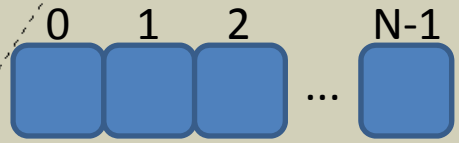
6.1 Arranjos (arrays)

Até ao momento apenas se trabalhou com números escalares.

Em LabView um arranjo (array) consiste numa colecção de elementos todos do **mesmo** tipo.

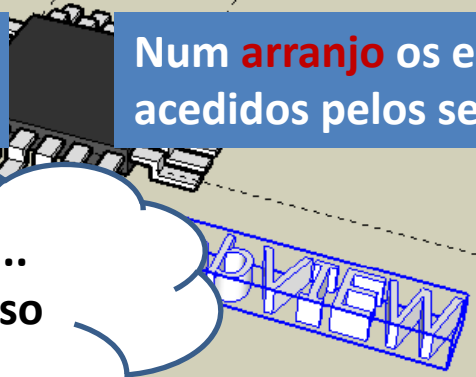
Um array pode ter uma ou mais **dimensões**.

Os índices são numerados de **0** a **N-1** onde **N** se refere ao número de elementos do array.



Num arranjo os elementos são acedidos pelos seus índices.

Que complicado... Valha-me D. Afonso Henriques!



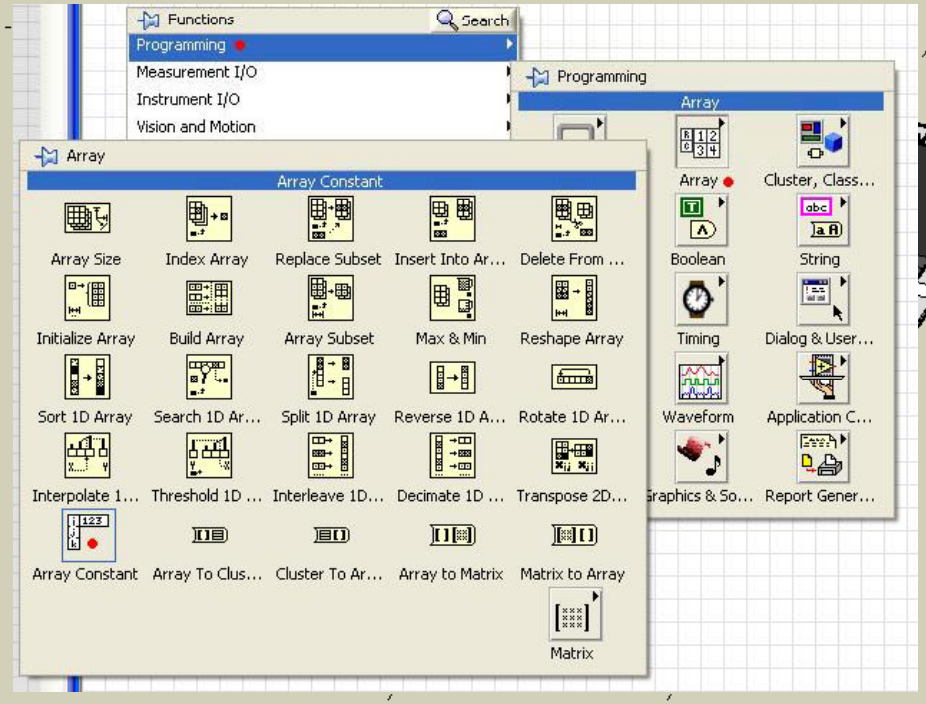
Formas de onda são normalmente armazenadas em arrays.

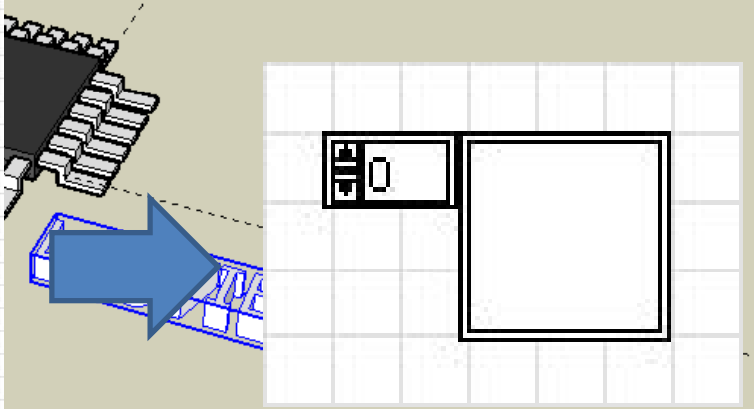
©2012 João Paulo Coelho

6.1 Arranjos (arrays)

Quando se cria um arranjo é necessário atribuir-lhe um tipo de dados. Assim devem efectuar-se dois passos:

1º Colocar o objecto ARRAY no diagrama de blocos





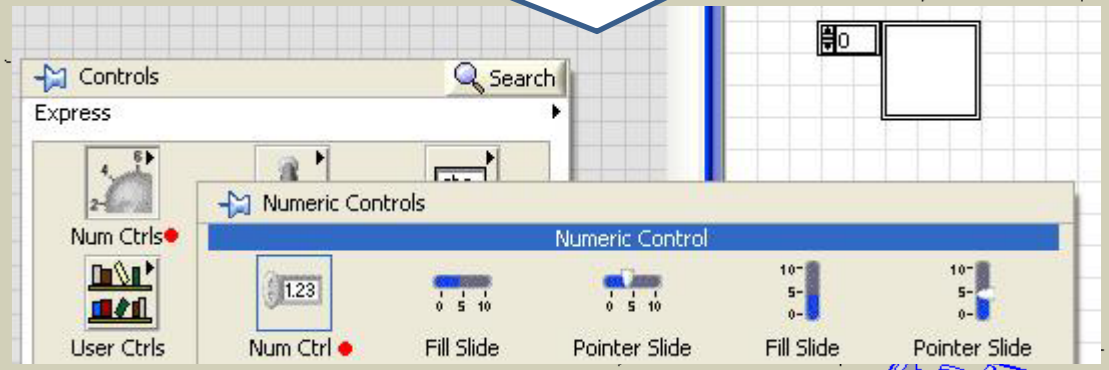
©2012 João Paulo Coelho

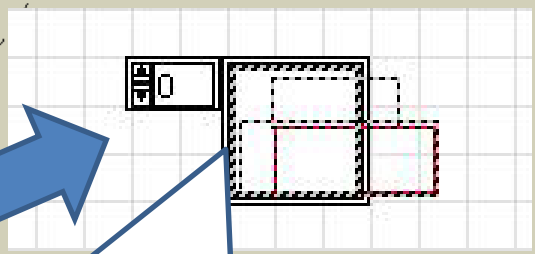
6.1 Arranjos (arrays)

2º Atribuir-lhe um tipo de DADOS:

Numérico:

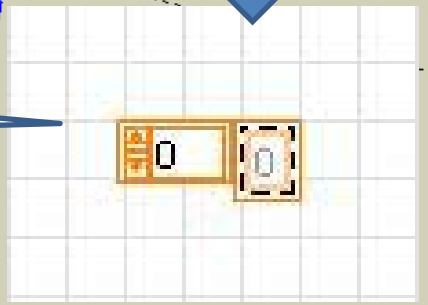
(a) Seleccionar controlo numérico





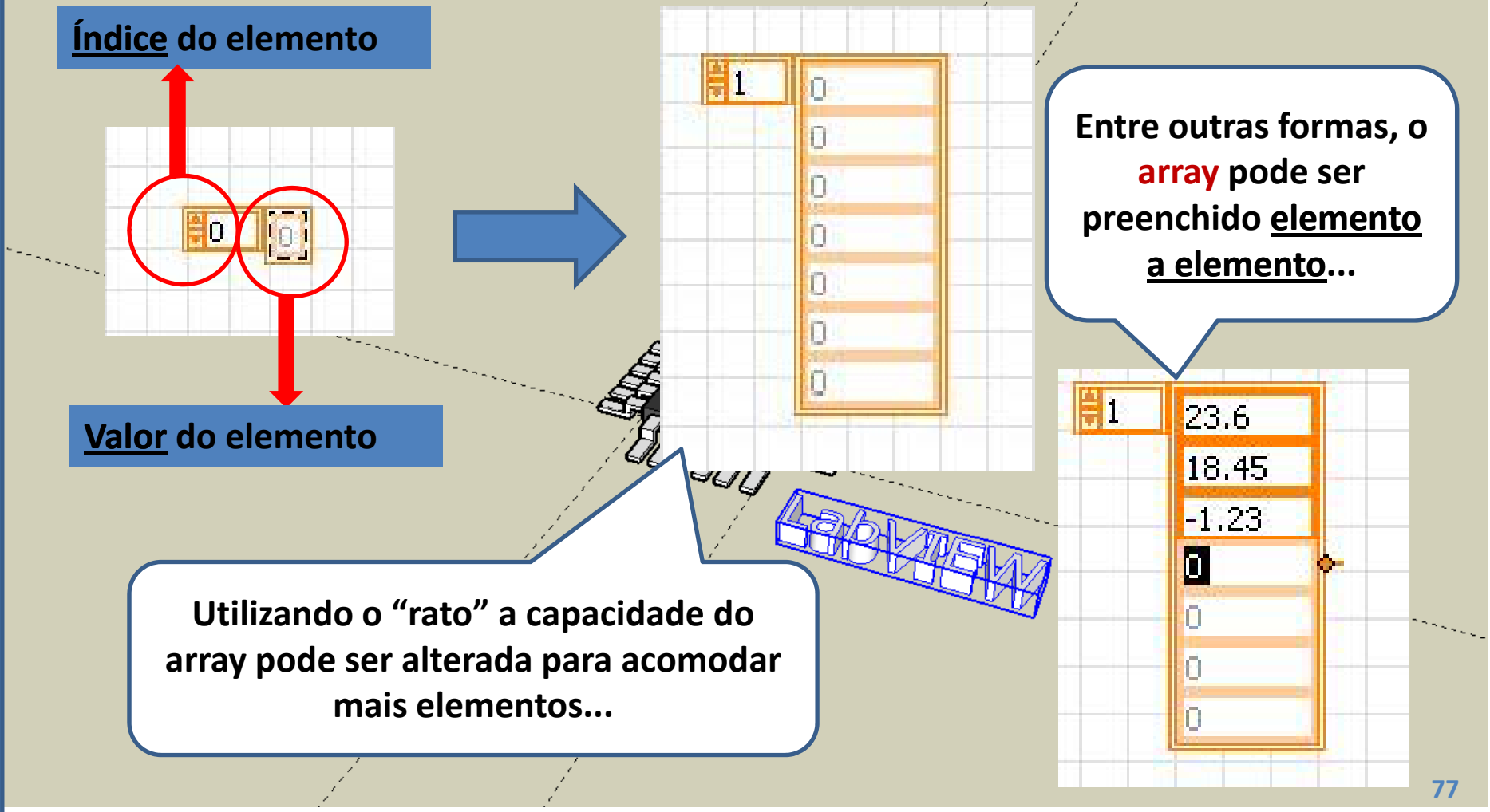
(b) "arrastá-lo" para dentro do objecto ARRAY

(c) Libertar o botão do "rato" e ... *volilá*



©2012 João Paulo Coelho

6.1 Arranjos (arrays)



©2012 João Paulo Coelho

6.1 Arranjos (arrays)

EXEMPLO: Apresentação do conteúdo de um vector num gráfico.

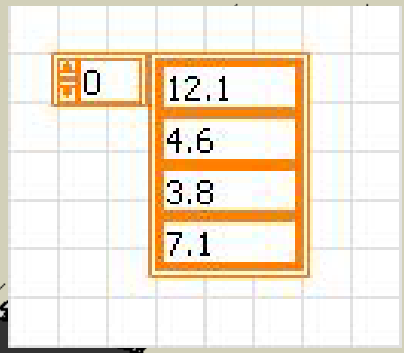
Afinal isto não tem nada que saber...

©2012 João Paulo Coelho

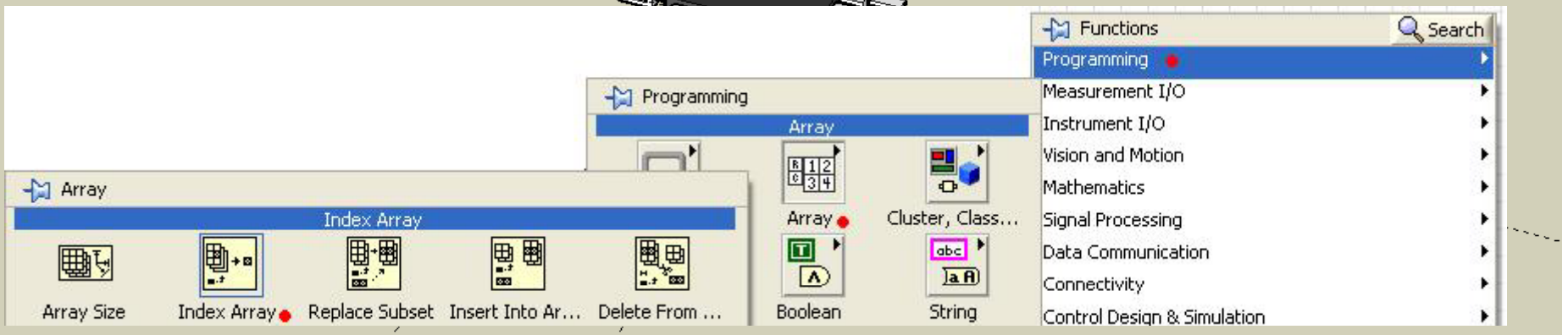
6.1 Arranjos (arrays)

EXEMPLO: Aceder a elementos num vector.

Introduza o seguinte ARRAY no diagrama de blocos...



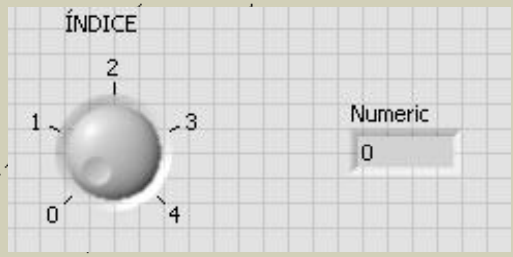
Acrescente o objecto "Index Array"

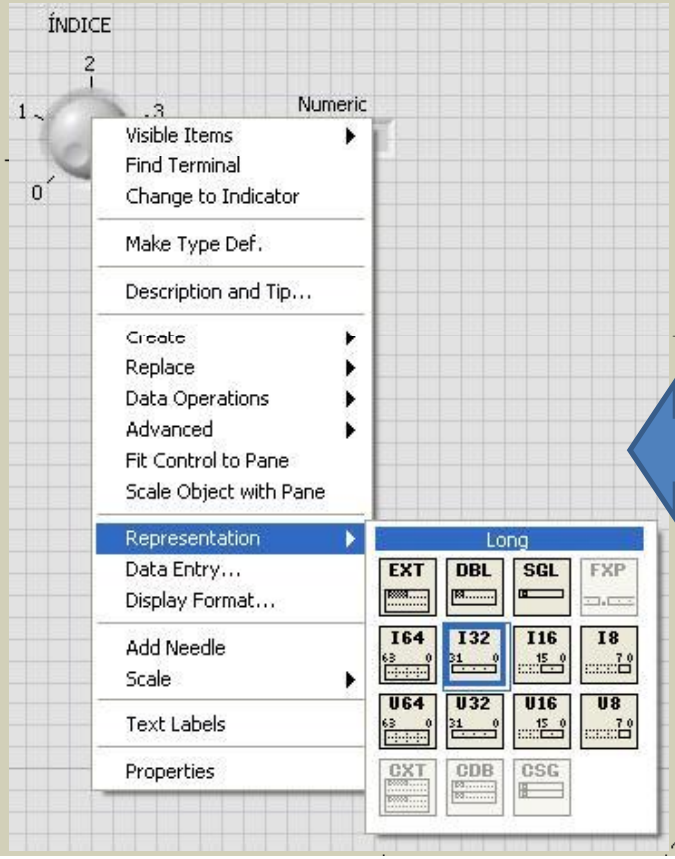


©2012 João Paulo Coelho

6.1 Arranjos (arrays)

Ao **Front Panel** acrescente um "**knob**" e um "**Numeric Ind.**"





O "knob" será responsável por definir o índice do elemento a aceder e o valor do elemento apontado por esse índice irá ser apresentado no "Numeric Indicator".

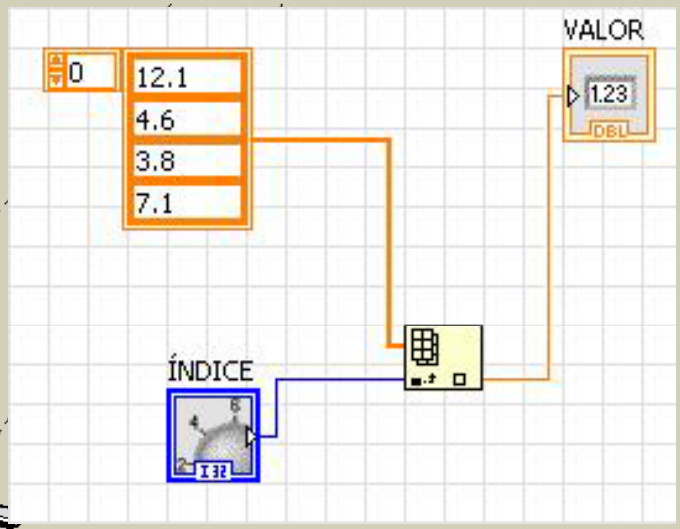
Como o índice deve ser um número inteiro vamos alterar o tipo de representação do "knob" para Inteiro 32 bits.

©2012 João Paulo Coelho

6.1 Arranjos (arrays)

Efectuar as seguintes ligações e testar.....

Questão: O que acontece quando se tenta aceder a um elemento do vector que ainda não foi definido?



Elementos que estão fora-de-jogo?

©2012 João Paulo Coelho

6.1 ARRAY de STRINGS

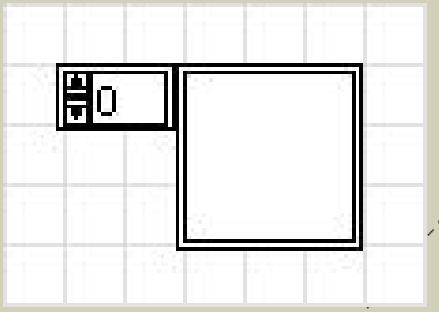
String:

É possível definir ARRAYS de outro tipo de dados ou até objectos.

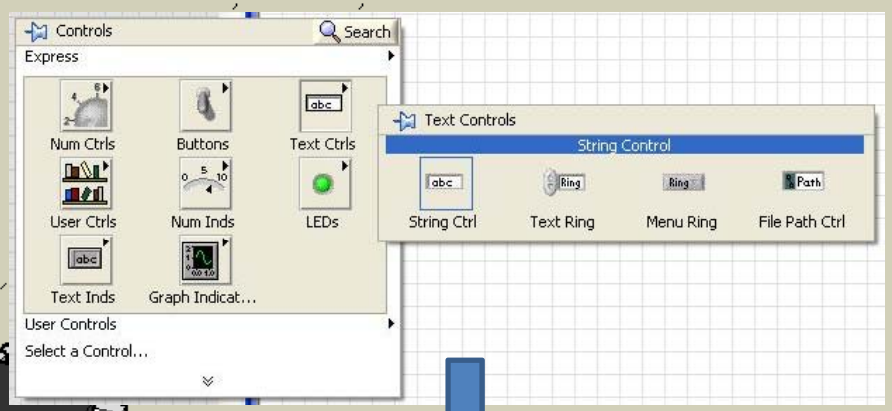
Um tipo muito útil é o ARRAY de STRINGS.

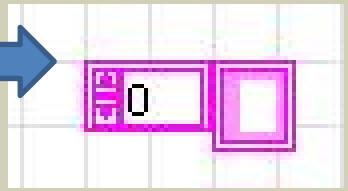
Criar um array de strings é semelhante à criação de um array numérico:

1º Colocar o objecto ARRAY no diagrama de blocos



2º Atribuir-lhe(1) o objecto "String Ctrl"





3º Preencher as posições pretendidas com caracteres alfanuméricos.

(1) "drag and drop" sobre o objecto ARRAY no diagrama de blocos.

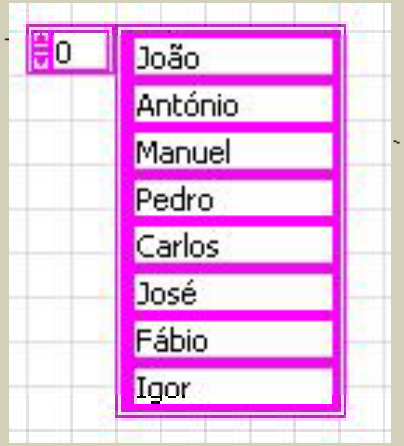
©2012 João Paulo Coelho

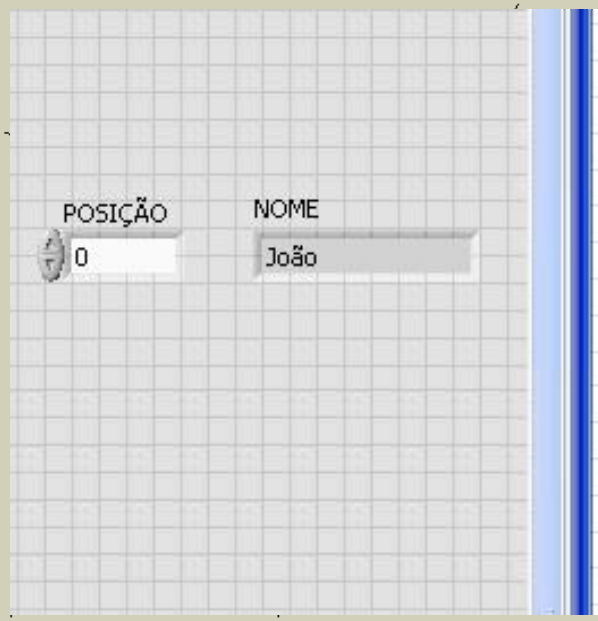
6.1 ARRAY de STRINGS

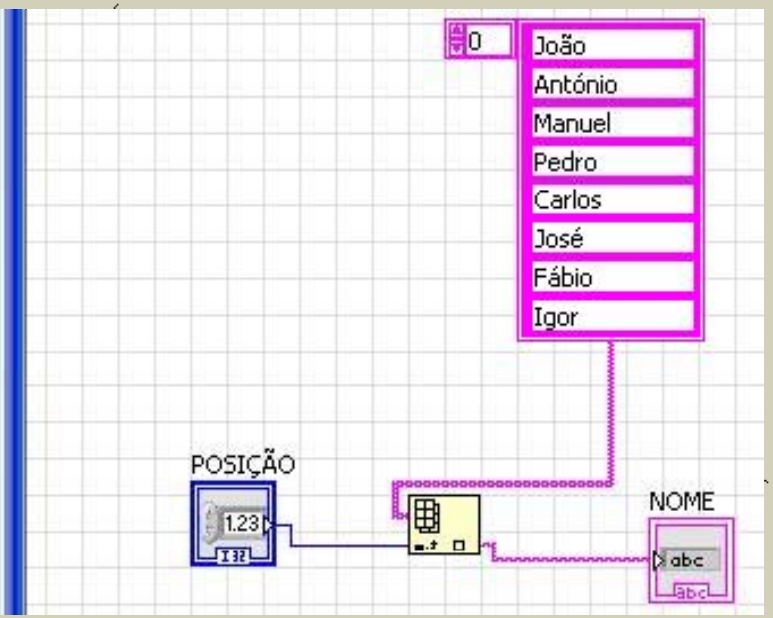
EXEMPLO:

1º Introduzir uma sequência de 8 nomes masculinos.

2º completar o programa de acordo com o esquema em baixo...







©2012 João Paulo Coelho

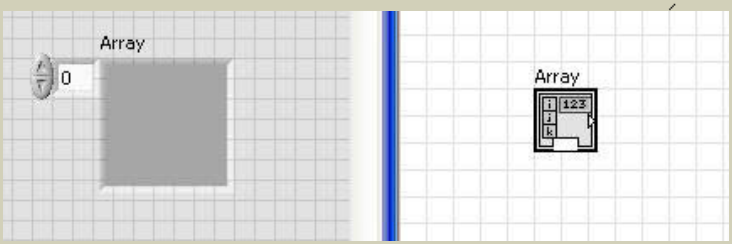
6.2 ARRAYS no FRONT PANEL

Até ao momento os ARRAYS foram considerados como CONSTANTES no diagrama de blocos.

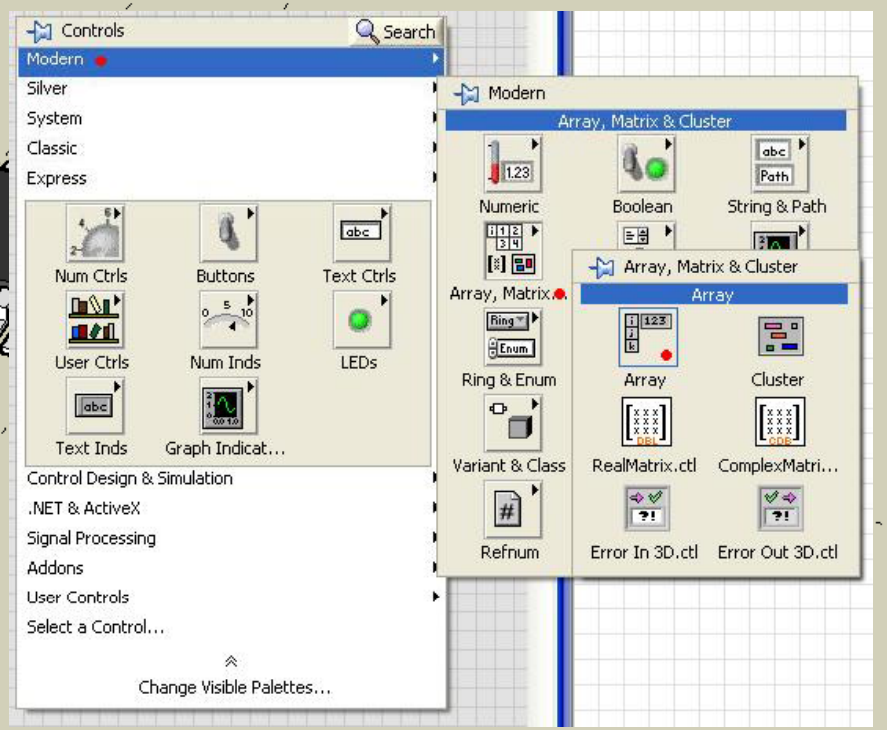
Alternativamente podem ser considerados OBJECTOS no FRONT PANEL

COMO?

No FRONT PANEL "clicar" com o botão direito do rato e aceder ao objecto ilustrado na figura ao lado...



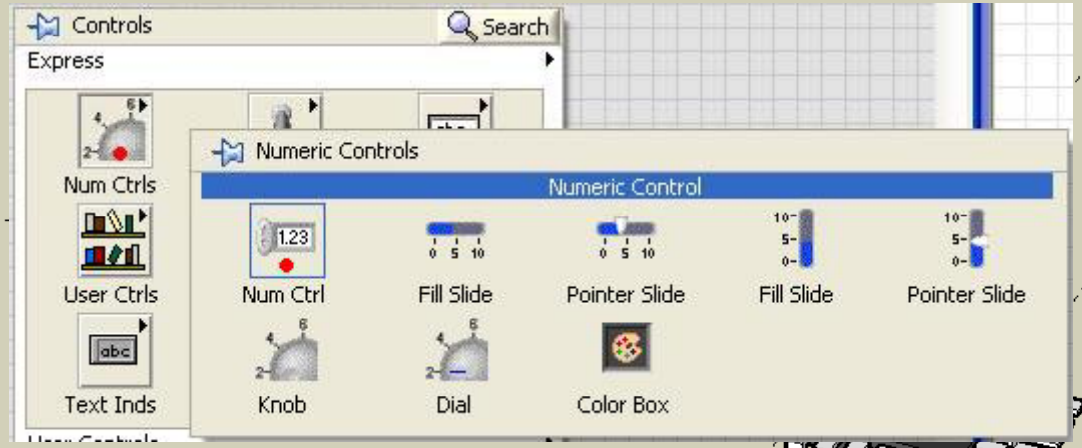
A cor PRETO indica que o ARRAY deve ser inicializado!

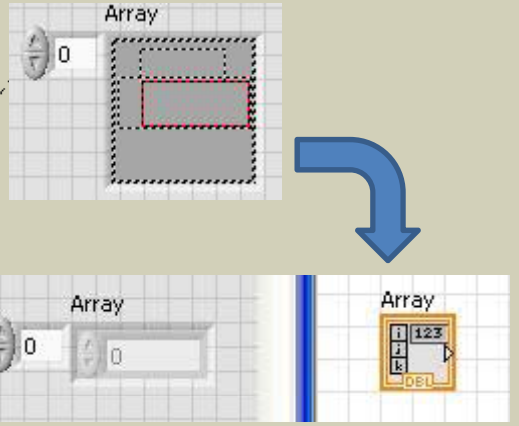


©2012 João Paulo Coelho

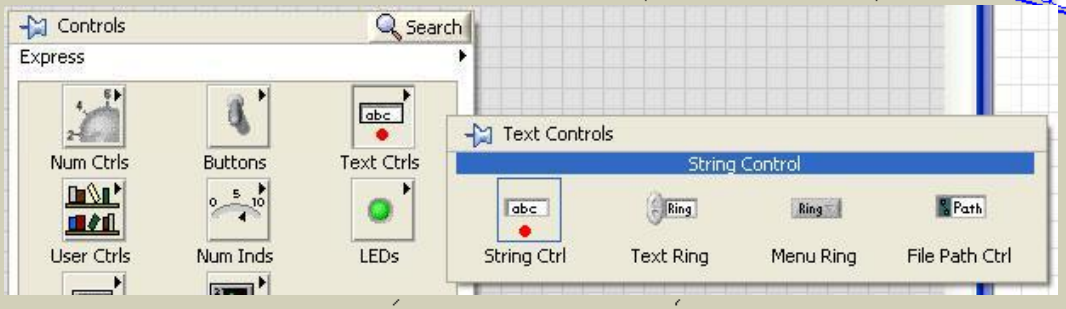
6.2 ARRAYS no FRONT PANEL

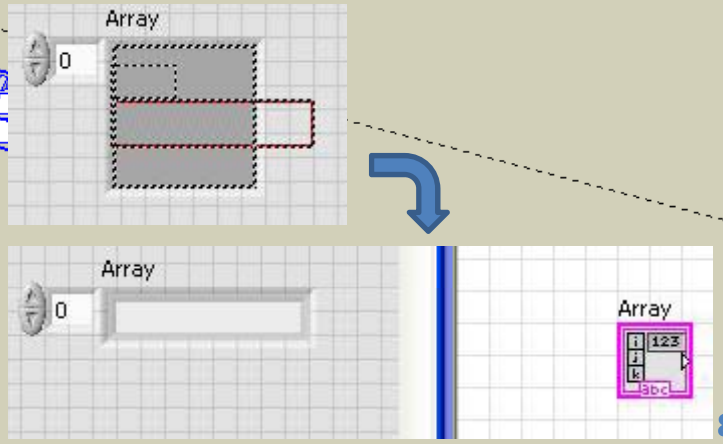
Inicialização com valores numéricos...





Inicialização com strings...

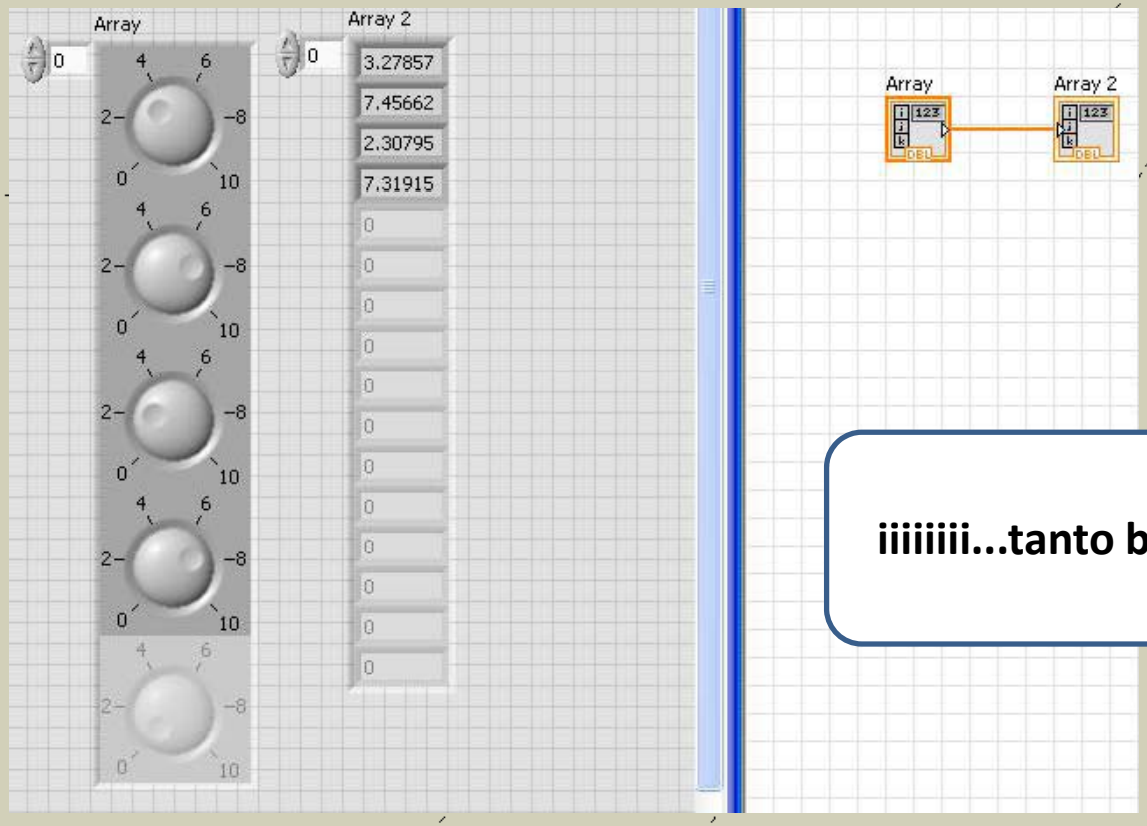




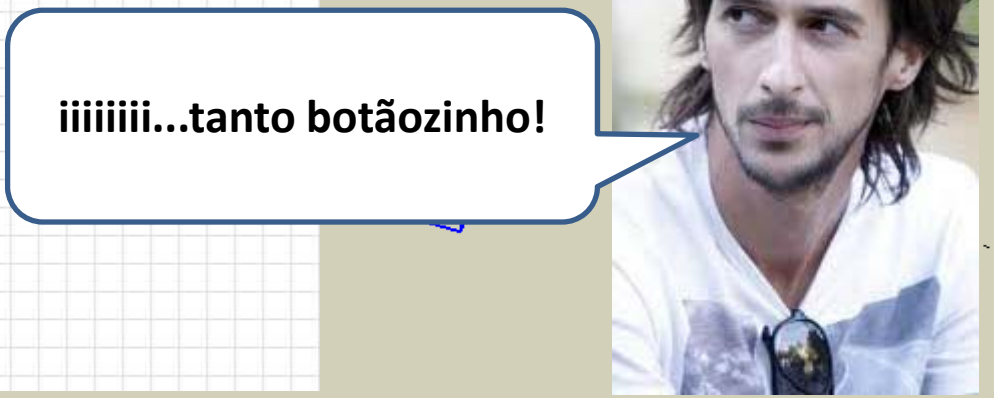
©2012 João Paulo Coelho

6.2 ARRAYS no FRONT PANEL

Inicialização com outros objectos...



EXEMPLO: Implemente e teste o seguinte programa...

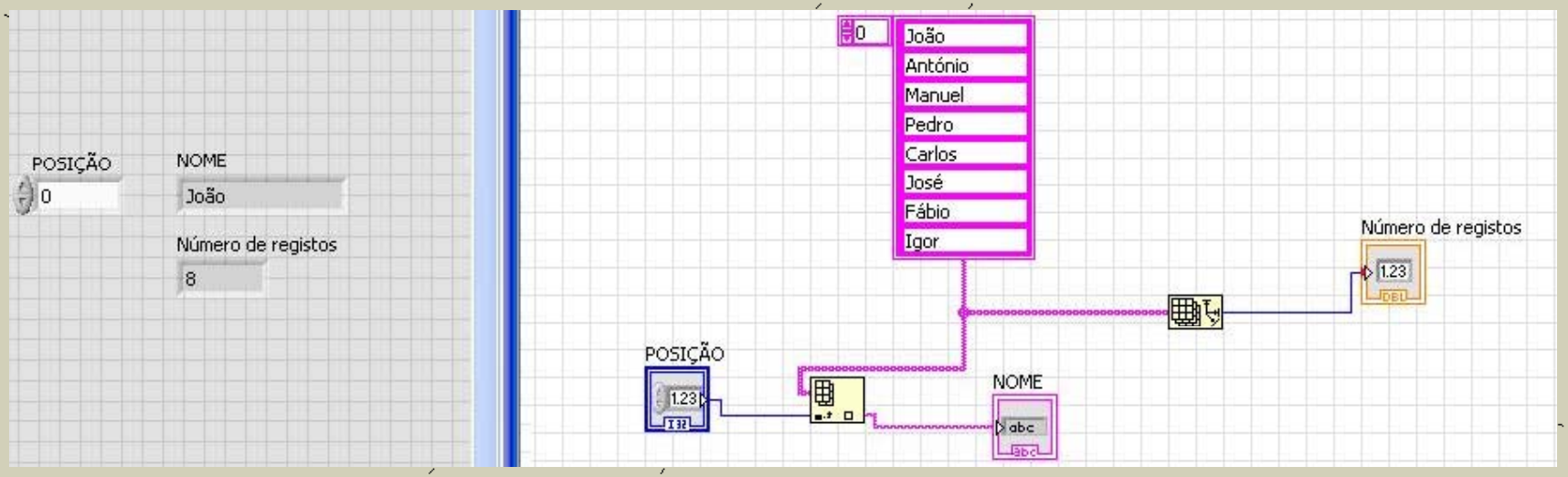


©2012 João Paulo Coelho

6.2 ARRAYS: operações

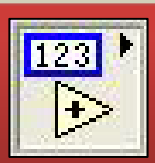
Algumas operações elementares que podem ser realizadas sobre ARRAYS:

NÚMERO DE ELEMENTOS: retorna o número de elementos de um ARRAY

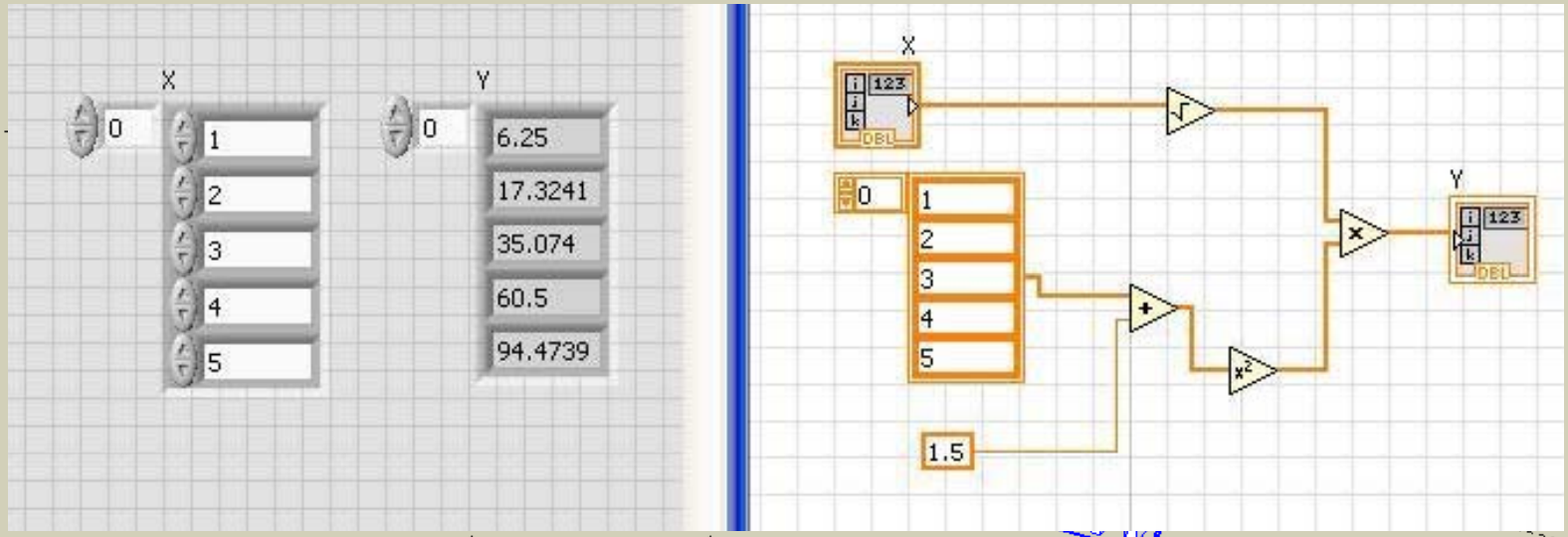


©2012 João Paulo Coelho

6.2 ARRAYS: operações



ARITMÉTICA ENTRE ARRAYS NUMÉRICOS

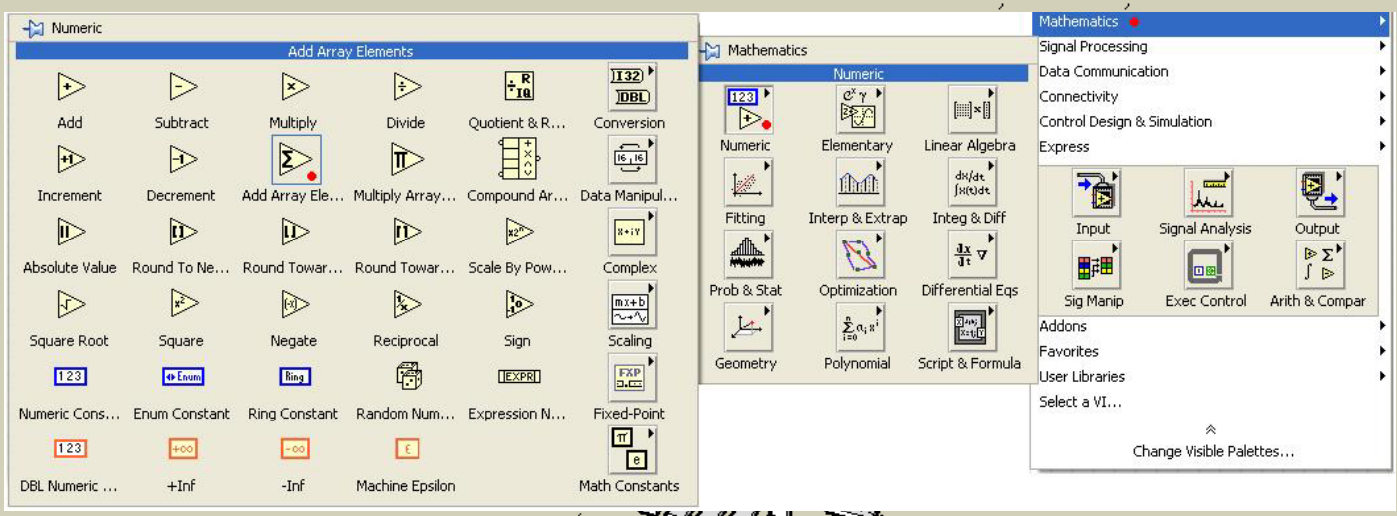


Exemplo: Calcular o valor médio de um vector

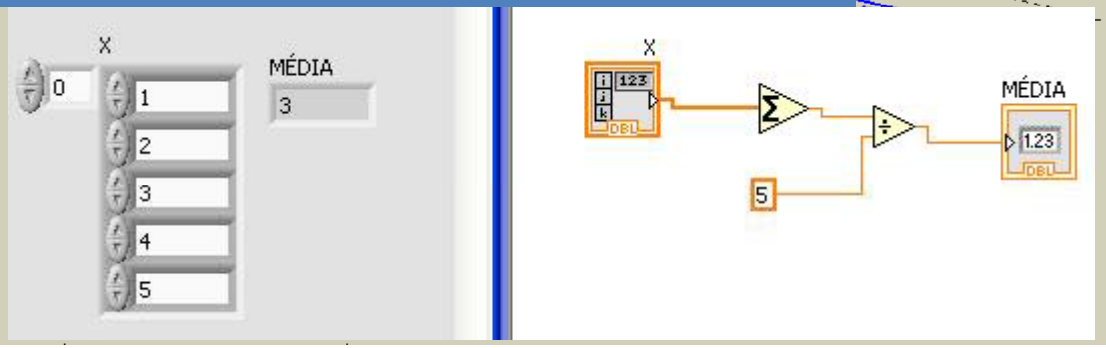
©2012 João Paulo Coelho

6.2 ARRAYS: operações

1º Aceder à operação "ADD ARRAY ELEMENTS"



2º Efectuar as ligações...

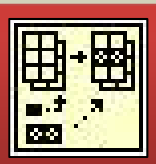


©2012 João Paulo Coelho

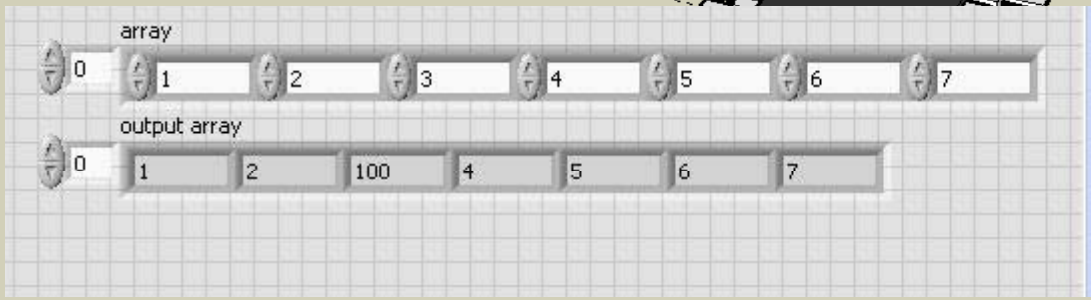
6.2 ARRAYS: operações

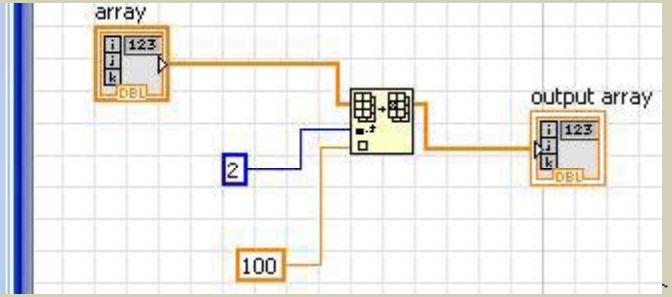
EXERCÍCIO: Modifique o programa anterior de modo a poder ser calculado automaticamente a média de um ARRAY de dimensão arbitrária

EXERCÍCIO: Desenhe um programa capaz de calcular o desvio padrão de um ARRAY



Alterar um elemento

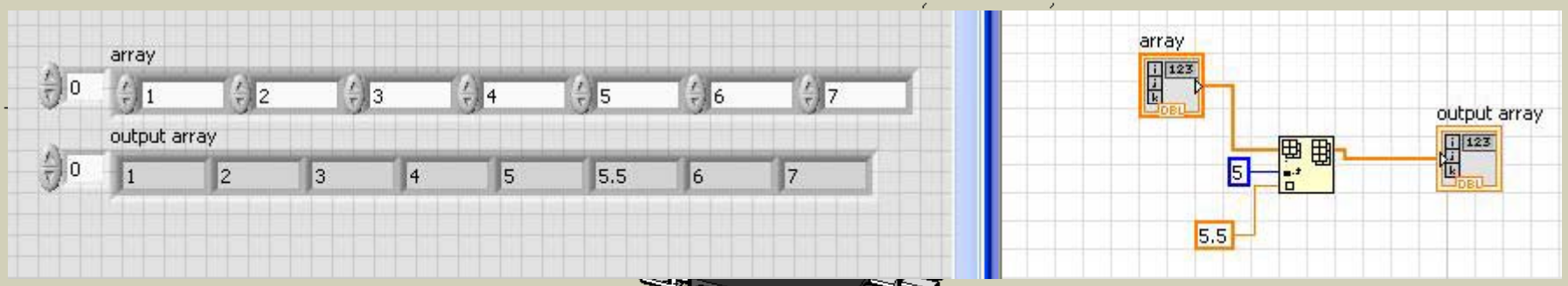




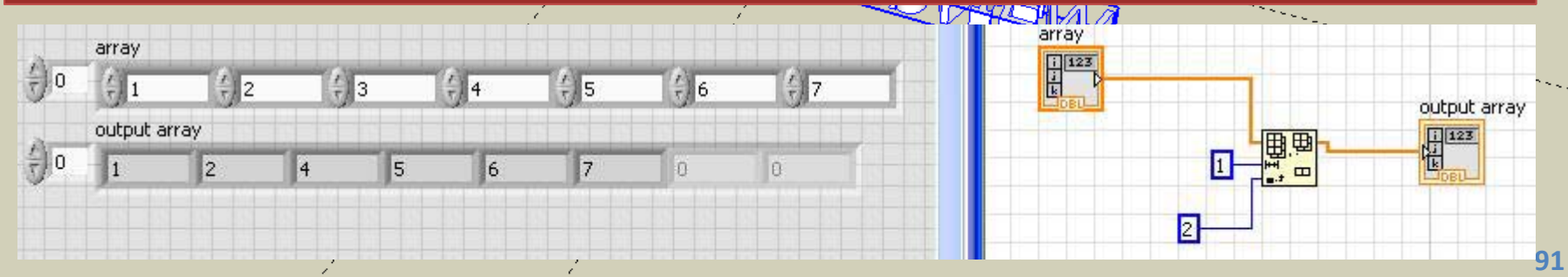
©2012 João Paulo Coelho

6.2 ARRAYS: operações

Adicionar um elemento



Remover um elemento



©2012 João Paulo Coelho

7.0 SUBVI's

O carácter modular do LabVIEW é potenciado com o conceito de SUBVI

Um SUBVI é um VI (instrumento virtual) utilizado "dentro" de outro VI

A utilização de SUBVI permite:

- **Tornar mais fácil a leitura e a depuração de um programa;**
- **Possibilita a reutilização de elementos de código já desenvolvidos.**

Será que vou conseguir entender esta estória de SUBVI's?

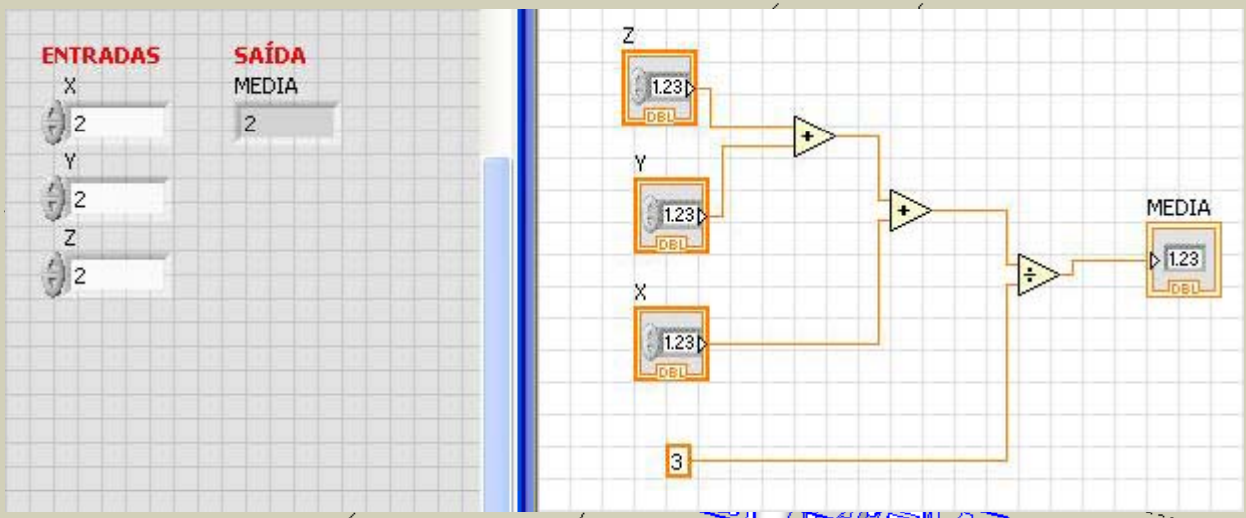
VI's podem ser "aninhados" em número arbitrário (VI's dentro de VI's que já envolvem outros VI's ...)

©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

EXEMPLO: SUBVI capaz de calcular a média de três valores.

1º Construir o programa...

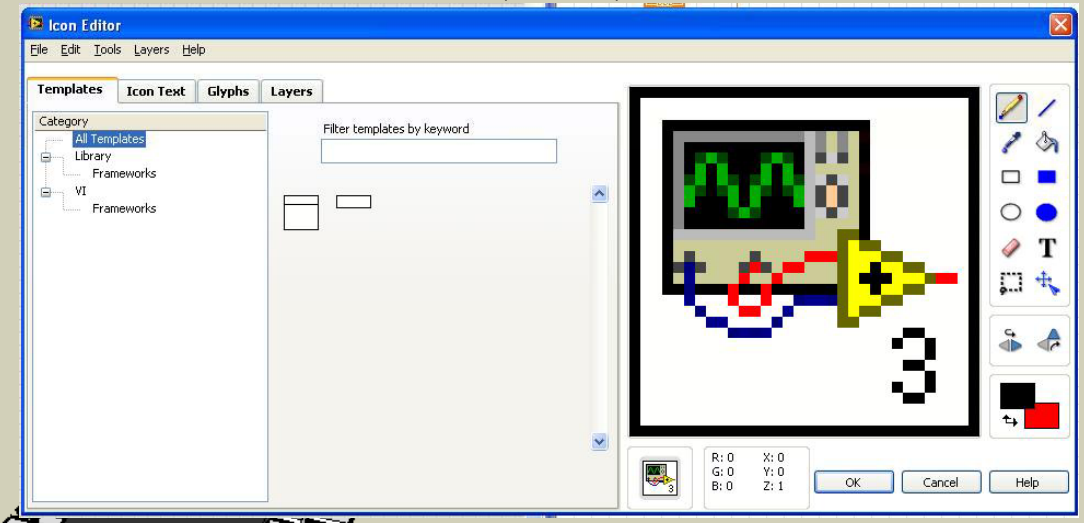


2º (opcional) Atribuir um ícone ao programa...

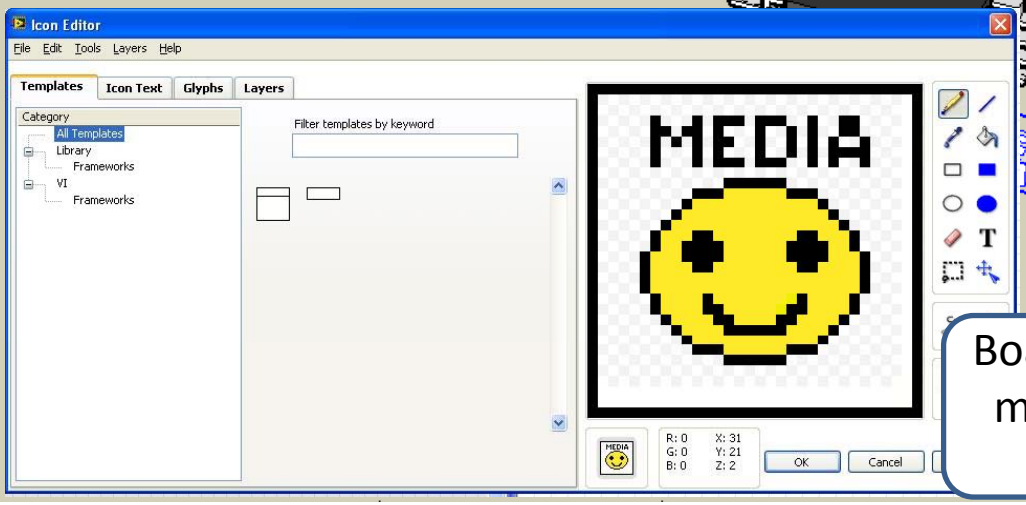
©2012 João Paulo Coelho

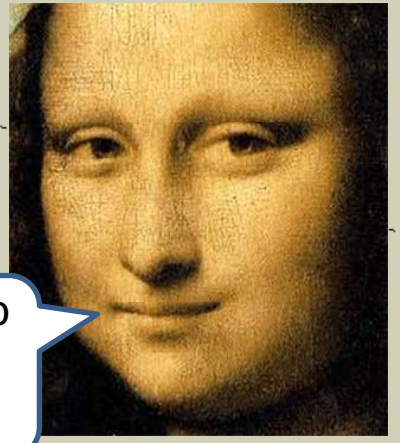
7.1 Construção de **SUBVI's**

Programa de desenho de ícones...



Uma proposta sorridente...





Boa tentativa...mas o meu sorriso é mais gracioso....

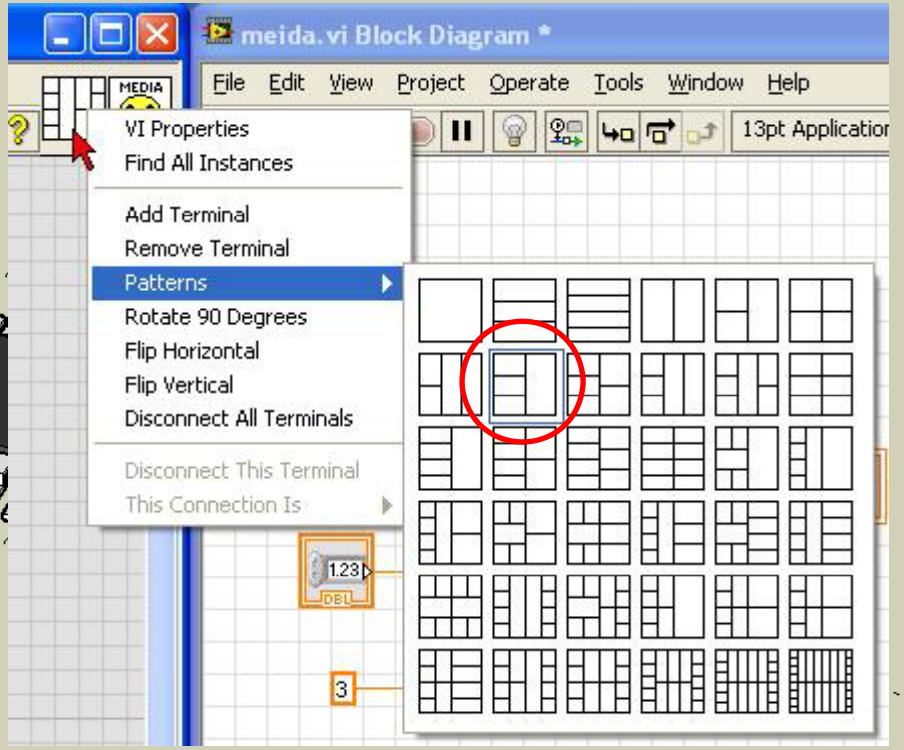
©2012 João Paulo Coelho

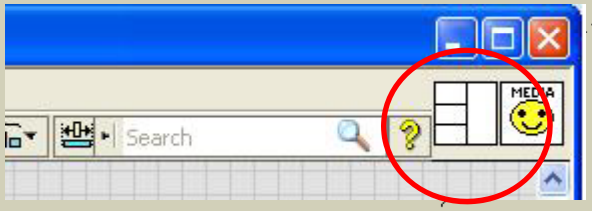
7.1 Construção de **SUBVI's**

3º atribuir sinais de entrada e saída...

Seleccionar no "connector pane" a topologia que mais se adapta ao programa.

Por convenção os terminais mais à esquerda são associados a "entradas" e os terminais mais à direita são associados às "saídas"





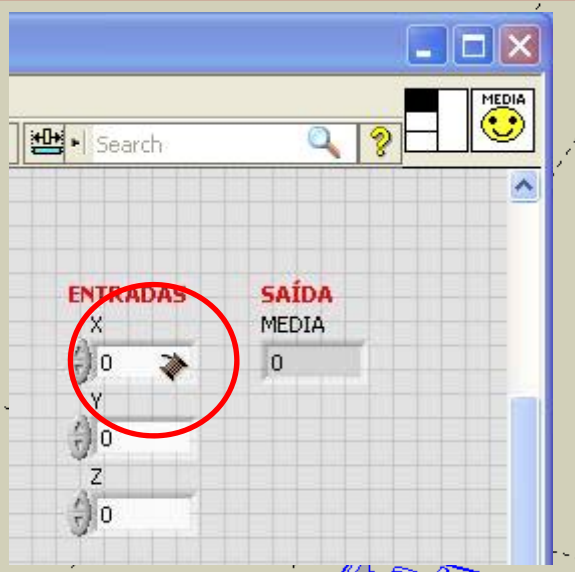
É normalmente considerada boa prática escolher um padrão de terminais que possua mais pares entrada/saída do que as necessárias de forma a contemplar possíveis alterações no SUBVI.

©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

Atribuir terminais aos "controlos" e "indicadores"

seleccionar no "**connector pane**" um dos terminais disponíveis (o terminal passa de "branco" para "preto")

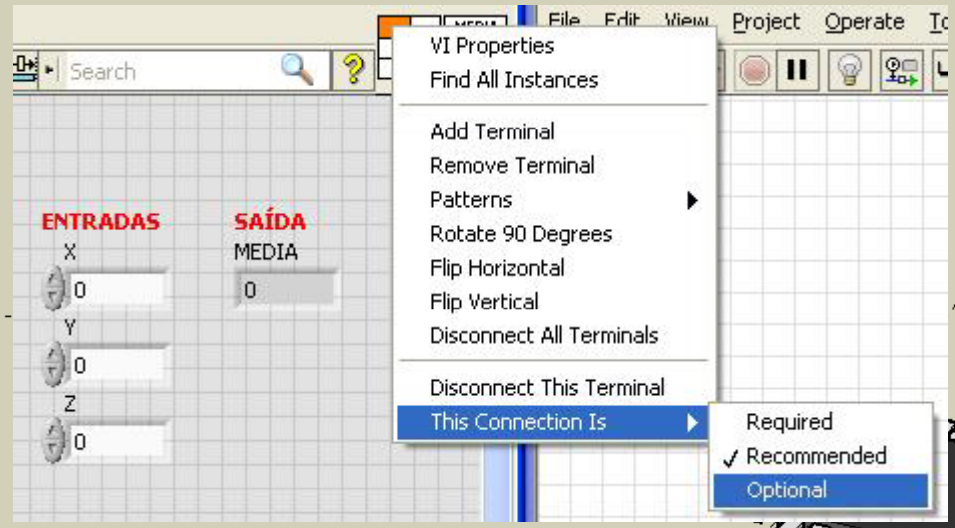


seleccionar no "**Front Panel**" qual o indicador/controlo a associar a esse terminal

O terminal recentemente associado passa a apresentar a cor laranja.

©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**



Cada terminal associado pode ter um de três modos distintos:

- **Necessário**
- **Recomendado**
- **Opcional**

Por defeito o valor é recomendado...

Aspecto do "connector pane" após todas as entradas e saídas serem associadas a um terminal...

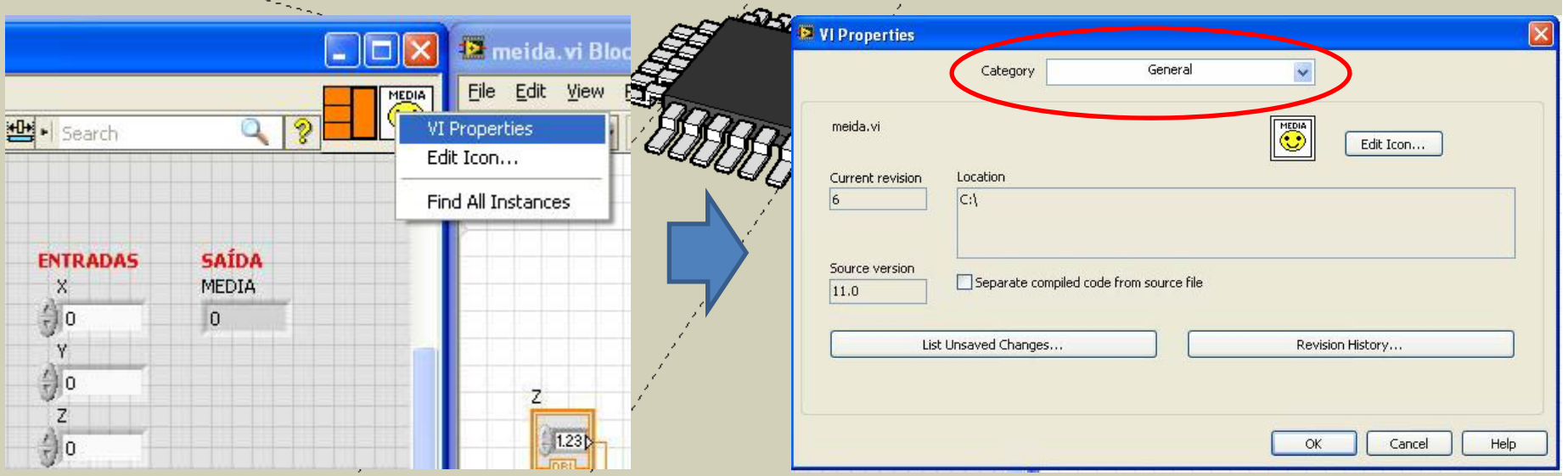
©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

4º (opcional mas recomendado) documentar o programa ...

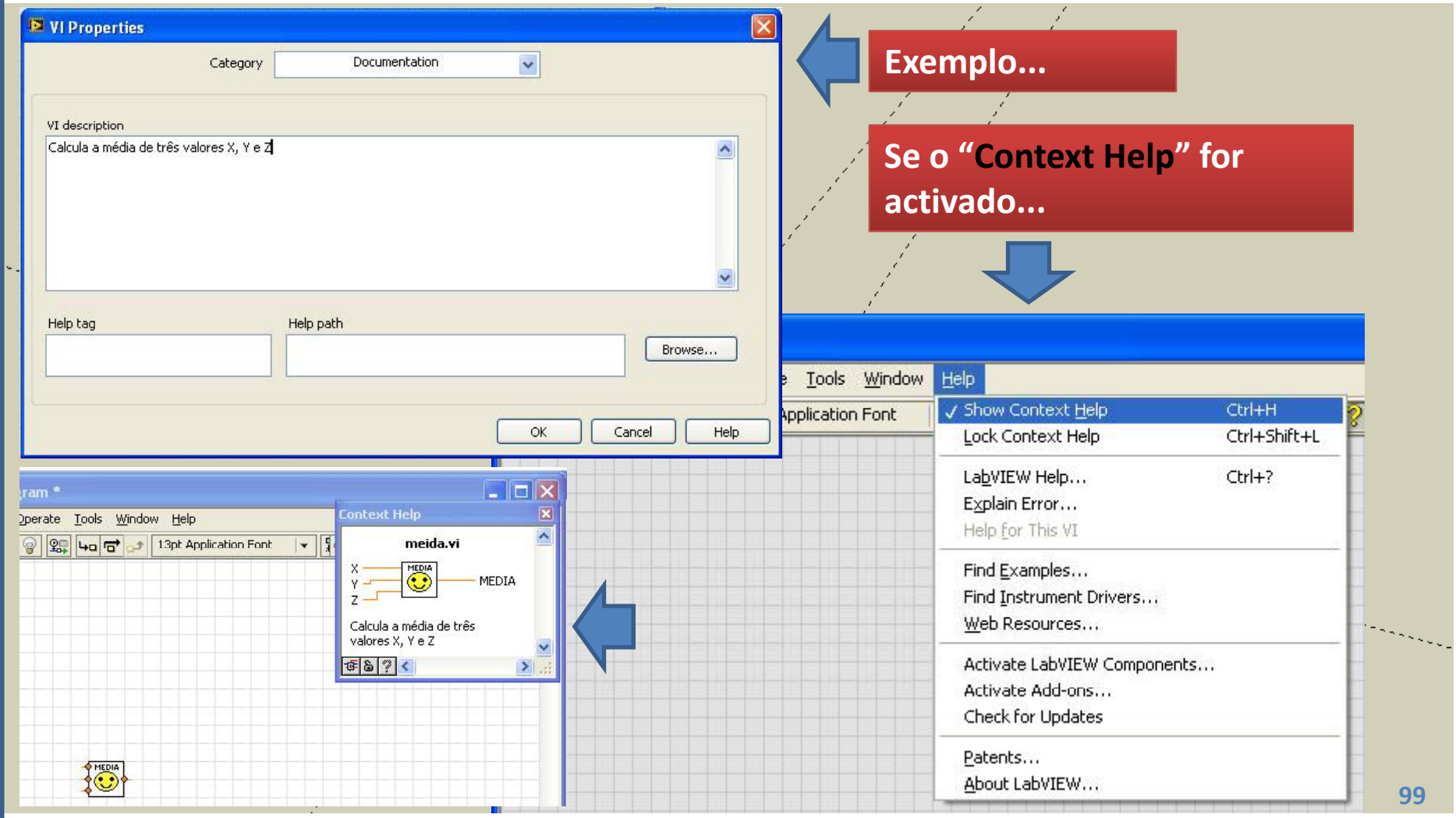
"click" no botão direito do rato sobre "connector pane" ou sobre o ícone e seleccionar "VI Properties"

Na janela "VI Properties" escolher em Category o item "Documentation"



©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

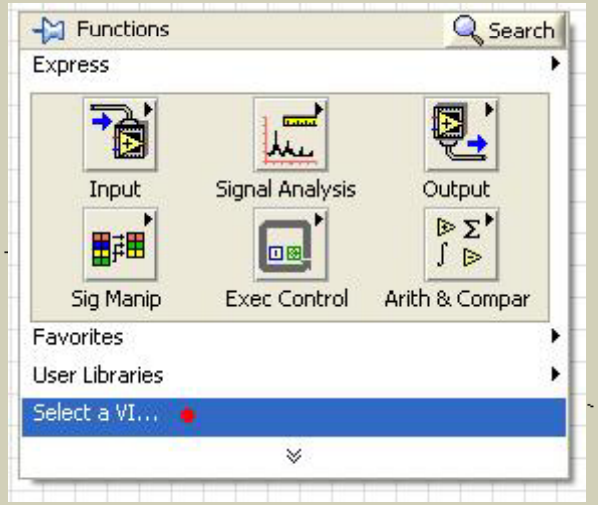


©2012 João Paulo Coelho

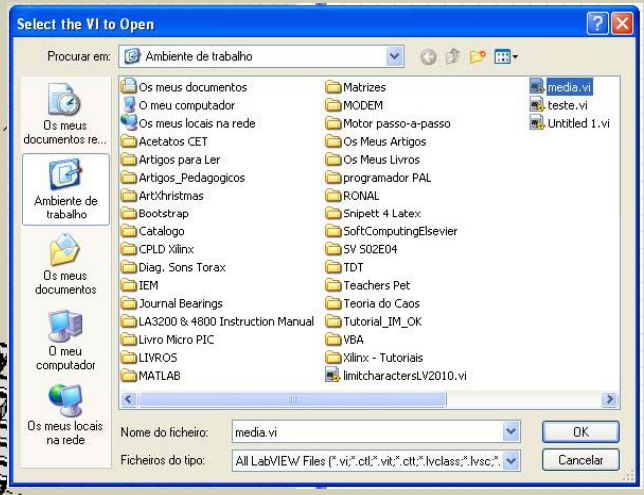
7.1 Construção de **SUBVI's**

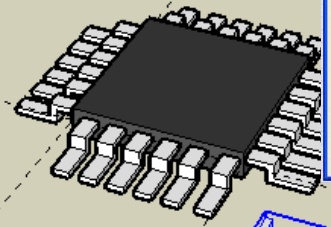
Inserir e utilizar um SUBVI num VI:

1º

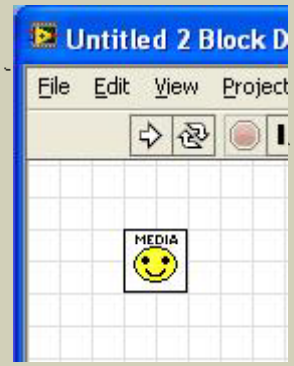


2º

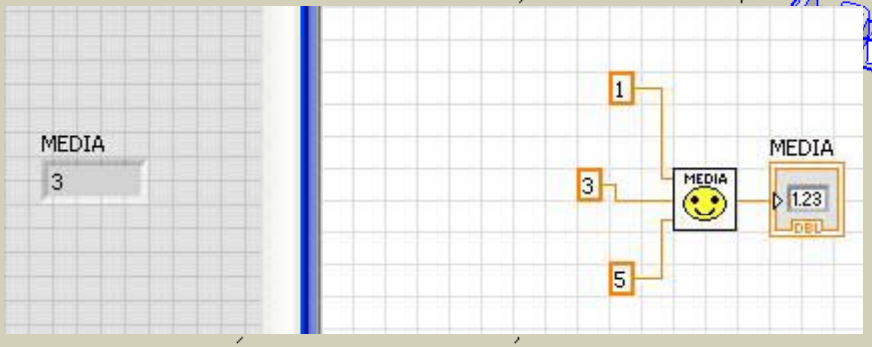




3º



4º

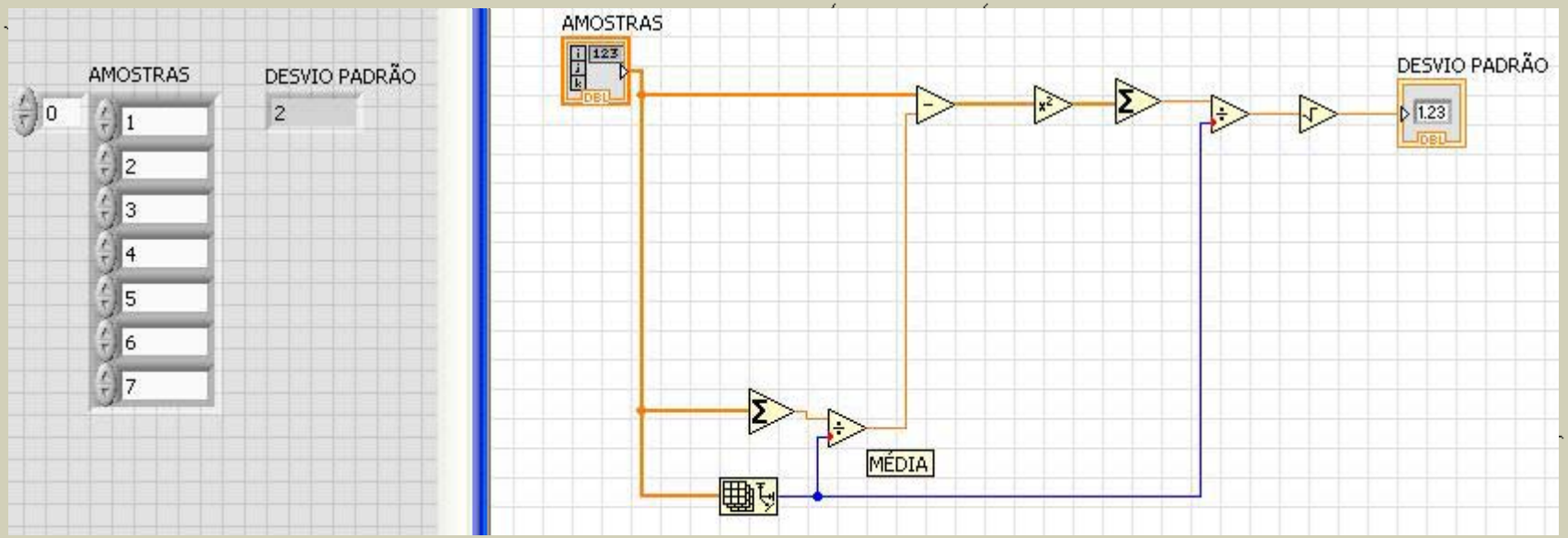


©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

Criar um **SUBVI** a partir de um excerto de um digrama de blocos...

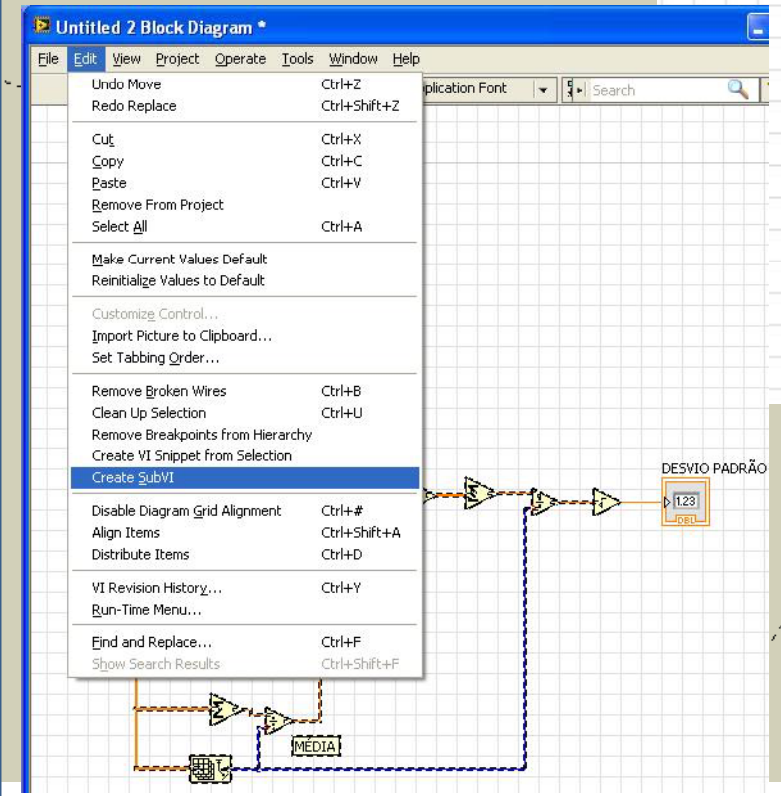
Considerar o seguinte VI...

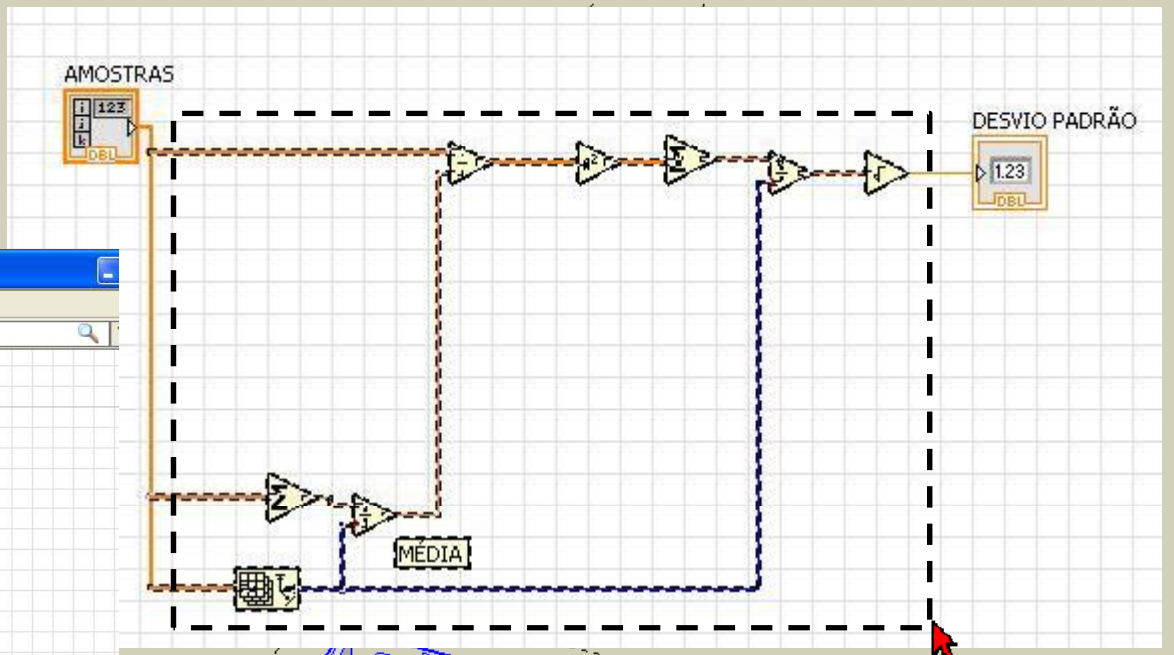


©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

Seleccionar a fracção do diagrama de blocos que se pretende substituir.



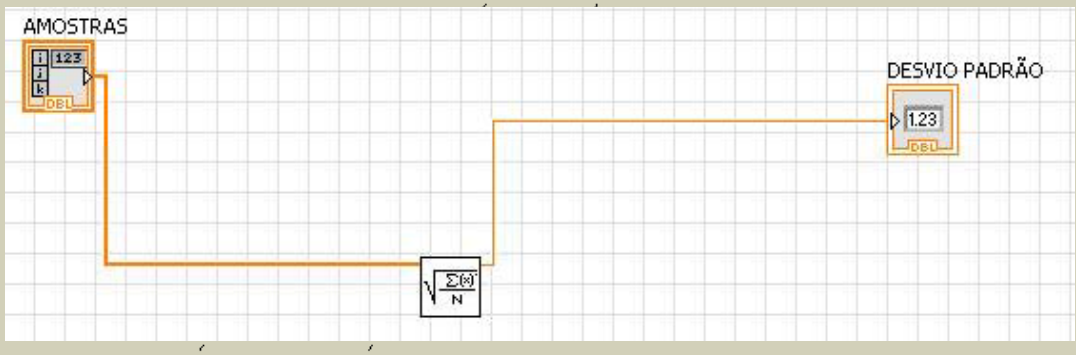


No menu "Edit" escolher "Create SubVI".

©2012 João Paulo Coelho

7.1 Construção de **SUBVI's**

Aspecto final já com novo ícone.



Deo Gratias... Estou sem palavras....

©2012 João Paulo Coelho

7.2 Exercícios

***EX 13**: Construa um SubVI capaz de calcular as quatro operações aritméticas elementares entre dois vectores (o produto e a divisão dizem respeito ao produto de Hadamard entre os dois vectores ou ente um dos vectores e outro obtido a partir do inverso dos elementos do segundo)*

***EX 14**: Utilize o SubVI desenhado anteriormente para calcular a seguinte sequência de operações entre os vectores V1, V2 e V3:*

*(V1+V2*V3)/(V4+1)*

EX 15**: Construa um SubVI capaz de gerar um número **inteiro** aleatório entre **1 e 49

©2012 João Paulo Coelho

8.0 Estruturas de Controlo

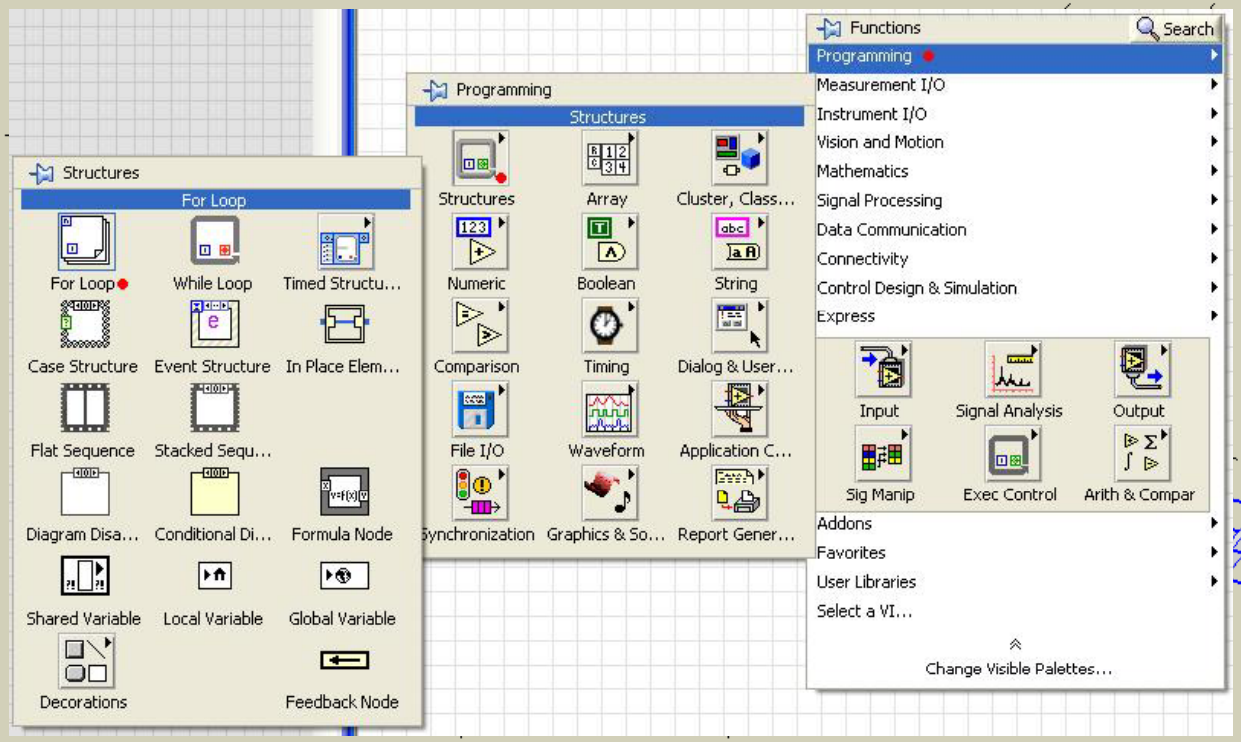
- Para além dos controlos, indicadores e VI, um diagrama de blocos pode também conter sequências para controlo de fluxo de dados.
- O LabVIEW suporta, entre outras, as seguintes estruturas de controlo:
  - Ciclos **FOR**
  - Ciclos **WHILE**
  - Estruturas condicionais **CASE**

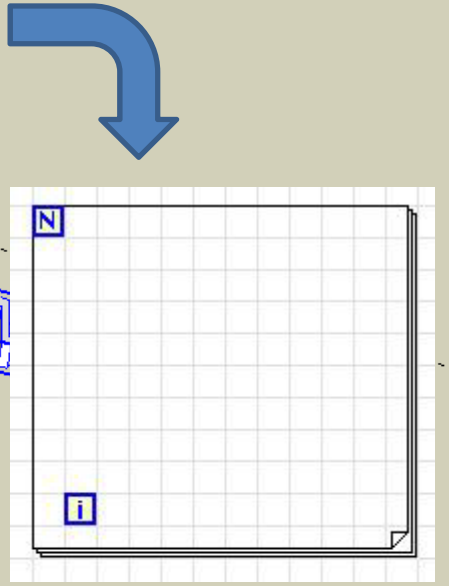
Afinal o LabVIEW é que é o *SPECIAL ONE*

©2012 João Paulo Coelho

8.1 Ciclos **FOR**

Utilizam-se sempre que se conhece, à partida, o número de ciclos a executar.





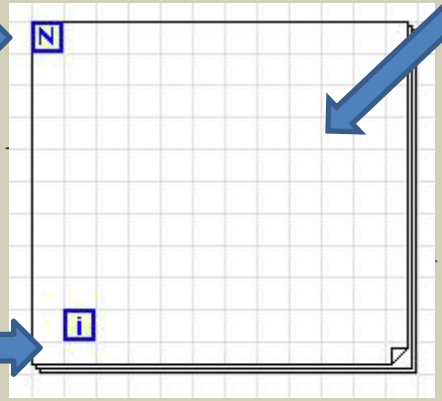
©2012 João Paulo Coelho

8.1 Ciclos **FOR**

Um ciclo **FOR** possui dois terminais:

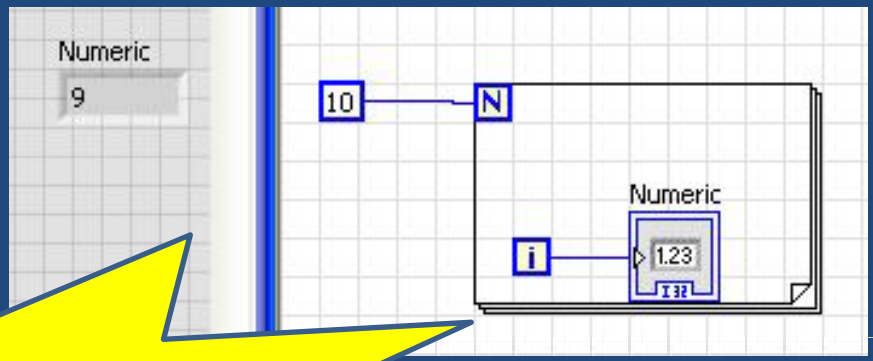
Terminal de contagem.

Código que deve ser executado repetidamente (subdiagrama).



Terminal de iteração.

Exemplo:



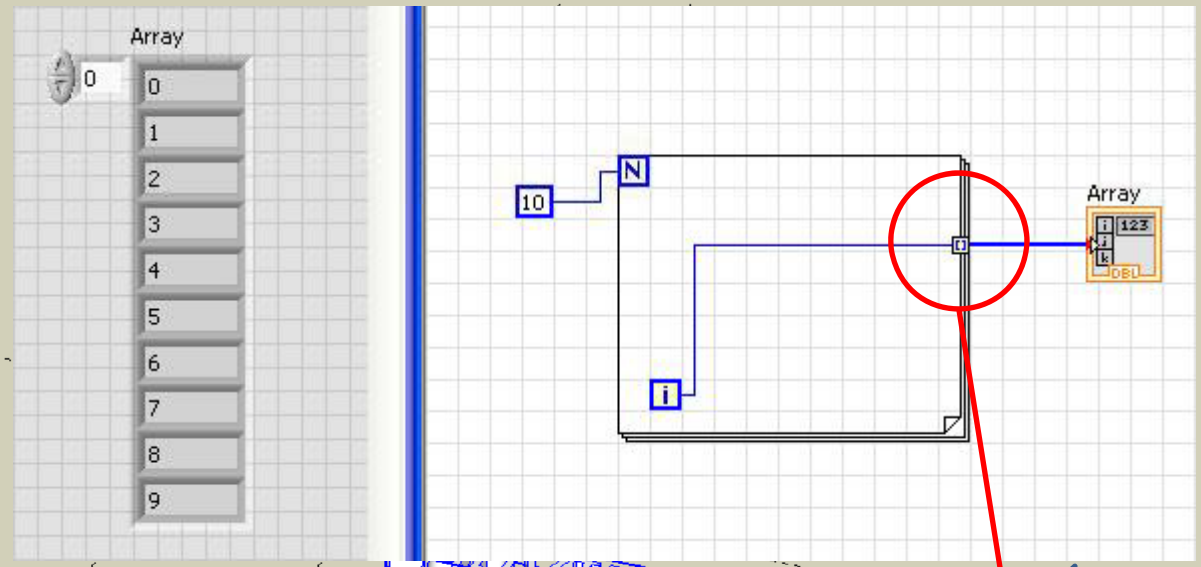
Execute com Highlight Execution ligado

©2012 João Paulo Coelho

8.1 Ciclos **FOR**: túneis

O ciclo FOR é frequentemente utilizado para inicializar ARRAYS.

O programa indicado ao lado preenche um ARRAY com os números inteiros de 0 a 9.

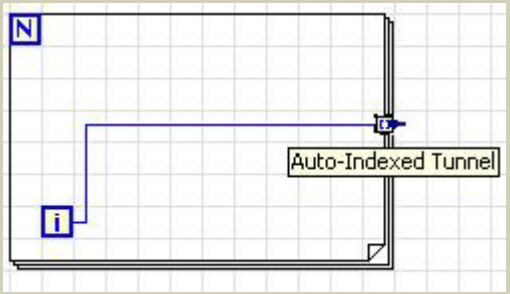


Os túneis permitem a passagem de informação através do ciclo.

Um túnel pode possuir a indexação automática habilitada ou não.

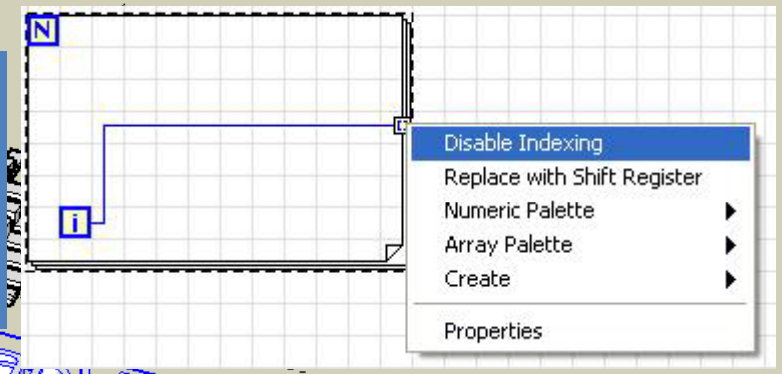
©2012 João Paulo Coelho

8.1 Ciclos **FOR**: túneis

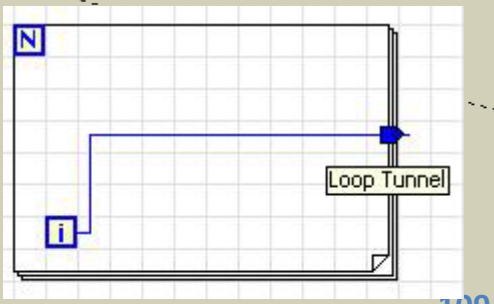


Quando a **indexação automática** se encontra **activa** o valor calculado num conector é enviado através do túnel em cada iteração do ciclo.

Quando a indexação automática se encontra **inactiva** o valor calculado num conector é enviado através do túnel apenas após a conclusão do ciclo.



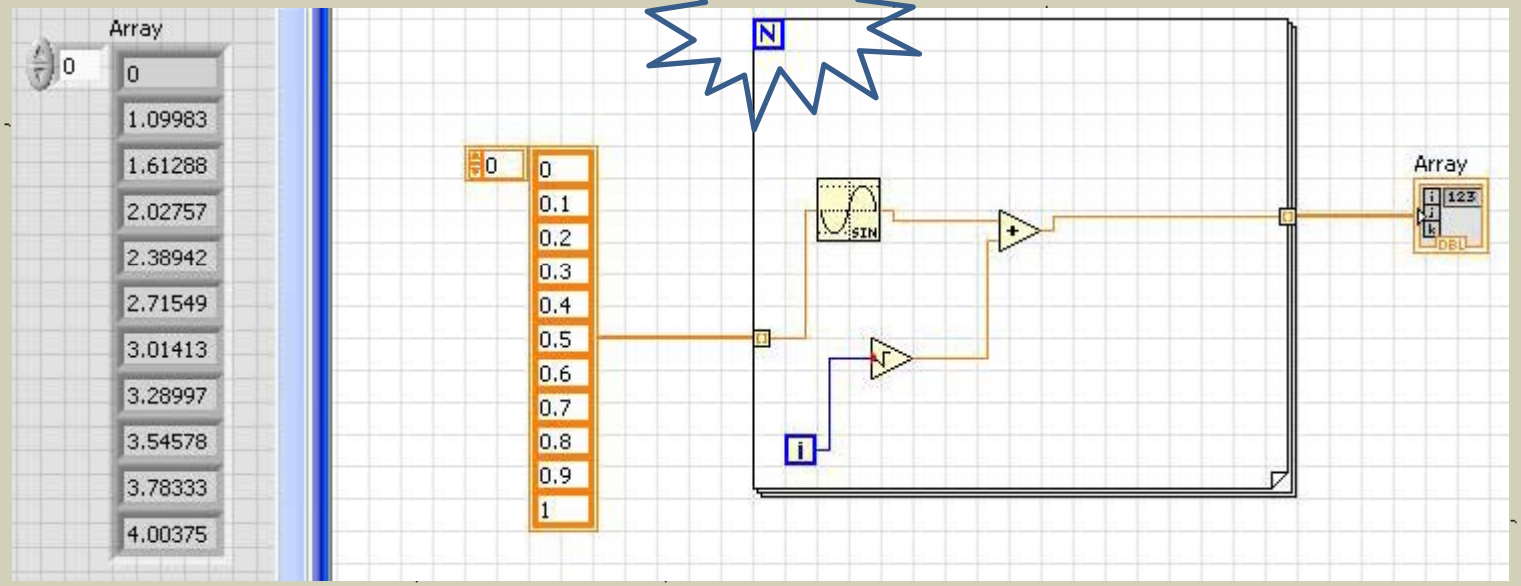
Na primeira situação a saída do túnel é um **ARRAY** e na segunda a saída é um **ESCALAR**.



©2012 João Paulo Coelho

8.1 Ciclos **FOR**: túneis

O valor de N (número de iterações do ciclo) pode não ser explicitado no caso de se enviar dados, por túnel, para o interior do ciclo.



Quando a indexação se encontra ACTIVA no túnel de entrada, o valor de N é admitido como sendo igual ao número de elementos do ARRAY.

©2012 João Paulo Coelho

8.1 Exercícios

***EX 16**: Construa um VI capaz de preencher um vector com os primeiros 100 valores da função:*

$$f(i)=\sin(pi*i/100)$$

onde i é um inteiro de 0 a 99

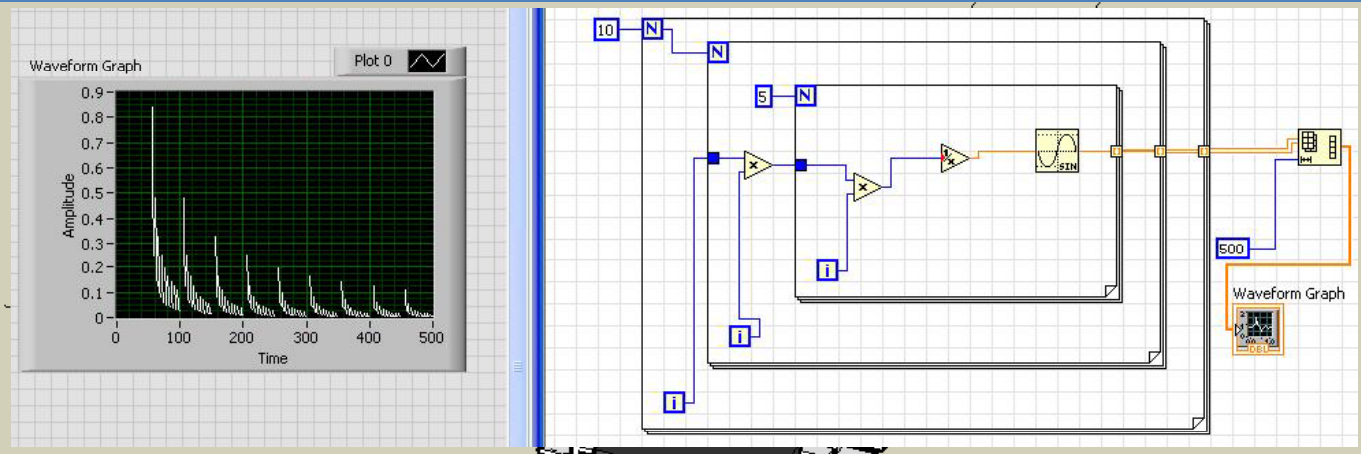
***EX 17**: Repita o mesmo exercício mas agora o valor i deve começar em -5. Para além disso um gráfico da função deve ser traçado.*

***EX 18**: Construa um VI que preencha um ARRAY numérico com 10 valores aleatórios inteiros entre 1 e 49. (sugestão: utilize o subVI que desenvolveu para o exercício 15)*

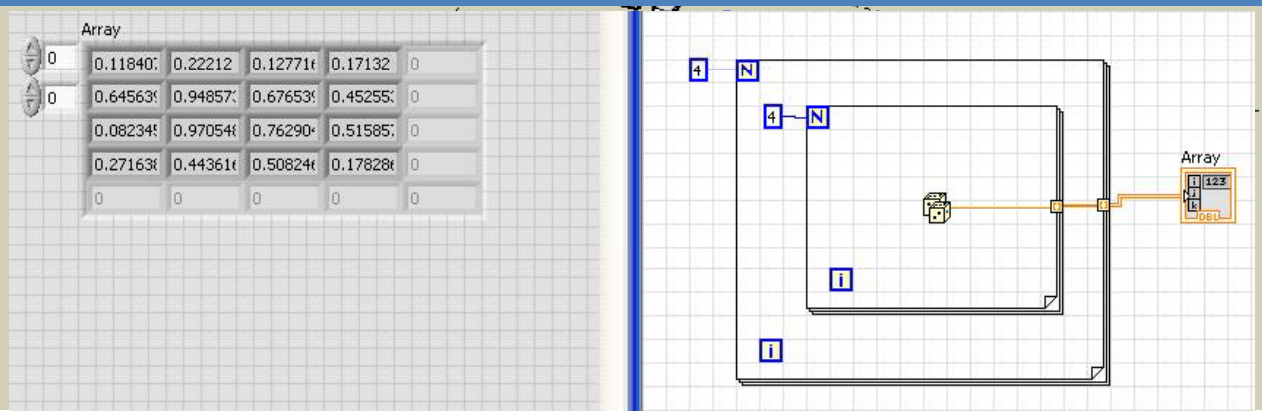
©2012 João Paulo Coelho

8.1 Ciclos **FOR**: aninhados

É possível colocar ciclos FOR no interior de outros ciclos FOR.



...utilizar esta estratégia para preencher ARRAYS multidimensionais...



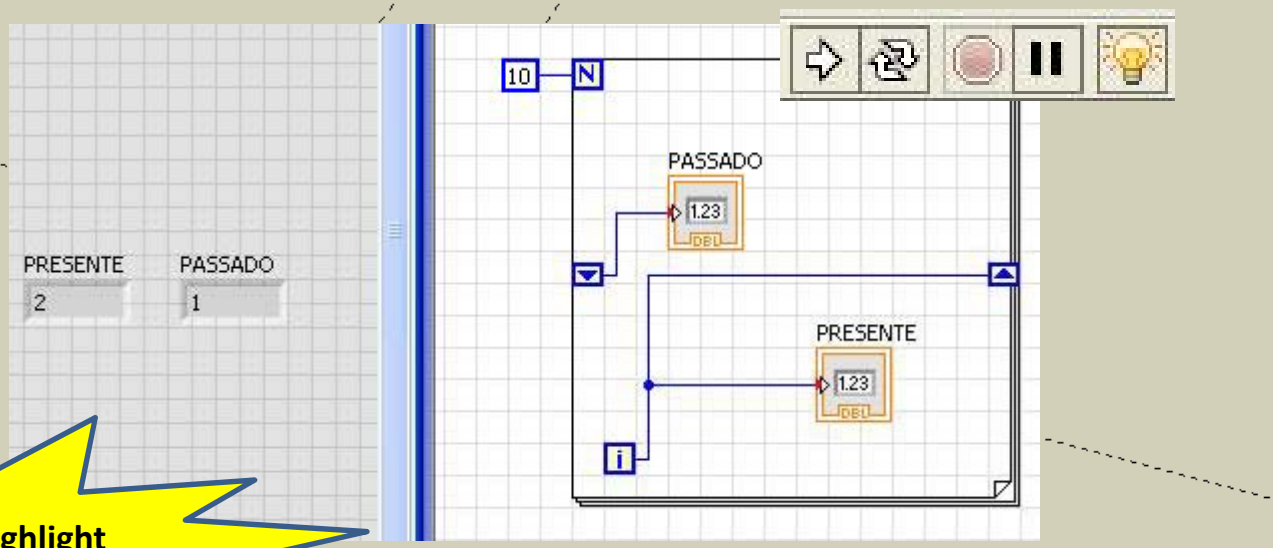
©2012 João Paulo Coelho

8.1 Ciclos **FOR**: Registos de Deslocamento

Frequentemente existe a necessidade de se utilizarem valores calculados numa iteração anterior do ciclo FOR.

O modo de aceder a valores passados consiste na criação de registos de deslocamento (Shift Registers).

EXEMPLO:



Execute com Highlight Execution ligado

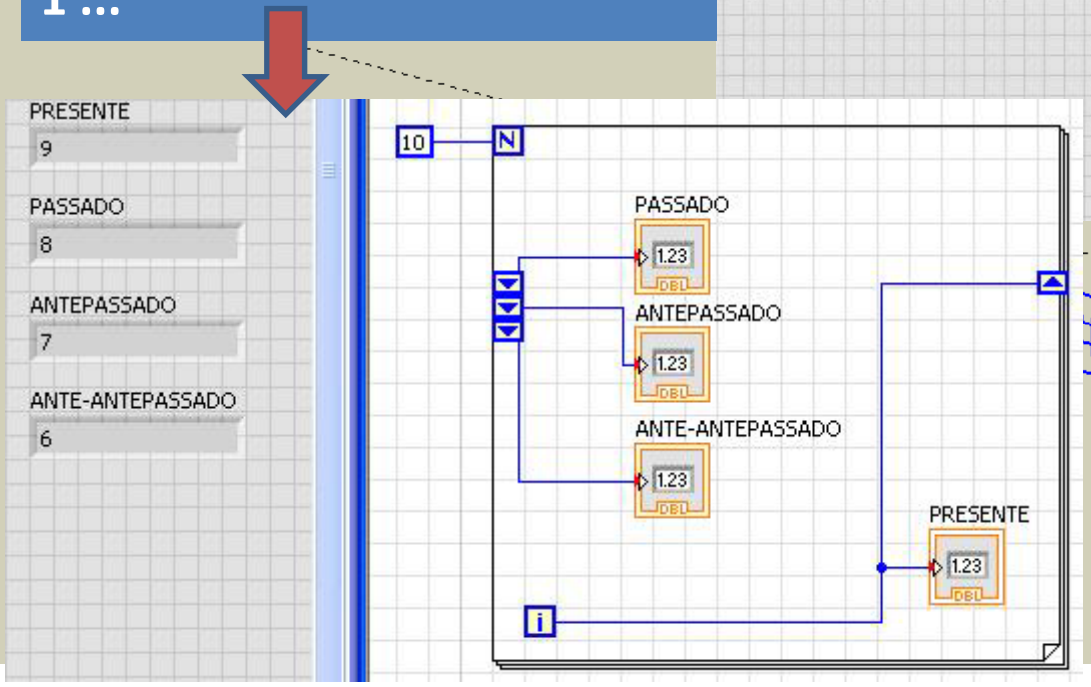
©2012 João Paulo Coelho

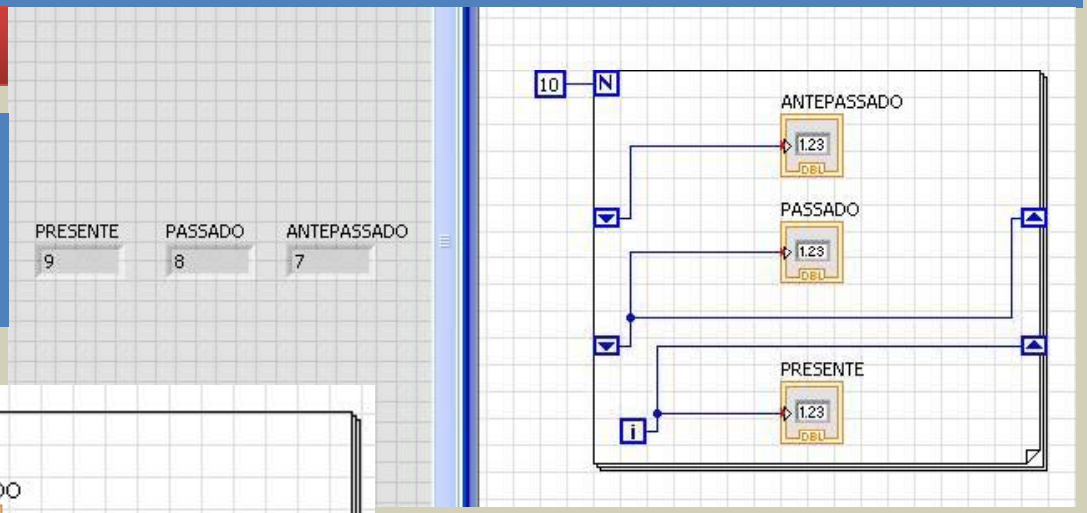
8.1 Ciclos **FOR**: Registos de Deslocamento

Podem ser adicionados um número arbitrário de shift registers.

EXEMPLO:

... Ou então um shift register com profundidade superior a 1 ...



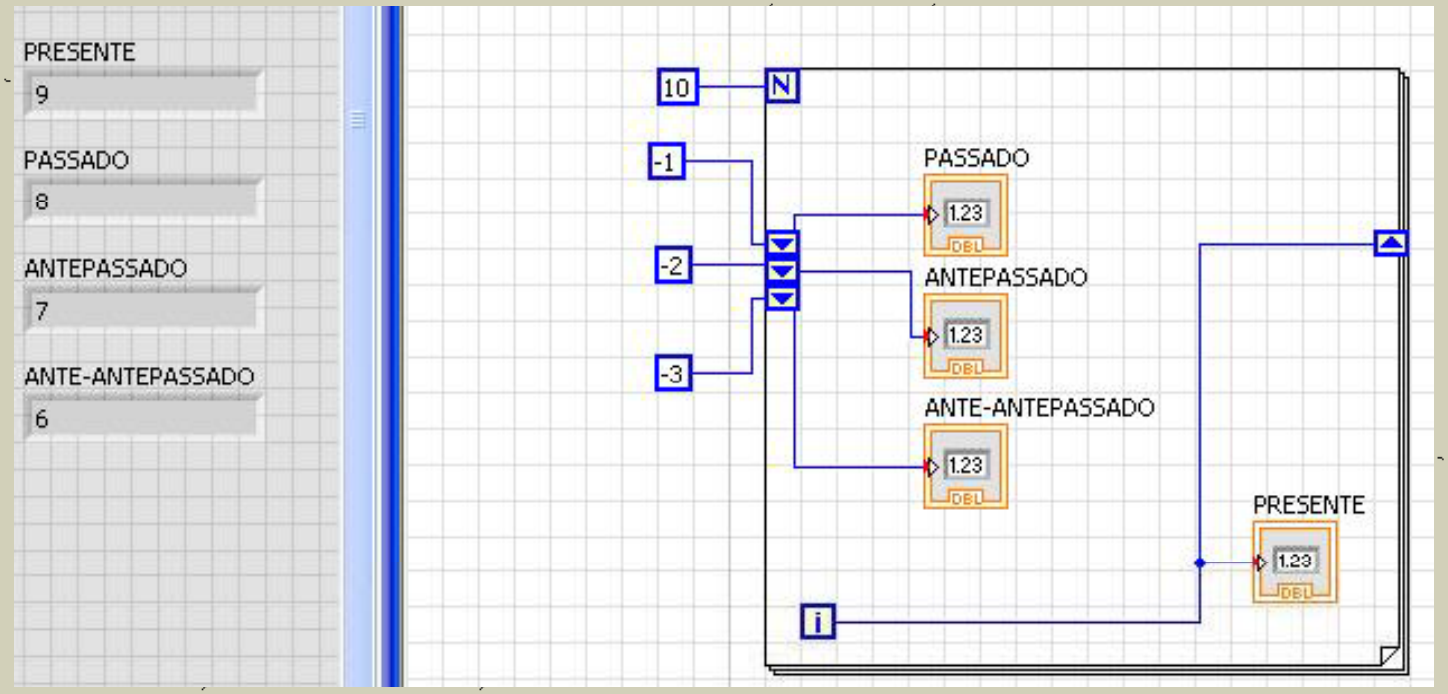


©2012 João Paulo Coelho

8.1 Ciclos **FOR**: Registos de Deslocamento

Os shift registers podem ser inicializados com valores diferentes de 0.

EXEMPLO:



©2012 João Paulo Coelho

8.1 Exercícios

***EX 19**: Construa um ARRAY bidimensional, utilizando ciclos **FOR**, com o seguinte aspecto:*

***EX 20**: Faça um VI capaz de apresentar num **ARRAY** os k primeiros números da sequência de **Fibonacci**:*

0, 1, 1, 2, 3, 5, 8, 13, 21, ...

O valor de k deve ser introduzido pelo utilizador a partir de um controlo numérico e deve ser superior a 2.

©2012 João Paulo Coelho