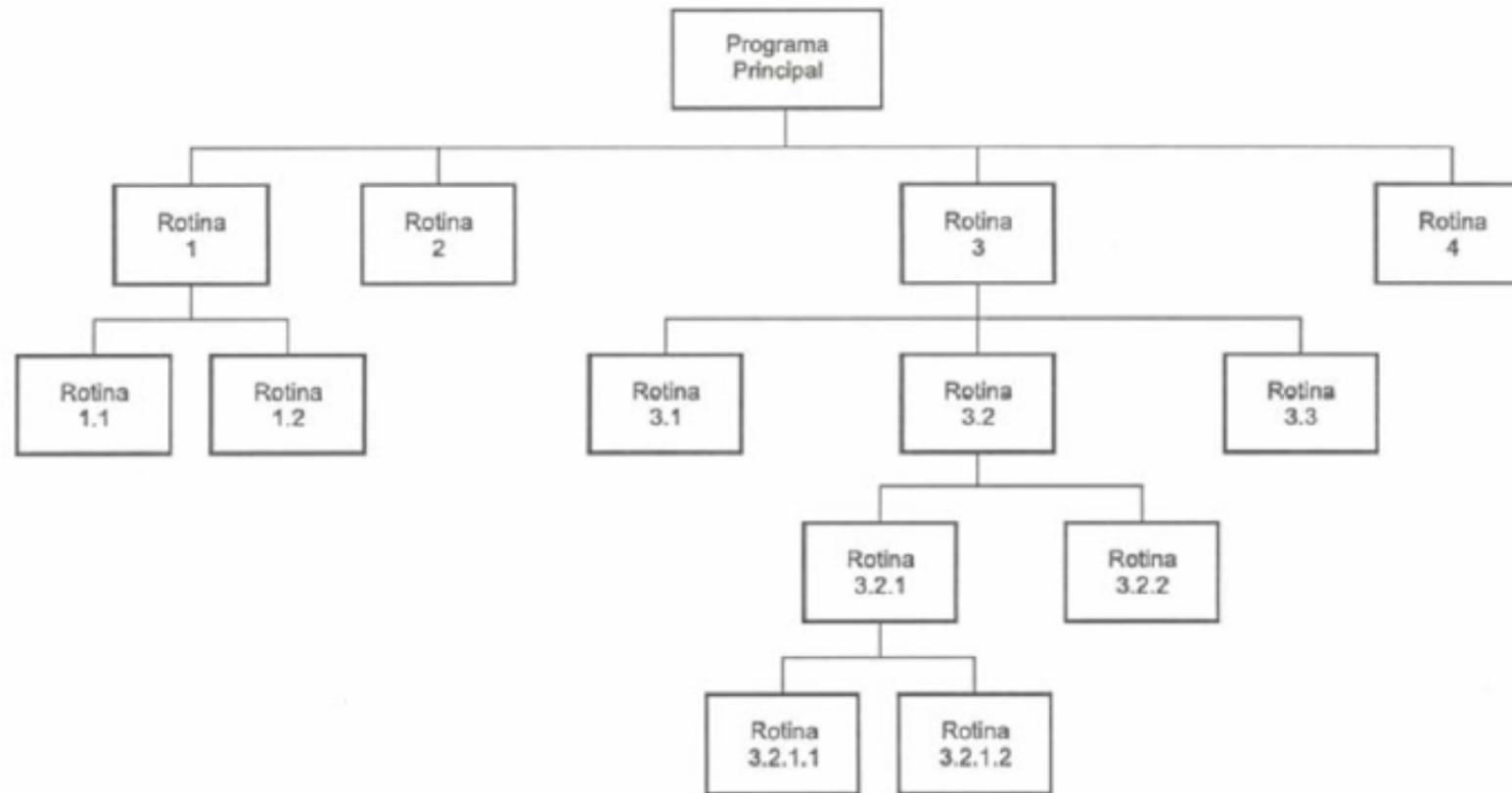




**LÓGICA DE PROGRAMAÇÃO**  
**AULA 5**

# Método Top-Down



# Sub-rotinas

Permite **dividir um problema** em problemas menores.

Uma **sub-rotina** é na verdade um **programa**, e sendo um programa poderá efetuar diversas operações computacionais (entrada, processamento e saída) e deverá ser tratada como foram os programas projetados até este momento.

As **sub-rotinas** são utilizadas na divisão de algoritmos complexos, permitindo assim possuir a **modularização** de um determinado problema, considerado grande e de difícil solução.

# Sub-rotinas

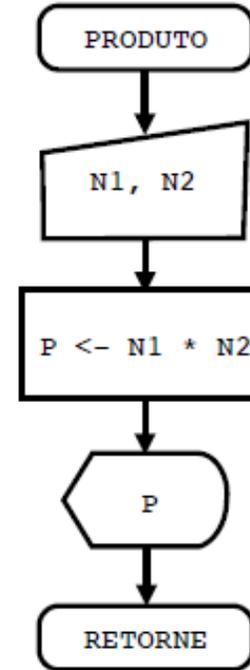
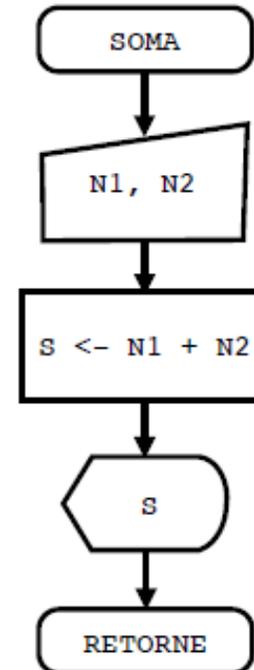
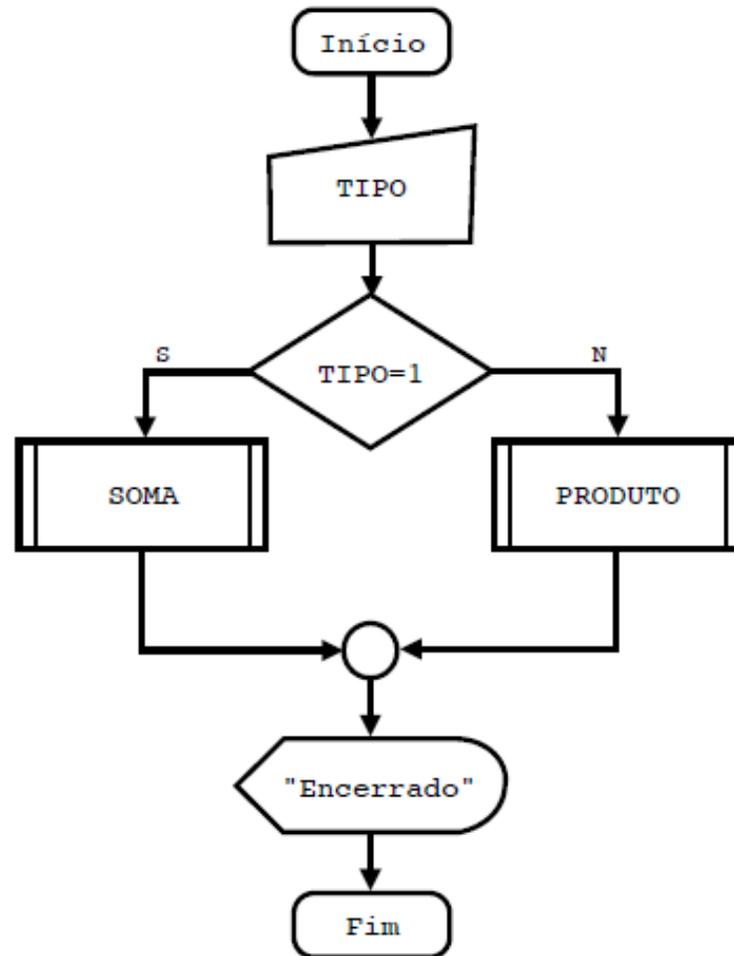
Existem dois tipos de **sub-rotinas**:

Os **Procedimentos** são blocos de programa executados e o controle de processamento retorna automaticamente para a primeira linha de instrução após a linha que efetuou a chamada da sub-rotina.

As **Funções** são blocos de programa capazes de retornar um determinado valor para o programa que fez a chamada da sub-rotina.



# Sub-rotinas



# Procedimentos

```
Algoritmo Rotinas01  
var  
    tipo: Inteiro
```

## Procedimento SOMA()

```
var  
    N1, N2, S: Inteiro  
inicio  
    Leia(N1, N2)  
    S <- N1 + N2  
    Escreva(S)  
FimProcedimento
```

## Procedimento PRODUTO()

```
var  
    N1, N2, P: Inteiro  
inicio  
    Leia(N1, N2)  
    P <- N1 * N2  
    Escreva(P)  
FimProcedimento
```

```
inicio  
    Leia(tipo)  
    Se (tipo = 1) entao  
        SOMA()  
    senao  
        PRODUTO()  
    FimSe  
    Escreva("Encerrado")  
FimAlgoritmo
```

# Funções

```
Algoritmo Rotinas02  
var  
    tipo, R: Inteiro
```

```
Funcao SOMA()
```

```
var  
    N1, N2, S: Inteiro  
inicio  
    Leia(N1, N2)  
    S ← N1 + N2  
    Retorne(S)  
FimFuncao
```

```
Funcao PRODUTO()
```

```
var  
    N1, N2, P: Inteiro  
inicio  
    Leia(N1, N2)  
    P ← N1 * N2  
    Retorne(P)  
FimFuncao
```

```
inicio  
    Leia(tipo)  
    Se (tipo = 1) entao  
        R ← SOMA()  
    senao  
        R ← PRODUTO()  
    FimSe  
    Escreva(R)  
FimAlgoritmo
```

# Escopo Global e Local

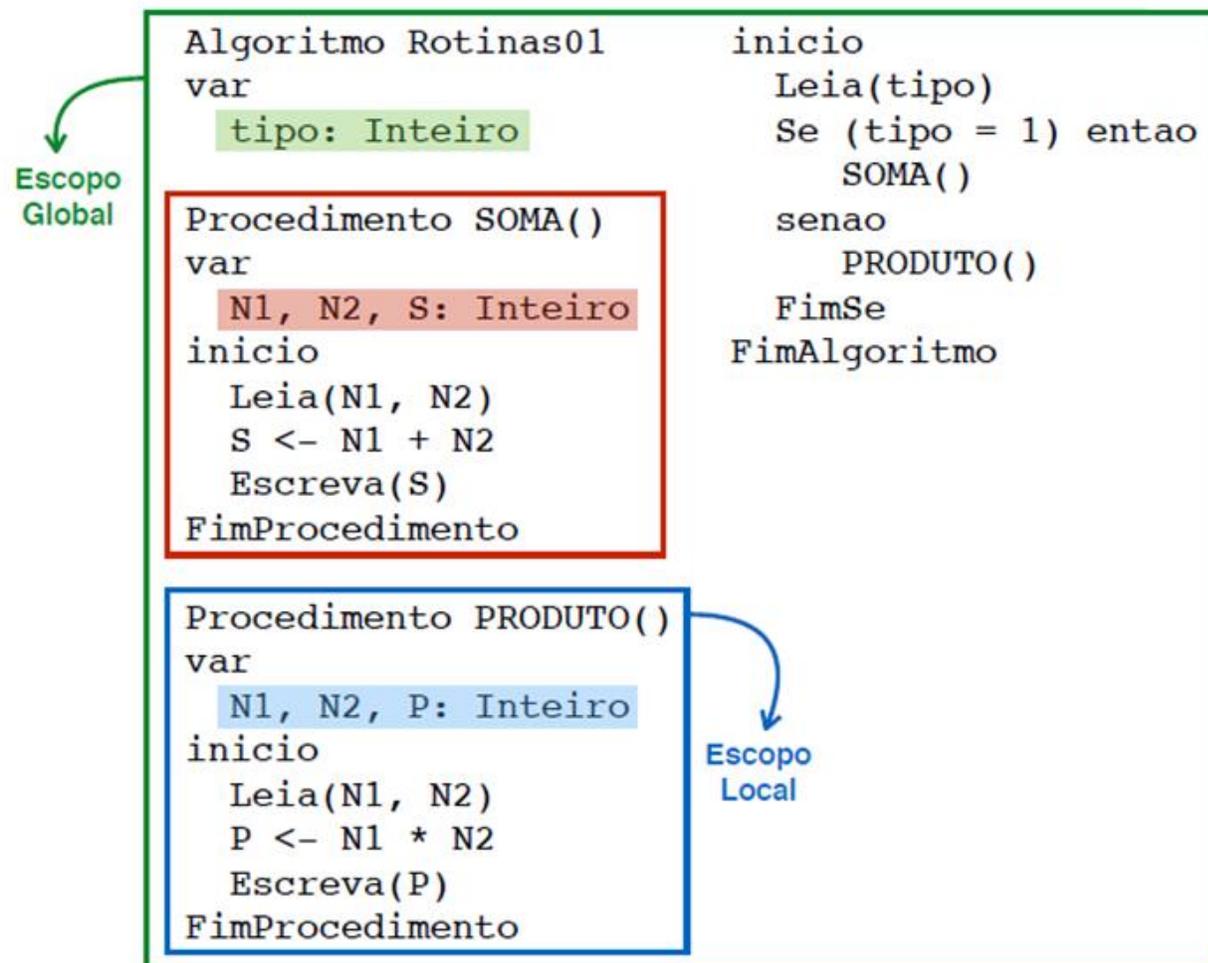
O **escopo** de uma variável ou sua **abrangência** está vinculado à sua **visibilidade** (global ou local) em relação às sub-rotinas de um programa, sendo que a sua visibilidade está relacionada à sua **hierarquia**.

Uma variável é considerada global quando é declarada no início do algoritmo principal de um programa, podendo ser utilizada por qualquer sub-rotina subordinada ao algoritmos principal.

Uma variável é considerada local quando é declarada no no corpo de uma sub-rotina, podendo ser utilizada apenas por essa sub-rotina.



## Escopo Global e Local



# Uso de Parâmetros

Têm por finalidade servir como um ponto de **comunicação bidirecional** entre uma sub-rotina e o programa principal ou uma outra sub-rotina hierarquicamente de nível mais alto.

```
A ← 3
B ← 5
SOMA(A, B)
SOMA(2, 6)
```

```
Procedimento SOMA(N1, N2: Inteiro)
var
    S: Inteiro
inicio
    S ← N1 + N2
    Escreva(S)
FimProcedimento
```

# Parâmetros Formais e Reais

São **formais** quando forem declarados por meio de variáveis **juntamente com a identificação** do nome da sub-rotina, tratados exatamente da mesma forma que são tratadas as variáveis globais ou locais.

São **reais** quando substituïrem os parâmetros formais, quando da **utilização da sub-rotina** por um programa principal ou por uma rotina chamadora.



# Passagem de Parâmetros

A **passagem de parâmetro por valor** caracteriza-se pela **não-alteração** do valor do **parâmetro real** quando o **parâmetro formal** é manipulado dentro da sub-rotina.

O valor passado pelo **parâmetro real** é **copiado** para o **parâmetro formal**, que no caso assume o papel de variável local da sub-rotina.

Desta forma, qualquer **modificação** que ocorra na variável local da sub-rotina **não afetará** o valor do parâmetro real.



# Passagem de Parâmetros

A **passagem de parâmetro por referência** caracteriza-se pela ocorrência de **alteração do valor** do **parâmetro real** quando o **parâmetro formal** é manipulado dentro da sub-rotina.

Desta forma, qualquer **modificação** feita no **parâmetro formal implica em alteração** no **parâmetro real** correspondente.

A alteração efetuada é **devolvida** para a rotina chamadora.

# Passagem de Parâmetros

```
Procedimento Calcular(X, Y: Inteiro)
inicio
  X ← X + 2
  Y ← Y - 2
  Escreva(X)
  Escreva(Y)
FimProcedimento
```

```
a ← 3
b ← 5
Calcular(a, b)
Escreva(a)
Escreva(b)
```

# Passagem de Parâmetros

```
Procedimento Calcular(var X, Y: Inteiro)
inicio
  X <- X + 2
  Y <- Y - 2
  Escreva(X)
  Escreva(Y)
FimProcedimento
```

```
a <- 3
b <- 5
Calcular(a, b)
Escreva(a)
Escreva(b)
```

# Anatomia de um Programa

“**Instrução** é um conjunto de caracteres referentes a uma função que o computador possa executar.”

“Se uma instrução está no formato de sequências de bits (0 e 1) em que o computador consegue executar diretamente, dizemos que está em **Linguagem de Máquina** ou **Absoluta**”

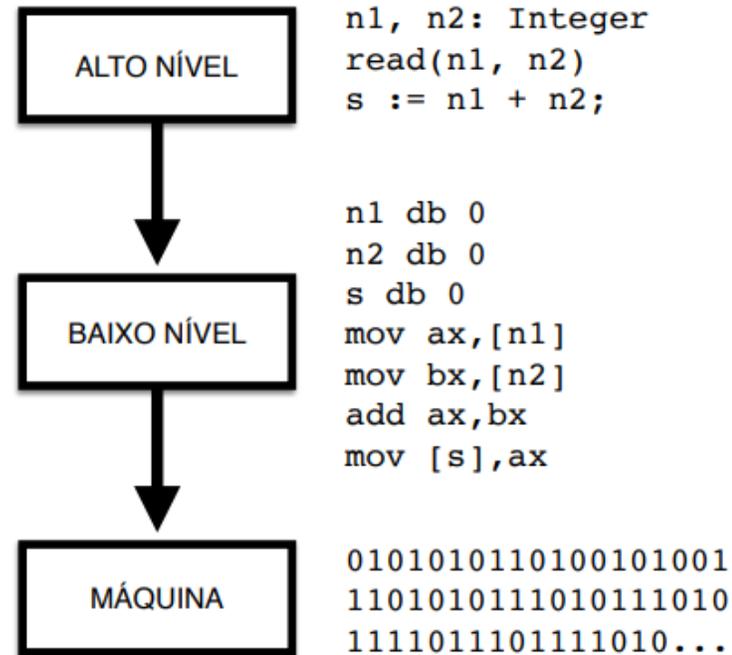
“Programadores usam uma **Linguagem Simbólica** (alto nível) para representar essas instruções, com uma sintaxe mais próxima do compreensível por nós.”

“Uma instrução em **Linguagem Simbólica**, pode significar várias instruções em **Linguagem de Máquina.**”

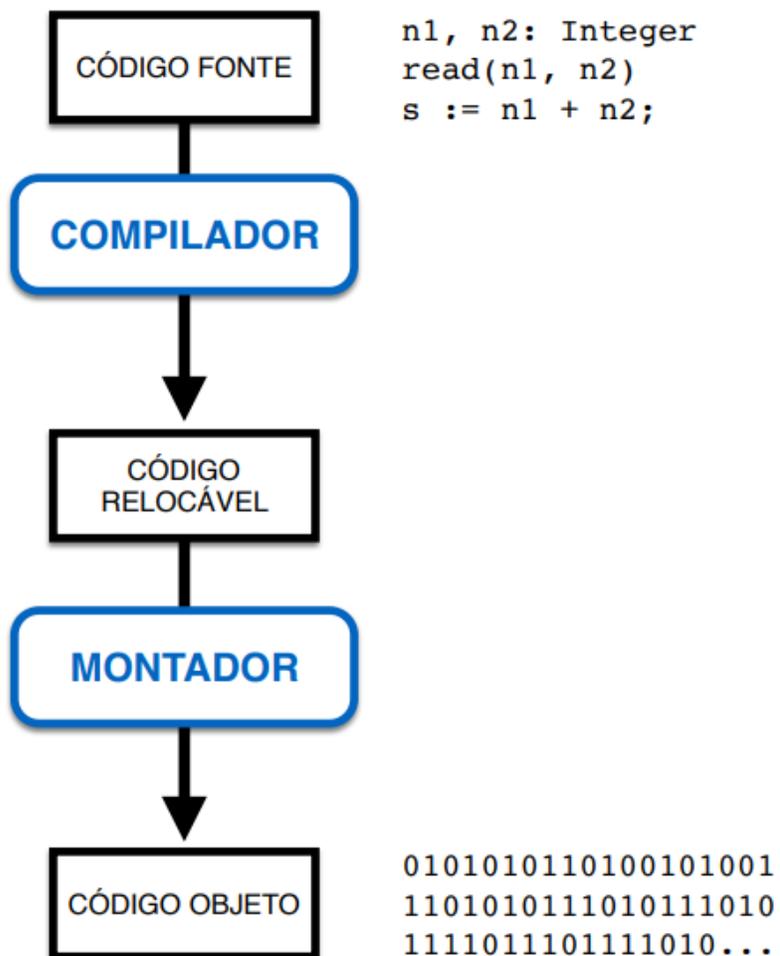


# Anatomia de um Programa

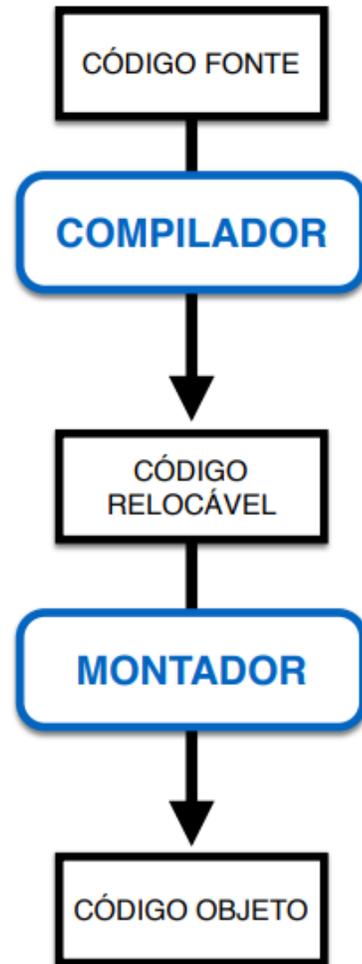
“Abaixo do nível das **Linguagens Simbólicas**, mas acima da **Linguagem de Máquina**, temos a **Linguagem Assembly** (baixo nível)”



# Processo de Compilação



# Processo de Compilação



“Um **Programa** pode existir em três níveis: **fonte** (simbólico), **relocável** e **objeto** (executável).”

“O **Compilador** vai verificar a sintaxe dos comandos, buscar por erros, realizar a tradução do código simbólico em múltiplas instruções essenciais.”

“O **Montador** ou **Assembler** realiza cálculos de endereçamento e transforma as instruções relocáveis em linguagem de máquina.”

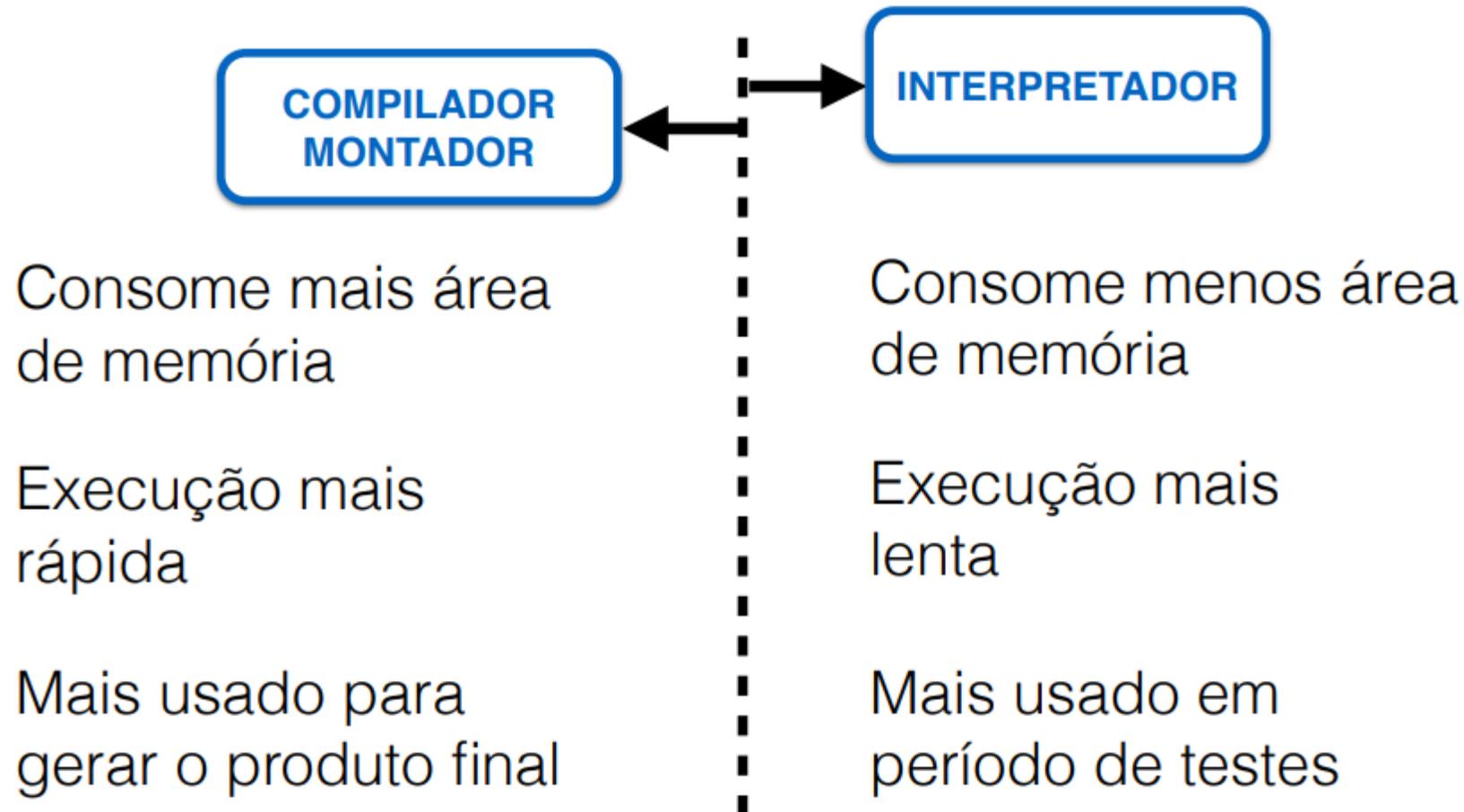


# Processo de Interpretação



“Um **Interpretador** faz a tradução gradativamente, comando por comando. Analisando sintaxe, procurando erro, convertendo o código e gerando a instrução **absoluta**.”

# Compilação vs. Interpretação



**85** – Parâmetros formais são aqueles declarados por meio de variáveis juntamente com a identificação do nome da sub-rotina. Parâmetros reais são aqueles que substituem os parâmetros formais quando da utilização da sub-rotina por um programa principal ou por uma rotina chamadora. Assinale a alternativa que contém a informação **incorreta** sobre passagens de parâmetros por valor e por referência a uma rotina.

- a) A passagem de parâmetro por referência caracteriza-se pela ocorrência de alteração do valor do parâmetro real quando o parâmetro formal é manipulado dentro da sub-rotina.
- b) No caso de passagem de parâmetros por valor, uma modificação que ocorra em uma variável local da sub-rotina afetará o valor do parâmetro real correspondente.
- c) A passagem de parâmetro por valor caracteriza-se pela não-alteração do valor do parâmetro real quando o parâmetro formal é manipulado dentro da sub-rotina.
- d) A passagem de parâmetro ocorre quando é feita uma substituição dos parâmetros formais pelos parâmetros reais no momento da execução da sub-rotina, e pode ser realizado de duas formas: por valor ou por referência.



**81** – Considerando os métodos de pesquisa em uma matriz. O método de pesquisa \_\_\_\_\_ divide a lista em duas partes e “procura” saber se a informação a ser pesquisada está acima ou abaixo da linha de divisão.

- a) seqüencial
- b) binária
- c) indexado
- d) indexado

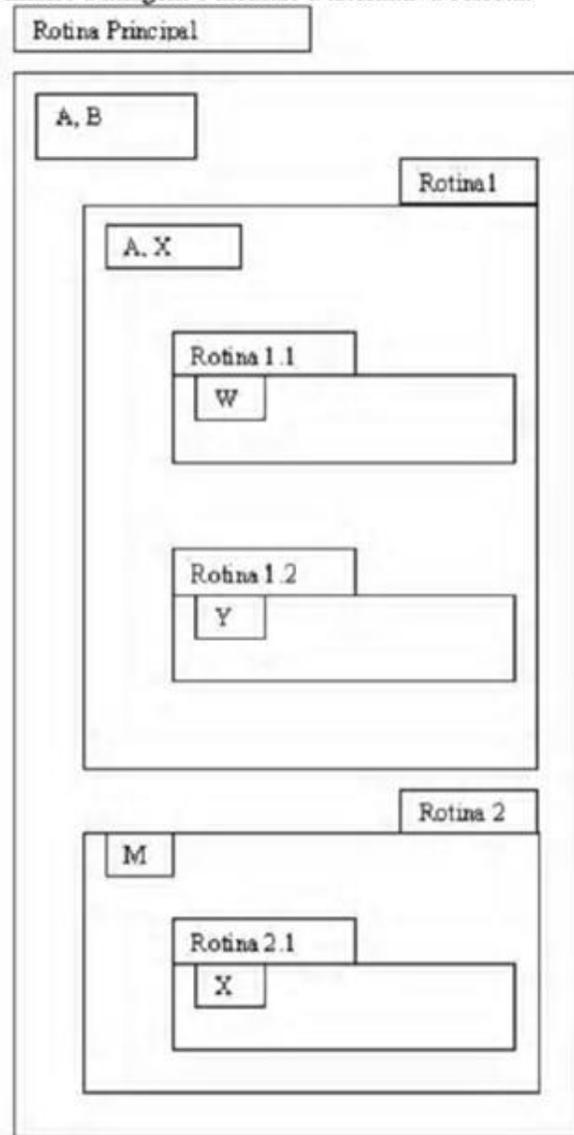
**66** – Assinale a alternativa correta em relação à definição de variáveis globais e locais.

- a) Uma variável global não pode ser visível a todas as sub-rotinas hierarquicamente subordinadas à rotina principal.
- b) As variáveis definidas como globais e locais precisam ser declaradas repetidas vezes dentro de cada sub-rotina.
- c) Uma variável local pode ser considerada global quando declarada no cabeçalho de uma sub-rotina, porém só é válida dentro da rotina à qual está declarada.
- d) Uma variável global é declarada no início do algoritmo principal de um programa, pode ser utilizada por qualquer sub-rotina subordinada ao algoritmo principal.

**46** – Marque a afirmativa correta.

- a) O método de pesquisa sequencial, geralmente, é mais rápido do que o método de pesquisa binária.
- b) O método de pesquisa binária ordena e pesquisa a informação desejada.
- c) O método de pesquisa binária, geralmente, é mais lento do que o método de pesquisa sequencial.
- d) O método de pesquisa binária necessita de que a informação esteja previamente ordenada.

- Analise a imagem e assinale a alternativa correta.



- a) A e B são variáveis locais às sub-rotinas 1 e 2.
- b) X é variável global em relação à rotina principal.
- c) A rotina 1.2 visualiza a variável W.
- d) M é global em relação à rotina 2.1.