

# Modularização

## - Variáveis globais

Variáveis globais são declaradas na seção de declarações de variáveis do algoritmo. Elas são conhecidas e podem ser alteradas por todos os módulos que constituem o algoritmo. Quando um módulo tem uma variável local ou um parâmetro com o mesmo nome de uma variável global o módulo dará preferência à variável local ou para o parâmetro. Vamos ver alguns exemplos:

# Modularização

## - Variáveis globais

Exemplo:

```
algoritmo "exemplo"
```

```
var
```

```
  j:inteiro
```

```
  procedimento f1(i: inteiro)
```

```
  inicio
```

```
    j<-18
```

```
    i<-17
```

```
    escreva (i)
```

```
  fimprocedimento
```

```
inicio
```

```
  j<-3
```

```
  f1(j)
```

```
  escreva (j)
```

```
fimalgoritmo
```

# Modularização

## - Variáveis globais

Exemplo:

```
algoritmo "exemplo"
```

```
var
```

```
  i:inteiro
```

```
  procedimento f1(i: inteiro)
```

```
  inicio
```

```
    i<-18
```

```
    escreva (i)
```

```
  fimprocedimento
```

```
inicio
```

```
  i<-3
```

```
  f1(i)
```

```
  escreva (i)
```

```
fimalgoritmo
```

### Exercício 38:

Analise o seguinte algoritmo e indique o que será impresso na saída padrão.

```
algoritmo "exercício variável global"
```

```
var num, first, sec: inteiro
```

```
    funcao func(first:inteiro; sec:inteiro):inteiro
```

```
    inicio
```

```
        first <- (first+sec)\2
```

```
        num <- num - first+1
```

```
        retorne (first)
```

```
    fimfuncao
```

```
inicio
```

```
    first <- 0
```

```
    sec <- 50
```

```
    num <- 10
```

```
    escreval ("num antes = ", num)
```

```
    escreval ("first antes = ", first)
```

```
    escreval ("sec antes = ", sec)
```

```
    num <- num + func(first, sec)
```

```
    escreval ("num depois = ", num)
```

```
    escreval ("first depois = ", first)
```

```
    escreval ("sec depois = ", sec)
```

```
280 fimalgoritmo
```

# Modularização

## Exercício 39:

Elabore um módulo que receba um vetor com quinze elementos de valores reais e retorne o mesmo invertido. **Observação:** não é permitida a utilização de um vetor auxiliar.

procedimento inveter\_vet(var vet: vetor [1..15] de real)

var i: inteiro

aux: real

inicio

para i de 1 ate 15\2 faça

aux <- vet[i]

vet[i] <- vet[15+1-i]

vet[15+1-i] <- aux

fimpara

fimprocedimento

// O número quinze está em vermelho pois representa

// o número de elementos do vetor.

// A única forma de um módulo retornar um vetor é

// através da passagem por referência.

## Modularização

### Exercício 40:

Elabore um algoritmo que manipule uma matriz de inteiros. O algoritmo deve possuir um módulo para inicializar a matriz com informações fornecidas pelo usuário, outro para apresentá-la na saída padrão, com o layout adequado, e por fim, um módulo que retorne os dois maiores valores contidos na matriz. O número de elementos contidos na matriz é fornecido pelo usuário, sendo que o número de elementos em uma dimensão não pode exceder 20. Os módulos aludidos devem ser utilizados de forma satisfatória pelo algoritmo e os módulos não devem fazer uso de variáveis globais em suas instruções.

# Modularização

## Exercício 41:

Os incas ficaram conhecidos pela grande civilização que reinou na região dos Andes durante vários séculos. O que pouca gente sabe é que os incas construíram pirâmides de base quadrada em que a única forma de se atingir o topo era seguir em espiral pela borda, que acabava formando uma escada em espiral. Estas pirâmides ainda se encontram escondidas na floresta amazônica e sua descoberta trará uma aplicação para este exercício.

## Modularização

Neste exercício você deverá fazer um algoritmo para verificar se uma matriz é ou não uma matriz inca. Seu algoritmo deve ter uma função que recebe como parâmetro, uma matriz quadrada  $A_{n \times n}$  de números inteiros e sua ordem retornando o resultado da verificação se a matriz é inca, ou seja, se partindo do canto superior esquerdo da matriz, no sentido horário, em espiral, a posição seguinte na ordem é o inteiro consecutivo da posição anterior. O valor de  $n$  não deve exceder 20.

# Modularização

Exemplo de uma matriz inca:

$$M = \begin{array}{|cccc|} \hline 1 & 2 & 3 & 4 \\ \hline 12 & 13 & 14 & 5 \\ \hline 11 & 16 & 15 & 6 \\ \hline 10 & 9 & 8 & 7 \\ \hline \end{array}$$