



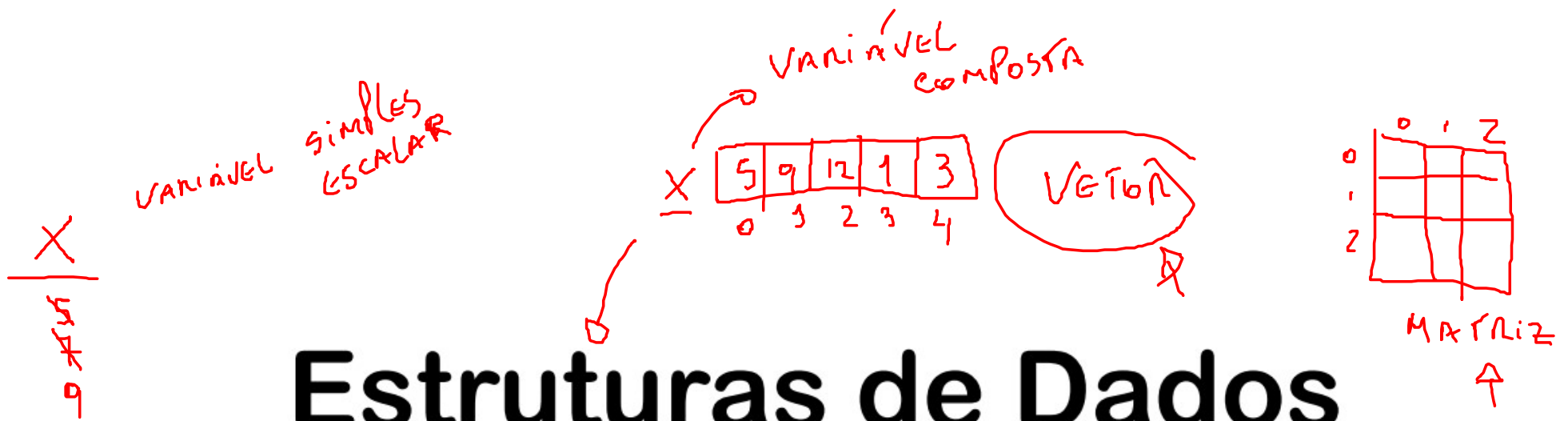
LÓGICA DE PROGRAMAÇÃO AULA 4

+55 (21) 99461-8818

@explicadoresnet

www.explicadores.net.br





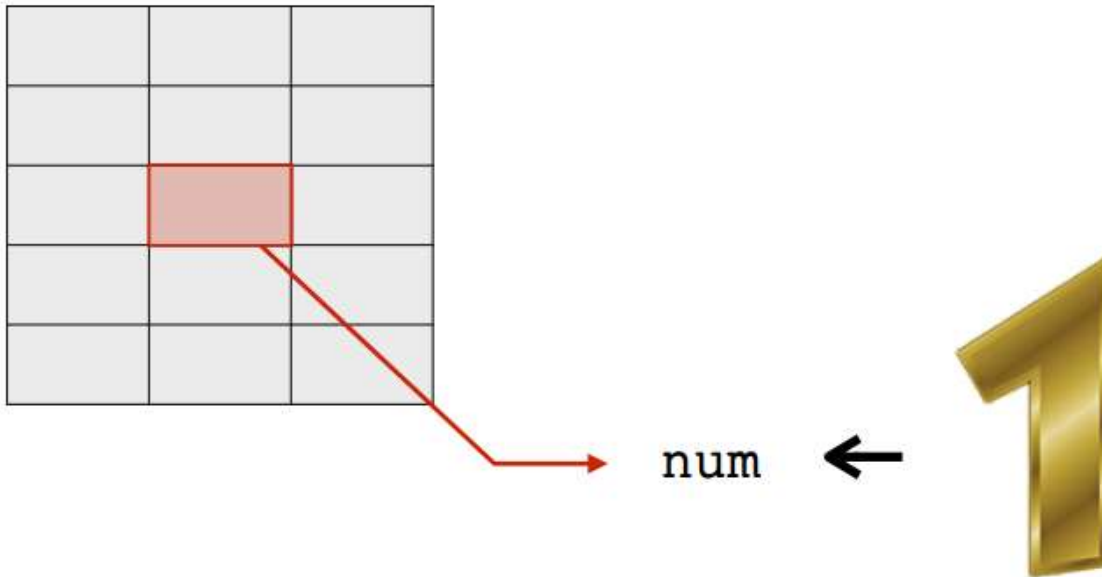
Estruturas de Dados

Homogêneas

↳ **um só tipo**

Variáveis Compostas Homogêneas

As **variáveis simples** permitem guardar **apenas um valor** por vez.



Variáveis Compostas Homogêneas

As **variáveis compostas** permitem guardar **vários valores** por vez, um em cada espaço.

[0]	[1]	[2]

- ✓ **variáveis indexadas**
- ✓ **variáveis subscritas**
- ✓ **arranjos**
- ✓ **vetores**
- ✓ **matrizes**
- ✓ **tabelas em memória**
- ✓ **arrays**

num [1] ← 4

$$\frac{X}{5}$$

	1	1	1	1		
Y	10	12	15	18	20	20
	0	1	2	3	4	5

4ª posição?
18

$$Y[3] = 18$$

$$Y[5] = 20$$

$$Y[4+1] = 20$$

$$Y[5] + Y[1] = 32$$

$$20 + 12$$

$$L \times C$$

$$4 \times 3 = 12$$

Z	1	2	3	2	1
	0	1	2	3	4

$$Z[Z[Z[4]]] = 3$$

	0	1	2
H ⁰	5	7	8
1	4	5	6
2	3	2	1
3	3	4	1

$$H\left[\begin{smallmatrix} L \\ 1 \end{smallmatrix}, \begin{smallmatrix} C \\ 2 \end{smallmatrix}\right] = 6$$

$$H[3,1] = 4$$

$$H\left[\begin{smallmatrix} 0+1, 1+1 \\ 1, 2 \end{smallmatrix}\right] = 6$$

$$H\left[\begin{smallmatrix} 1, 1 \\ 5 \end{smallmatrix}\right] + H\left[\begin{smallmatrix} 3, 1 \\ 4 \end{smallmatrix}\right] = 9$$

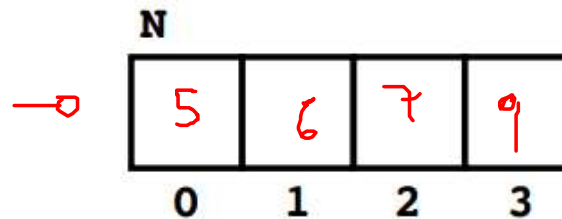
5

Fundamentos

```
Algoritmo Composto
var
  N: vetor[0..3] de Inteiro ④
inicio
  Escreva("Informe um valor")
  Leia(N[0]) ①
  Escreva("Informe um valor")
  Leia(N[1])
  Escreva("Informe um valor")
  Leia(N[2])
  Escreva("Informe um valor")
  Leia(N[3])
FimAlgoritmo
```

Diagram illustrating the algorithm and its execution flow:

- A blue arrow points from the word "dimensão" to the range "0..3" in the array declaration.
- A red circle with the number "4" is next to the declaration "N: vetor[0..3] de Inteiro".
- A blue box highlights the first two lines of the loop: "Escreva('Informe um valor')", "Leia(N[0])".
- A red circle with the number "1" is next to "Leia(N[0])".
- Four blue curved arrows point from the "Leia" statements to the corresponding indices in the array diagram below: from "Leia(N[0])" to index 0, from "Leia(N[1])" to index 1, from "Leia(N[2])" to index 2, and from "Leia(N[3])" to index 3.



Fundamentos

Algoritmo Composto

var

N: vetor[0..3] de Inteiro

i: Inteiro

inicio

Para i <- 0 ate 3 faca

Escreva("Informe um valor")

Leia(N[i])

FimPara

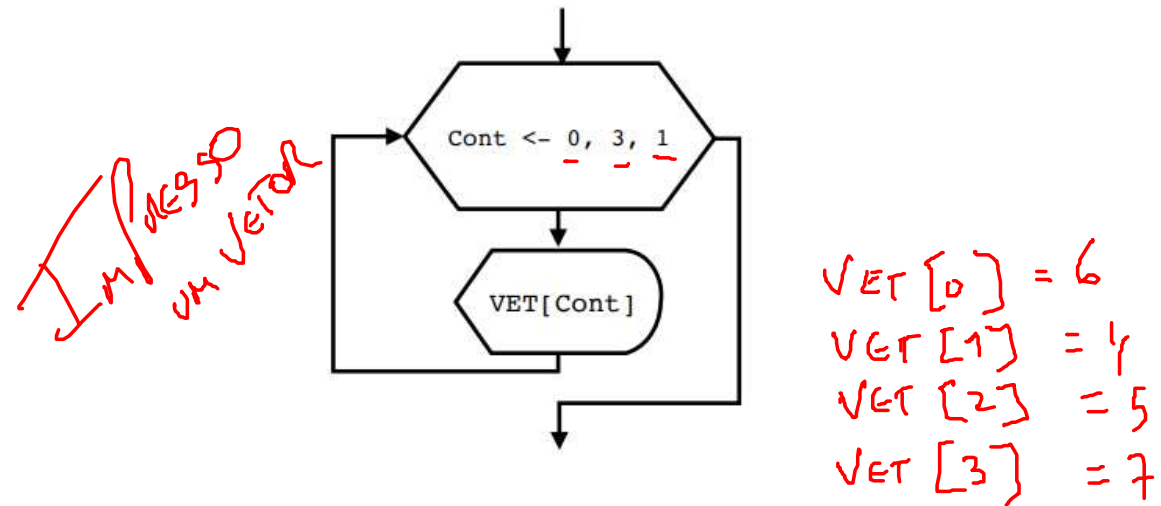
FimAlgoritmo

11
2
3

N[0]
N[1]
N[2]
N[3]

N			
5	12	8	7
0	1	2	3

Exibição da Estrutura



VET

6	4	5	7
0	1	2	3

Variáveis Compostas Homogêneas Multidimensionais

[0]			
[1]			
	[0]	[1]	[2]

num [1 , 2] ←

4



Fundamentos

```

Algoritmo Composto
var
  M: vetor[0..2, 0..3] de Inteiro
  L, C: Inteiro
inicio
  Para L <- 0 ate 2 faca
    Para C <- 0 ate 3 faca
      Escreva("Informe um valor")
      Leia(M[L,C])
    FimPara
  FimPara
FimAlgoritmo
    
```

Handwritten table for L and C values:

L	C
0	0
1	0
2	0
0	1
1	1
2	1
0	2
1	2
2	2
0	3
1	3
2	3

M

0				
1				
2				
	0	1	2	3

Handwritten list of (L, C) pairs:

- 0,0
- 0,1
- 0,2
- 0,3
- 1,0
- 1,1
- 1,2
- 1,3
- 2,0
- 2,1
- 2,2
- 2,3

Isolando uma Linha

```
Algoritmo Composto
var
  M: vetor[0..2, 0..3] de Inteiro
  L, C: Inteiro
inicio
  Para C <- 0 ate 3 .faca
    Escreva(M[1,C])
  FimPara
FimAlgoritmo
```

1,0
1,1
1,2
1,3

M					
0					
1					
2					
	0	1	2	3	

Isolando uma Coluna

```
Algoritmo Composto
var
  M: vetor[0..2, 0..3] de Inteiro
  L, C: Inteiro
inicio
  Para L <- 0 ate 2 faça
    Escreva(M[L,2])
  FimPara
FimAlgoritmo
```

0,2
1,2
2,2

M

0				
1				
2				
	0	1	2	3

Isolando a Diagonal Principal

Algoritmo Composto

var

M: vetor[0..2, 0..2] de Inteiro

L, C: Inteiro

inicio

Para C ← 0 ate 2 faça

Escreva(M[C,C])

FimPara

FimAlgoritmo

→ b {
0,0 ✓
1,1 ✓
2,2 ✓

0,0
1,1
2,2

M

0	0,0 ✓		
1		1,1 ✓	
2			2,2 ✓
	0	1	2

– Considerando a matriz bidimensional TABELA [1..8, 1..5], assinale a afirmativa correta.

$$\begin{array}{c} \text{L} \quad \text{C} \\ 8 \times 5 = 40 \end{array}$$

- a) A matriz tem 5 linhas, 8 ~~colunas~~ e 40 elementos.
- b) A matriz tem 8 linhas, 5 ~~colunas~~ e no mínimo 40 elementos.
- c) A matriz tem 5 ~~linhas~~, 8 ~~colunas~~ e podem ser armazenados até 40 elementos.
- ~~d)~~ A matriz tem 8 linhas, 5 ~~colunas~~ e podem ser armazenados até 40 elementos.

– Dado o seguinte trecho de programa em Português Estruturado,

```
J ← 2  
A[1] ← 5  
enquanto ( J < 8 ) faça  
    A[J] ← A[J - 1] + 3  
    J ← J + 1  
fim_enquanto
```

assinale a alternativa que contém o valor correto de A[6].

- a) 5
- b) 8
- c) 17
- d) 20

– No algoritmo do programa abaixo, em português estruturado, temos as seguintes entradas para os valores da variável X quando requisitados: $X[1] = 2$, $X[2] = 3$, $X[3] = 8$, $X[4] = 3$. Identifique nas respostas abaixo qual será a saída do programa.

programa CONTA

var

RESULTADO : real
X : conjunto[1..4] de real
T : inteiro
Y : inteiro

início

Y $\leftarrow$ 4
RESULTADO $\leftarrow$ 0
para T de 1 até 4 passo 1 faça
 leia X[T]
 RESULTADO $\leftarrow$ RESULTADO + X[T]
fim_para
RESULTADO $\leftarrow$ RESULTADO / Y
escreva RESULTADO

fim

- a) 2
- b) 8
- c) 3
- d) 4

Variáveis Compostas Heterogêneas

~~em~~ valores de tipos diferentes!

As **variáveis compostas heterogêneas** ou **registros** permitem definir o **tipo primitivo** de cada posição.

X : INTEIRO

```
Algoritmo Composto
tipo
  Ficha = Registro
  Registro
    cod: Inteiro
    nome: Caractere
    peso: Real
  FimRegistro
var
  Pessoa: Ficha
início
  Leia (Pessoa.cod)
  Leia (Pessoa.nome)
  Leia (Pessoa.peso)
FimAlgoritmo
```

Pessoa

cod	—
nome	—
peso	—



Registros de Vetores

Algoritmo Composto

tipo

Bimestre = vetor[1..4] de Real

Aluno = Registro

mat: Inteiro

nome: Caractere

notas: Bimestre

FimRegistro

var

MeuAluno: Aluno

inicio

Leia (MeuAluno.nome)

Leia (MeuAluno.notas[1])

Leia (MeuAluno.notas[3])

FimAlgoritmo

MeuAluno

Mat				
nome				
notas	•		•	
	1	2	3	4



Vetores de Registros

Algoritmo Composto

tipo

Ficha = Registro

cod: Inteiro

nome: Caractere

peso: Real

FimRegistro

var

Cadastro: vetor[1..3] de Ficha

inicio

Leia (Cadastro[1].nome)

Leia (Cadastro[2].peso)

FimAlgoritmo

Cadastro

cod		cod		cod	
nome	-	nome		nome	
peso		peso	.	peso	
1		2		3	

Pesquisa Sequencial

→ DESORGANIZADA

Consiste em efetuar a busca da informação desejada **a partir do primeiro elemento** sequencialmente **até o último**.

Localizando a informação no caminho, ela é apresentada. Este método de pesquisa é **lento**, porém **eficiente** caso o vetor encontra-se com seus elementos **desordenados**.

→

1	5	4	15	25	35	3	18	20	21
---	---	---	----	----	----	---	----	----	----

Pesquisa Binária

Divide o vetor em duas partes e procura saber se a informação a ser pesquisada está acima ou abaixo da linha de divisão.

Se estiver acima, toda a metade abaixo é **desprezada**.

E assim vai sendo executada até **encontrar ou não** a informação pesquisada.

Esse método é em média mais **rápido** que o sequencial, porém exige que a matriz esteja **previamente classificada**.

ORDENADAS

1	5	4	15	25	35	3	18	20	21
1	3	4	5	15	18	20	21	25	35

sort

CLASSIFICAÇÃO

Método Top-Down

→ Lógica modular

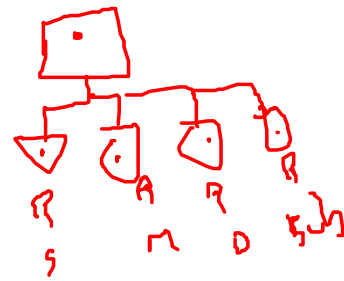


Modulo ou sub-rotinas

O método mais adequado para a **programação estruturada**.

Antes de iniciar o programa, deverá ter em mente as **tarefas principais** que este deverá executar.

Conhecidas todas as tarefas, imaginar como deverá ser o **programa principal**, que vai controlar todas as outras tarefas.



Tendo definido o programa principal, é iniciado o **processo de detalhamento**. Desta forma são definidos vários algoritmos, um para cada rotina separada.



Método Top-Down



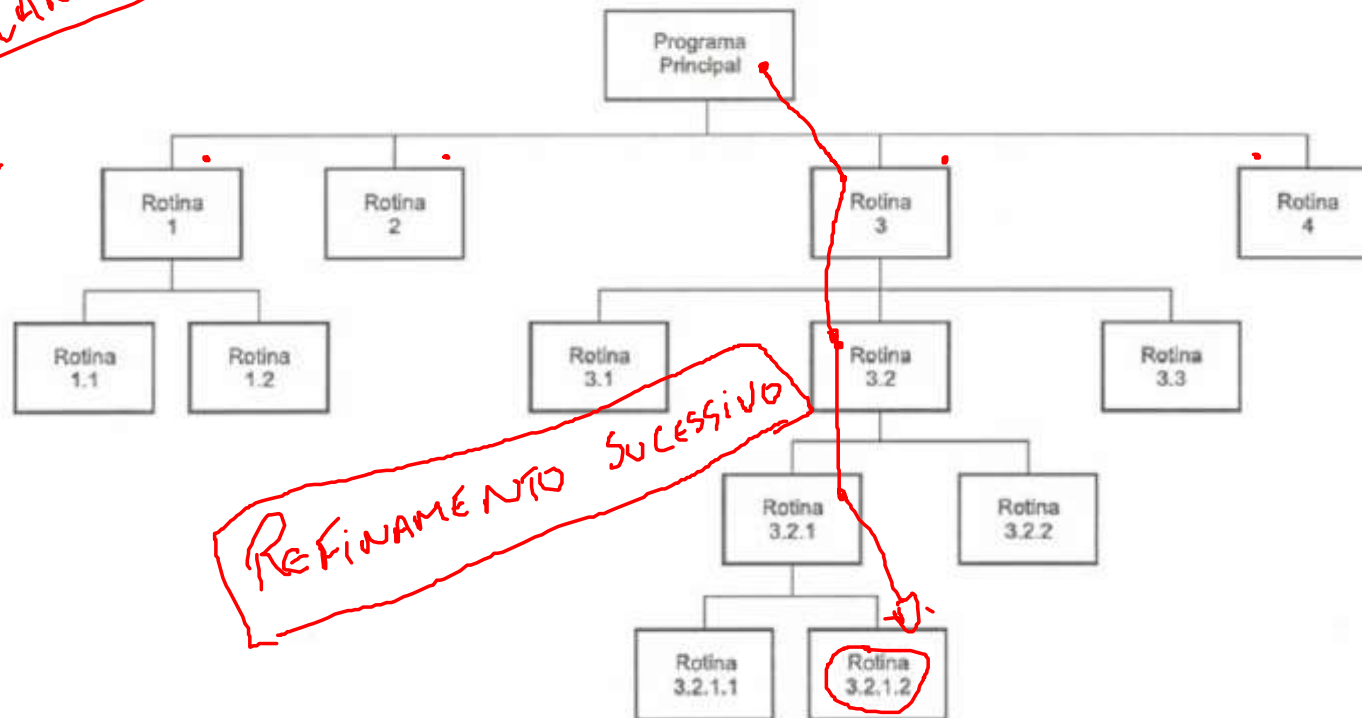
O método **Top-Down** faz com que o programa tenha uma estrutura semelhante a um **organograma**. *hierarquia*

A utilização do método “**de cima para baixo**” permite que seja efetuado cada **módulo** do programa em separado.

Outro detalhe a ser considerado é que muitas vezes existem em um programa trechos de códigos que são **repetidos várias vezes**. Esses trechos poderão ser utilizados como **sub-rotinas**, proporcionando um programa menor e mais fácil de ser alterado num futuro próximo.

Método Top-Down

MODULARIZAÇÃO



REFINAMENTO SUCESSIVO