NASCIMENTO PIC 9(008). 05 TELEFONE PIC X(018).</p> <p>PROCEDURE DIVISION. INITIALIZE CLIENTE REPLACING NUMERIC BY ZEROS ALPHANUMERIC BY SPACES.</p>	<p>INICIALIZA CLIENTE TROCANDO O CONTEÚDO DAS VARIÁVEIS NUMÉRICAS POR 0 E DAS ALFANUMÉRICAS POR ESPAÇOS</p>
---	---

#### Exemplo2:

<p>WORKING-STORAGE SECTION. 01 ws-registo.</p> <p>05 ws-numero PIC 9(004). 05 ws-nome PIC X(030). 05 ws-profissao PIC X(025).</p> <p>PROCEDURE DIVISION. INITIALIZE ws-registo. *&gt; se usasse... INITIALIZE ws-registo REPLACING NUMERIC DATA BY 3 ALPHANUMERIC DATA BY "X"</p>	<p>INICIALIZA ws-numero com zeros e ws-nome e ws-profissao com espaços O CONTEÚDO DAS VARIÁVEIS NUMÉRICAS SÃO INICIALIZADAS A 3 E AS ALFANUMÉRICAS POR "X"</p>
---	--

### Comando/verbo GO TO

Utilizado para desvio no fluxo do programa. Para ser utilizado Para que possa ser utilizado é necessária a definição de parágrafos, que são rótulos ou endereços que dividem a aplicação em blocos.

#### SINTAXE:

GO [TO] <PARÁGRAFO> [DEPENDING ON <VARIÁVEL>].

### Exemplo:

<b>WORKING-STORAGE SECTION.</b> <b>77 NUMERO PIC 9(004).</b> <b>PROCEDURE DIVISION.</b> <b>INICIO.</b> <b>DISPLAY "INFORME O NUMERO (ZERO ABANDONA):</b> <b>ACCEPT NUMERO</b> <b>IF NUMERO = 0</b> <b>GO TO FIM</b> <b>END-IF</b> <b>DISPLAY "VOCE DIGITOU O NUMERO: " NUMERO</b> <b>GO TO INICIO.</b> <b>FIM.</b> <b>GOBACK.</b>	<b>INICIO.</b> <b>MOSTRA</b> <b>"INFORME O NUMERO (ZERO ABANDONA): " RECEBE</b> <b>NUMERO</b> <b>SE NUMERO = 0 VAI PARA FIM SE-FIM</b> <b>MOSTRA MENSAGEM</b> <b>"VOCE DIGITOU O NUMERO: " NUMERO VAI PARA</b> <b>INICIO.</b> <b>FIM.</b> <b>ENCERRA.</b>
---	--

## Comando/verbo **PERFORM**

Este verbo/comando é uma outra forma de alterar o controlo de fluxo num programa escrito em COBOL. Tem dois propósitos principais:

- Transferir o controlo da execução do programa para um bloco específico de código;
- Executar um bloco de código interactivamente (a ver mais tarde).

Exemplo (de utilização): A variável IterNum é inicializada com o valor 5, e o verbo **PERFORM** executa o parágrafo DisplayGreeting 5 vezes.

<b>IDENTIFICATION DIVISION.</b> <b>PROGRAM-ID. CobolGreeting.</b> <b>*&gt;Program to display COBOL greetings</b> <b>DATA DIVISION.</b> <b>WORKING-STORAGE SECTION.</b> <b>01 IterNum PIC 9 VALUE 5.</b> <b>PROCEDURE DIVISION.</b> <b>BeginProgram.</b> <b>PERFORM DisplayGreeting IterNum TIMES.</b> <b>STOP RUN.</b> <b>DisplayGreeting.</b> <b>DISPLAY "Greetings from COBOL".</b>	Greetings from COBOL Greetings from COBOL Greetings from COBOL Greetings from COBOL Greetings from COBOL
--	--

## Aritmética em COBOL

A maioria das linguagens de programação procedimentais executam cálculos atribuindo o resultado de uma expressão aritmética (ou função) a uma variável. Em COBOL, o verbo **COMPUTE** é usado para avaliar expressões aritméticas, mas também existem comandos específicos para adicionar (**ADD**), subtrair (**SUBTRACT**), multiplicar (**MULTIPLY**) e dividir (**DIVIDE**).

O tamanho máximo de cada operando é de 18 dígitos decimais.

À excepção de **COMPUTE**, **DIVIDE** com o **REMAINDER** (resto da divisão inteira) e alguns formatos exóticos para adicionar e subtrair, a maioria dos verbos de aritmética de COBOL estão em conformidade com a metalinguagem do modelo mostrada na Figura seguinte:

$$\text{VERB Operand1\#il} \left\{ \begin{array}{c} \text{TO} \\ \text{FROM} \\ \text{BY} \\ \text{INTO} \end{array} \right\} \left\{ \begin{array}{l} \{ \text{OperandResult\#i} \text{ [ROUNDED]} \} \dots \\ \text{Operand\#il GIVING} \{ \text{Result\#i} \text{ [ROUNDED]} \} \dots \end{array} \right\} \\ \left[ \left\{ \begin{array}{c} \text{ON SIZE ERROR} \\ \text{[NOT ON SIZE ERROR]} \end{array} \right\} \text{StatementBlock END - VERB} \right]$$

## Operadores

Existem 5 operadores aritméticos que podem ser usados nas expressões aritméticas:

Precedência	Símbolo	Significado
1	**	Potenciação
2	*	Produto
	/	Divisão
3	+	Soma
	-	Subtração

### Regras:

- Todo operador aritmético deve ser precedido e seguido por, no mínimo um espaço;
- As expressões dentro de parênteses, são executadas primeiro;
- Quando existirem diversos parênteses, os mais internos serão executados primeiro;
- Quando não há parênteses, as operações são executadas de acordo com a tabela acima;
- Se a sequência de operações do mesmo nível hierárquico, não for esclarecida pelos parênteses as operações serão efetuadas da esquerda para a direita.

### Exemplo:

- A / B \* C será interpretada como se tivesse sido escrita da seguinte forma: (A / B) \* C
- A + B / C + D \*\* E \* F + G é interpretada como: A + (B / C) + ( ( D \*\* E ) \* F ) + G

## Comando/verbo COMPUTE

O comando COMPUTE atribui o resultado de uma expressão aritmética a um item.

**COMPUTE** calcula a expressão aritmética à direita do sinal de igual e atribui o seu resultado ao item/variável à esquerda do sinal de igual.

**Atenção:** Nunca devemos usar o verbo COMPUTE para atribuir valores a variáveis, para isso devemos usar o verbo MOVE

**COMPUTE** {Result#i[ ROUNDED ]}... = Arithmetic Expression

$\left[ \begin{array}{l} \text{ON SIZE ERROR} \\ \text{NOT ON SIZE ERROR} \end{array} \right\} \text{StatementBlock } \underline{\text{END - COMPUTE}}$

### Exemplos:

COMPUTE SOMA = 5.

COMPUTE VALOR = QUANTIDADE.

COMPUTE SALARIO = (SALARIO \* 1.5) + ADIC.

COMPUTE TAXA = 5.6 \* CAPITAL.

COMPUTE Result = 90 - 7 \* 3 + 50 / 2

01 Result PIC 9(4) VALUE 3333.

Antes	3333
Depois	<b>0094</b>

Isto é equivalente a:

COMPUTE Result = 90 - (7 \* 3) + (50 / 2)

01 Result PIC 9(4) VALUE 3333.

Antes	3333
Depois	<b>0094</b>

COMPUTE Euro ROUNDED = Dolar / TaxaDeCambio

01 Euro PIC 9(5)V99 VALUE 3425.15.

01 Dolar PIC 9(5)V99 VALUE 1234.75.

01 TaxaDeCambio PIC 9V9(4) VALUE 1.3017.

	Euro	Dolar	TaxaDeCambio
Antes	3425.15	1234.75	1.3017
Depois	<b>0948.57</b>	1234.75	1.3017

### Cláusula ROUNDED

O arredondamento de valores é bastante comum quando um processo envolve cálculos. Caso ele esteja presente na instrução de cálculo ou na fórmula, o arredondamento será automático, e respeitará os limites definidos nas variáveis envolvidas.

#### Exemplo:

**77 Valor PIC 9(003)v99.**

O cálculo encontrou o valor 23,458 (3 decimais) e o conteúdo de VALOR será 23,46 se utilizarmos a cláusula ROUNDED. Se omitirmos o arredondamento, o conteúdo de VALOR será 23,45.

O que significa que quando no campo recetor foi reservado um certo número de posições decimais e esse campo receber o resultado de uma operação cujo número de

posições decimais excede àquele, então os dígitos mais à direita do resultado serão perdidos.

Para se eliminar tal truncamento, utiliza-se a cláusula **ROUNDED**, que irá arredondar o dígito menos significativo.

Exemplos de aplicação da opção ROUNDED			
Valor calculado	Picture	Resultado truncado	Resultado arredondado
342,736	999v99	342,73	342,74
342,734	999V99	342,73	342,73
342,534	999	342	343
342,464	999	342	342
5,958	9V99	5,95	5,96
12,821565	99V9(5)	12,82156	12,82157
55,55	9(002)V9	55,5	55,6

## Cláusula ON SIZE ERROR

A cláusula ON SIZE ERROR permite ao programador controlar situações de erro envolvendo cálculos (e outras também), como tentativa de divisão por zero e estouro da variável de resultado, que ocorre quando a variável de destino não suporta o valor encontrado.

Por exemplo, se a variável determinada para receber um determinado resultado tiver 5 posições (PIC9(5)) e se esse resultado for 123456, não irá haver lugar ao seu armazenamento!!!

### Exemplo:

<b>77 A PIC 9(002).</b> <b>77 B PIC 9(002) VALUE 0.</b> <b>77 C PIC 9(002).</b> <b>PROCEDURE DIVISION.</b> <b>SET A to 6.</b> <b>DIVIDE A BY B GIVING C ON SIZE ERROR</b> <b>DISPLAY "ERRO NA DIVISÃO"</b> <b>END-DIVIDE</b>	<b>ATRIBUI O VALOR 6 À VARIÁVEL A ( O MESMO que VALUE 6)</b> <b>DIVIDE A POR B COLOCANDO O RESULTADO EM C SE DER ERRO (DIVISÃO POR 0) MOSTRA UMA MENSAGEM "ERRO NA DIVISÃO"</b> <b>DIVIDE-FIM.</b>
<b>77 A PIC 9(002) VALUE 6.</b> <b>77 B PIC 9(002) VALUE 25.</b> <b>77 C PIC 9(002).</b> <b>PROCEDURE DIVISION.</b> <b>COMPUTE C = A + B ON SIZE ERROR</b> <b>DISPLAY "ESTOURO DE VARIÁVEL !!"</b> <b>NOT ON SIZE ERROR DISPLAY "CALCULO OK "</b> <b>END-COMPUTE</b>	<b>CALCULA C = A + B</b> <b>EM CASO DE ERRO (ESTOURO) MOSTRA A MENSAGEM "ESTOURO DE VARIÁVEL"</b> <b>EM CASO DE NÃO HAVER ERRO MOSTRA A MENSAGEM "CALCULO OK"</b> <b>CALCULA-FIM</b>

## O verbo ADD

Verbo usado para adição de valores (o que pode ser feito com o verbo COMPUTE mas de forma menos simples)

$$\text{ADD Operand\#il...} \left\{ \begin{array}{l} \text{TO } \{\text{OperandResult\#i [ROUNDED]}\}... \\ \text{[TO]Operand\#il GIVING \{Result\#i [ROUNDED]\}...} \end{array} \right\}$$

[ON SIZE ERROR StatementBlock END - ADD]

Existem dois formatos para o comando ADD:

### FORMATO 1

**ADD** <operando1>...<operandoN> **TO** <operando-resul1> [ROUNDED]... <operando-result1> [ROUNDED]  
[on size error BlocoDeDeclarações end - ADD]

A soma dos operandos operandos1, ...operandoN é adicionada aos valores já existentes nos operando-resultado, operando-result1,..., operando-resultN

Os operandos à direita da cláusula TO só podem ser variáveis!

### FORMATO 2

**ADD** <operando1> ...<operandoN> [**TO**] <operando-resul1> [ROUNDED]... <operando-result1> [ROUNDED]  
**GIVING** <variavel1> [ROUNDED]... <variavelN> [ROUNDED]

A soma dos operandos operandos1, ...operandoN é adicionada aos valores já existentes nos operando-resultado, operando-result1,..., operando-resultN e colocada na(s) variáveis variavel1...variavelN após terem sido inicializadas a zero.

A Cláusula TO é opcional pelo que podemos ter:

**ADD** <operando1> ...<operandoN> **GIVING** <variavel1> [ROUNDED]... <variavelN> [ROUNDED]

### Exemplos:

1. Supondo que Num1 tem o valor 2, Num2 tem o valor 4, Num3 e Num4 têm o valor 1 e que Result tem o valor 5 (cada exemplo é independente):

ADD Num1, Num2 TO Num3, Num4.	São adicionados os valores dos itens Num1 e Num2 (2+4) e atribuídos a Num3 e Num4 que ficam com o valor 7
ADD Num1, 2, Num3 TO Num4 GIVING Result.	São adicionados os valores dos itens Num1, 2 e Num3 (2+2+1) a Num4 e atribuídos a Result que fica com o valor 6
ADD Num1, Num2, Num3 TO Num4 GIVING Result.	São adicionados os valores dos itens Num1, Num2 e Num3 (2+4+1) a Num4 que fica inalterado e o Result assume o valor 8
ADD Num1, Num3 GIVING Result.	São adicionados os valores dos itens Num1 e Num3 (2+1) e o item Result assume o valor 3

2.

01 dinheiro PIC 9(3) VALUE 364.  
01 Total PIC 9(4) VALUE 1000.  
ADD dinheiro TO Total.

	dinheiro	Total
Antes	364	1000
Depois	364	1364

3.

01 dinheiro PIC 9(3) VALUE 364.  
01 Total PIC 9(4) VALUE 1000.  
ADD dinheiro, 20 TO Total.

	dinheiro	Total
Antes	364	1000
Depois	364	1384

4.

01 dinheiro PIC 9(3) VALUE 364.  
01 Total PIC 9(4) VALUE 1000.  
01 cheques PIC 9(4) VALUE 1445.  
ADD dinheiro, cheques TO Total.

	dinheiro	cheques	Total
Antes	364	1445	1000
Depois	364	1445	2809

## O verbo SUBTRACT

Verbo usado para a subtração de valores.

SUBTRACT Operand#il...{FROM} { {OperandResult#i [ROUNDED]}...  
{ {Operand#il GIVING Result#i [ROUNDED]}... }  
[ON SIZE ERROR StatementBlock END-SUBTRACT]

Existem dois formatos para o comando SUBTRACT:

### FORMATO 1

**SUBTRACT** <operando1>...<operandoN> **FROM** <operando-resul1> [ROUNDED]... <operando-resul1> [ROUNDED]  
[on size error BlocoDeDeclarações end - SUBTRACT]

A soma dos operandos operandos1, ...operandoN é SUBTRAÍDA dos valores já existentes nos operando-resultado, operando-result1,..., operando-resultN

Os operandos à direita da cláusula FROM só podem ser variáveis!

### FORMATO 2

**SUBTRACT** <operando1> ...<operandoN> **FROM** <operando-i1> [ROUNDED]... <operando-iN> [ROUNDED]  
**GIVING** <variavel1> [ROUNDED]... <variavelN> [ROUNDED]

A soma dos operandos operandos1, ...operandoN é SUBTRAÍDA dos valores já existentes nos operando-resultado, operando-result1,..., operando-resultN e colocada na(s) variáveis variavel1...variavelN após terem sido inicializadas a zero.

Os valores dos operandos operando1...operandoN não são alterados por esta instrução.

## Exemplos:

1. Supondo que Num1 tem o valor 2, Num2 tem o valor 4, Num3 e Num4 têm o valor 1, Result1 tem o valor 5 e Result2 tem o valor -5 (cada exemplo é independente):

SUBTRACT Num1, Num2 FROM Result1.	São adicionados os valores dos itens Num1 e Num2 (2+4) e o resultado é subtraído ao valor de Result1, que fica com o valor -1
SUBTRACT Num1, Num2 FROM Num3 GIVING Result1.	São adicionados os valores dos itens Num1 e Num2 (2+4) e o resultado é subtraído ao valor de Num3 (que fica inalterado) e o resultado assume o valor -5
SUBTRACT Num1, Num2 FROM Result1, Result2.	São adicionados os valores dos itens Num1 e Num2 (2+4) e o resultado é subtraído ao valor de Result1, que fica com o valor -1, e subtraído ao valor de Result2, que fica com o valor -11

2.

01 Num1 PIC 9(4) VALUE 364.  
 01 Num2 PIC 9(4) VALUE 1000.  
 01 Num3 PIC 9(4) VALUE 5555.  
 01 Resultado PIC 9(4) VALUE 1445.  
 SUBTRACT Num1, Num2 FROM Num3 GIVING Resultado.

	Num1	Num2	Num3	Resultado
<b>Antes</b>	364	1000	5555	1445
<b>Depois</b>	364	1000	5555	<b>4191</b>

3.

SUBTRACT Num1, Num2 FROM Resultado1, Resultado2.  
 01 Num1 PIC 9(4) VALUE 364.  
 01 Num2 PIC 9(4) VALUE 1000.  
 01 Resultado1 PIC 9(4) VALUE 5555.  
 01 Resultado2 PIC 9(4) VALUE 1445.

	Num1	Num2	Resultado1	Resultado2
<b>Antes</b>	364	1000	5555	1445
<b>Depois</b>	364	1000	<b>4191</b>	<b>0081</b>

4.

SUBTRACT Taxa, PR, Pensao, Cobranca FROM Pagamento GIVING NetPag.  
 01 Pagamento PIC 9(4)V99 VALUE 6350.75.  
 01 Taxa PIC 9(4)V99 VALUE 2333.25.  
 01 PR PIC 9(4)V99 VALUE 1085.45.  
 01 Pensao PIC 9(4)V99 VALUE 1135.74.  
 01 Cobranca PIC 9(3)V99 VALUE 170.50.  
 01 NetPag PIC 9(4)V99 VALUE ZEROS.

	Pagamento	Taxa	PR	Pensao	Cobranca	NetPag
<b>Antes</b>	6350.75	2333.25	1085.45	1135.74	170.50	0000.00
<b>Depois</b>	6350.75	2333.25	1085.45	1135.74	170.50	<b>1625.81</b>



5.

<b>WORKING-STORAGE SECTION.</b> <b>77 A PIC 9(002) VALUE 0.</b> <b>77 B PIC 9(002) VALUE 0.</b> <b>77 C PIC -Z9.</b> <b>PROCEDURE</b> <b>DIVISION.</b> <b>DISPLAY "Insira A: "</b> <b>ACCEPT A</b> <b>DISPLAY "Insira B: "</b> <b>ACCEPT B</b> <b>SUBTRACT B FROM A GIVING C ON SIZE ERROR</b> <b>DISPLAY "IMPOSSIVEL CALCULAR, VARIÁVEL PEQUENA"</b> <b>GOBACK</b> <b>END-SUBTRACT</b> <b>DISPLAY " A - B = " C</b> <b>ACCEPT OMITTED</b> <b>GOBACK.</b>	<b>MOSTRA "Insira A: "</b> <b>RECEBE A</b> <b>MOSTRA "Insira B: "</b> <b>RECEBE B</b>  <b>SUBTRAI B a A OBTENDO C</b> <b>EM CASO DE ERRO</b> <b>MOSTRE A MENSAGEM "IMPOSSIVEL CALCULAR, VARIÁVEL PEQUENA !"</b> <b>ENCERRA</b> <b>SUBTRAI-FIM</b> <b>MOSTRE "A - B = " C</b> <b>AGUARDA UMA TECLA</b> <b>ENCERRA.</b>
---	---

## O verbo MULTIPLY

Verbo usado para a multiplicação de valores.

$\text{MULTIPLY Operand\#il \{BY\} \left\{ \begin{array}{l} \text{OperandResult\#i [ROUNDED]} \dots \\ \text{Operand\#il GIVING Result\#i [ROUNDED]} \dots \end{array} \right\} }$ $[\text{ON SIZE ERROR StatementBlock END - MULTIPLY}]$
---

### FORMATO 1

**MULTIPLY** <operando1> **BY** <operando-m1> [ROUNDED]... <operando-mN> [ROUNDED]  
 [on size error BlocoDeDeclarações end - MULTIPLY]

O operando1 é multiplicado pelos valores já existentes nos operandos operandom1 ... operandomN .

Os operandos à direita da cláusula BY só podem ser variáveis! Estas assumirão o resultado.

### FORMATO 2

**MULTIPLY** <operando1> **BY** <operando-m1> **GIVING** <variavel1> [ROUNDED]... <variavelN> [ROUNDED]  
 [on size error BlocoDeDeclarações end - MULTIPLY]

O produto do operando1 pelo operando-m1 é colocado na(s) variáveis vaiável1 .. variável n, que perdem o seu valor inicial.

## Exemplos:

1.

MULTIPLY preco BY Quantidade GIVING Produto ROUNDED.

(multiplica o preco pela quantidade e coloca o resultado na variável produto, sendo este arredondado)

2.

MULTIPLY dinheiro BY Membros GIVING Total.  
 DISPLAY "Alerta: resultado demasiado grande para a variável Total."  
 01 dinheiro PIC 9(3)V99 VALUE 052.24.  
 01 Membros PIC 9(4) VALUE 1024.  
 01 cheques PIC 9(5)V99 VALUE ZEROS.

	dinheiro	Membros	Total
<b>Antes</b>	052.24	1024	00000.00
<b>Depois</b>	052.24	1024	<b>53493.76</b>

1.

WORKING-STORAGE SECTION.  
 77 A PIC S 9(002) VALUE 0.  
 77 B PIC S 9(002) VALUE 0.  
 77 C PIC S9(003).  
 PROCEDURE DIVISION.  
 DISPLAY "VALOR DE A:  
 ACCEPT A  
 DISPLAY "VALOR B: "  
 ACCEPT B  
 MULTIPLY A BY B GIVING C  
 ON SIZE ERROR  
 DISPLAY "IMPOSSIVEL CALCULAR, VARIÁVEL PEQUENA !"  
 GOBACK  
 END-MULTIPLY  
 DISPLAY " A \* B = " C  
 ACCEPT OMITTED  
 GOBACK.

## O verbo DIVIDE

Verbo usado para a divisão de valores.

$$\text{DIVIDE Operand\#il} \left\{ \begin{array}{l} \text{BY} \\ \text{INTO} \end{array} \right\} \left\{ \begin{array}{l} \{\text{OperandResult\#i [ROUNDED]}\}... \\ \{\text{Operand\#il GIVING Result\#i [ROUNDED]}\}... \end{array} \right\}$$

[ON SIZE ERROR StatementBlock END - DIVIDE]

$$\text{DIVIDE Operand\#il} \left\{ \begin{array}{l} \text{INTO} \\ \text{BY} \end{array} \right\} \text{Operand\#il GIVING} \{\text{Quotient\#i [ROUNDED]}\} \text{REMAINDER Remainder\#i}$$

[ { ON SIZE ERROR } StatementBlock END - DIVIDE ]

O parâmetro **REMAINDER** só pode ser usado com os formatos 2 e 3 (GIVING) , e quando os operandos envolvidos forem números inteiros. A sua função é guardar o resto da divisão na variável resto.

## FORMATO 1

**DIVIDE** <operando1> **INTO** <operando-d1> [ROUNDED]...  
[on size error BlocoDeDeclarações end - DIVIDE]

Divide o operando-d1 pelo operando1 sendo guardado o resultado no operando-d1

**DIVIDE** operando1 **INTO** operandod1

## FORMATO 2

**DIVIDE** <operando1> **INTO** <operando-d1> **GIVING** <variavel1> [ROUNDED]... <variavelN> [ROUNDED]  
[on size error BlocoDeDeclarações end - DIVIDE]

Divide o operando-d1 pelo operando1 sendo guardado o resultado em variavel1, permanecendo os valores de operando1 e operando-d1 inalterados:

**DIVIDE** operando1 **INTO** operandod1 **GIVING** variavel1

## FORMATO 3

**DIVIDE** <operando1> **BY** <operando-d1> **GIVING** <quociente> [ROUNDED] REMAINDER resto  
[ON SIZE ERROR/NOT ON SIZE ERROR BlocoDeDeclarações end - DIVIDE]

Divide o operando1 pelo operando-d1 sendo guardado o resultado na variavel **quociente** (após ter sido inicializada a zero):

**DIVIDE** operando1 **BY** operandod1 **GIVING** variavel1

### Exemplos:

1.

DIVIDE 11 INTO Quoc ON SIZE ERROR GO TO FIM

2.

DIVIDE 15 INTO Num1, Num2.  
01 Num1 PIC 9(4) VALUE 2444.  
01 Num2 PIC 9(3) VALUE 354.

	Num1	Num2
<b>Antes</b>	2444	354
<b>Depois</b>	162	023

3. Neste exemplo o resultado do valor calculado não é um valor inteiro, por isso, o dígito à esquerda do ponto decimal é truncado. Devido a ter sido requerido o ROUNDED, o resultado foi arredondado para 272 (de 271.7826086956522).

DIVIDE Qtd BY Unidades GIVNG Media ROUNDED.  
01 Qtd PIC 9(5) VALUE 31255.  
01 Unidades PIC 9(3) VALUE 115.  
01 Media PIC 9(4) VALUE ZEROS.

	Qtd	Unidades	Media
<b>Antes</b>	31255	115	0000
<b>Depois</b>	31255	115	0272

4. Este exemplo usa o segundo formato do DIVIDE, obtendo tanto o quociente como o resto da divisão:

```
DIVIDE 215 BY 10 GIVING quociente REMAINDER Rem.
      01 quociente PIC 999 VALUE ZEROS.
      01 Rem PIC 9 VALUE ZEROS.
```

	quociente	Rem
<b>Antes</b>	000	0
<b>Depois</b>	<b>021</b>	<b>5</b>

## Exercícios:

1. Considere o seguinte programa em COBOL:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. F3_Exercicio1.
*> Programa sobre operações aritméticas
DATA DIVISION.
    WORKING-STORAGE SECTION.
        01 Num1    PIC 9 VALUE 5.
        01 Num2    PIC 9 VALUE 4.
        01 Resultado PIC 99 VALUE ZEROS.
PROCEDURE DIVISION.
CalcularResultado.
    DISPLAY "Introduza um número com um só dígito: "
    ACCEPT Num1
    DISPLAY "Introduza um número com um só dígito: "
    ACCEPT Num2
    MULTIPLY Num1 BY Num2 GIVING Resultado
    DISPLAY "O resultado é = ", Resultado
    STOP RUN.
```

a) Explique cada um das linhas.

b) Apresente o output do programa, supondo que que são introduzidos os valores 3 e 6, respetivamente, para Num1 e Num2.

2. De acordo com os itens de dados definidos em baixo, preencha a tabela de acordo com as declarações apresentadas:

```
01 Num1 PIC 99.
01 Num2 PIC 99.
01 Num3 PIC 99.
01 Num4 PIC 99.
```

	Valores antes da declaração				Valores após a declaração			
Declaração	Num1	Num2	Num3	Num4	Num1	Num2	Num3	Num4
ADD Num1 TO Num2	25	30						
ADD Num1, Num2 TO Num3, Num4	13	04	05	12				
ADD Num1, Num2, Num3 GIVING Num4	04	03	02	01				
SUBTRACT Num1 FROM Num2 GIVING Num3	04	10	55					
SUBTRACT Num1, Num2 FROM Num3	05	10	55					
SUBTRACT Num1, Num2 FROM Num3 GIVING Num4	05	10	55	20				
MULTIPLY Num1 BY Num2	10	05						
MULTIPLY Num1 BY Num2 GIVING Num3	10	05	33					
DIVIDE Num1 INTO Num2	05	64						
DIVIDE Num2 BY Num1 GIVING Num3 REMAINDER Num4	05	64	24	88				
COMPUTE Num1 = 5 + 10 * 30 / 2	25							

3. Dado :

01 A.

05 PIC B X(4) VALUE 'THIS'.

05 PIC C X(2) VALUE 'IS'.

05 PIC D X(3) VALUE 'FUN'.

01 W.

05 PIC X X(3).

05 PIC Y X(2).

05 PIC Z X(3).

Indica qual seria o conteúdo de X, Y e Z, após

MOVE A TO W

4. Exprime as seguintes fórmulas usando a instrução/verbo COMPUTE

a.  $a = \frac{b}{c} + b - (c \times d)$

b.  $a = \frac{b+c}{d} - \frac{c}{e+f}$

c.  $c = \frac{(a+b)^2}{a-b} + a - (b + c)$

5. Escreve um programa em Cobol que simule uma calculadora. Deve efetuar as operações básica potenciação, multiplicação divisão, soma e subtração. Os dados devem ser reais e fornecidos pelo utilizador e não devem ultrapassar valores de 3 dígitos inteiros e devem ter 3 casas decimais. Ao apresentar o resultado da divisão deve apresentar o valor arredondado.

**Bibliografia/webgrafia:**

<https://mainframesupport.wordpress.com/2012/07/15/figurative-constants/>

<http://www.mainframestechhelp.com/tutorials/cobol/>

Beginning COBOL for Programming, Michael Coughlan, Editora Apress

Linguagem Cobol, Marcio Adroaldo da Silva em [www.controlsyst.com](http://www.controlsyst.com)

Mainframe Apostila de Cobol, G & P Treinamentos em <http://www.csis.ul.ie/cobol/>

<https://www.apostilando.com/apostila/2962/manual-pratico-de-programacao-em-cobol>

<https://www.tutorialspoint.com/cobol/>

COBOL in 21 days, em <http://kickme.to/tiger/>

