GRUPO EDUCACIONAL Opet

ALGORITMOS

PROFª: SIMONE DOMINICO

ARRANJOS UNIDIMENSIONAIS (VETORES OU ARRAYS)

Lembrando

RECAPITULANDO...

- Vimos como declarar variáveis e os seus tipos:

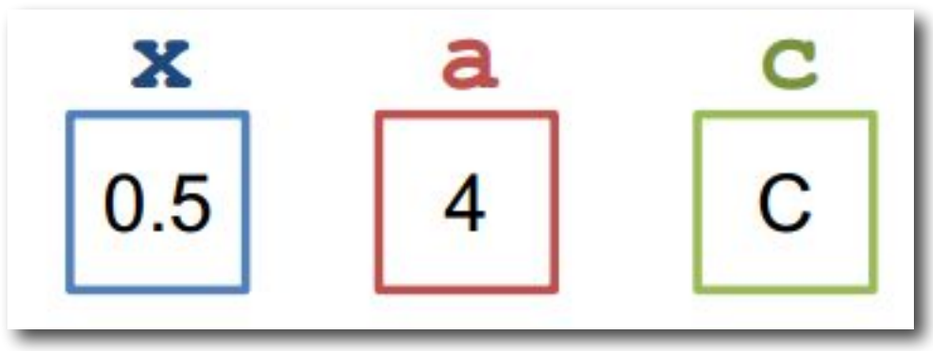
REAL

INTEIRO

CARACTER

- Vimos também como cada variável de um tipo consegue armazenar apenas um valor:

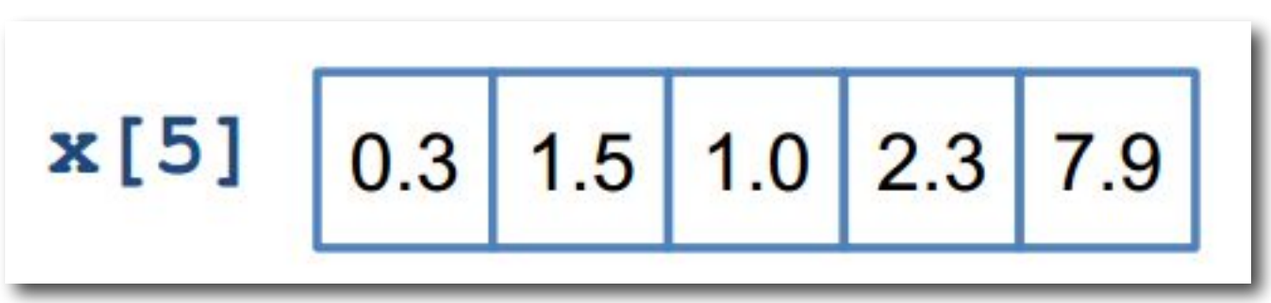
```
inteiro a = 4
real x = 0.5
caracter c = 'C'
```

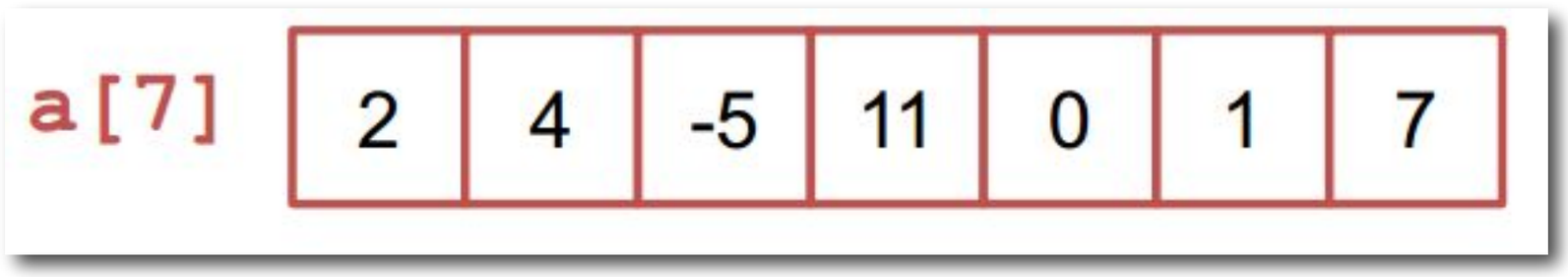


Agora

NESSA AULA...

- Veremos como podemos utilizar arranjos para definir e armazenar vários valores de um mesmo tipo





Analisando

CONSIDERE O ENUNCIADO

- Ler 30 valores e calcular a média aritmética dos mesmos.
- Análise do problema:
  - Quantas variáveis são necessárias para ler os 30 valores?

a) 30 variáveis?

b) 1 variável?

30 VARIÁVEIS DIFERENTES OU **1 VARIÁVEL ACUMULADORA** ONDE TODOS OS VALORES LIDOS SÃO SOMADOS PARA DEPOIS CALCULAR A MÉDIA

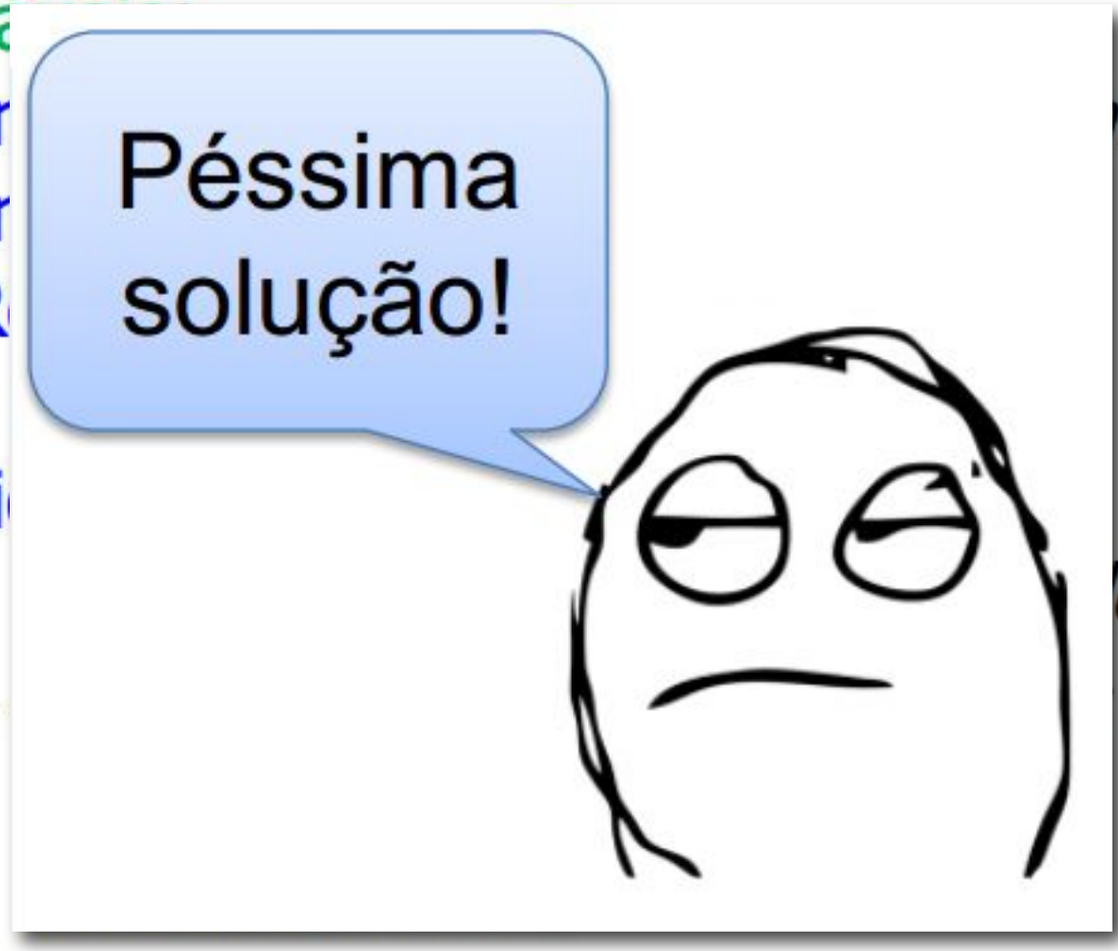
Solução

SOLUÇÃO COM 30 variáveis

```
// Lê 30 valores e calcula sua média aritmética
Variáveis:
    Inteiro: valor1, valor2, valor3, valor4,....,valor30
    Inteiro: soma
    Real:  media

Início
      Ler(valor1, valor2, valor3,....valor30)
      soma = valor1+valor2+valor3+...+valor30
      media = soma / 30
      Escrever(media)
Fim
```

SOLUÇÃO COM 30 VARIÁVEIS

```
// Lê 30 valores e calcula sua média aritmética
Variá
    Ir                                          lor4,...,valor30
    Ir
    R
Iníci                                           or30)
                                                ...+valor30
Fim
```



Acumuladora

SOLUÇÃO 1 VARIÁVEL ACUMULADORA

```
// Lê N valores e calcula sua média aritmética
Constantes: N = 30
Variáveis:
   Inteiro: valor, soma, i
     Real: media

Início
        soma = 0;
        Para (i=0; i<N; i++) {
            Ler (valor)
            soma = soma + valor
        }
       media = soma / N
       Escrever(media)
Fim
```

SOLUÇÃO 1 VARIÁVEL ACUMULADORA

```
// Lê N valores e calcula sua média aritmética
Constantes: N = 30
Variáveis:
   Inteiro:
    Real:

Início
        soma
        Para
            Le
            so
        }
      media
      Escrever(media)
Fim
```

se...

SE O PROBLEMA FOSSE

- Ler 30 valores, calcular a média aritmética dos mesmos e imprimir a lista de valores que ficarem acima da média.
- Análise do problema:
  - Quantas variáveis são necessárias para ler os 30 valores?

a) 30 variáveis?

b) 1 variável?

SE O PROBLEMA FOSSE

- Ler 30 valores, calcular a média aritmética dos mesmos e imprimir a lista de valores que ficarem acima da média.
- Análise do problema:
  - Quantas variáveis são necessárias para ler os 30 valores?

a) 30 variáveis?

b) 1 variável?

COM 1 VARIÁVEL ACUMULADORA SE PERDEM OS VALORES ORIGINALMENTE LIDOS E COM 30 VARIÁVEIS JÁ VIMOS QUE A SOLUÇÃO NÃO É DAS MAIS ELEGANTES

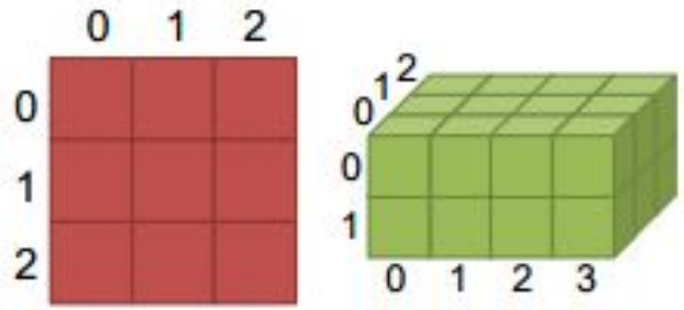
ARRANJOS

- Estruturas para armazenamento de múltiplos elementos de dados
  - Armazenamento de dados de mesmo tipo
  - Armazenamento contíguo na memória
  - Acesso indexado

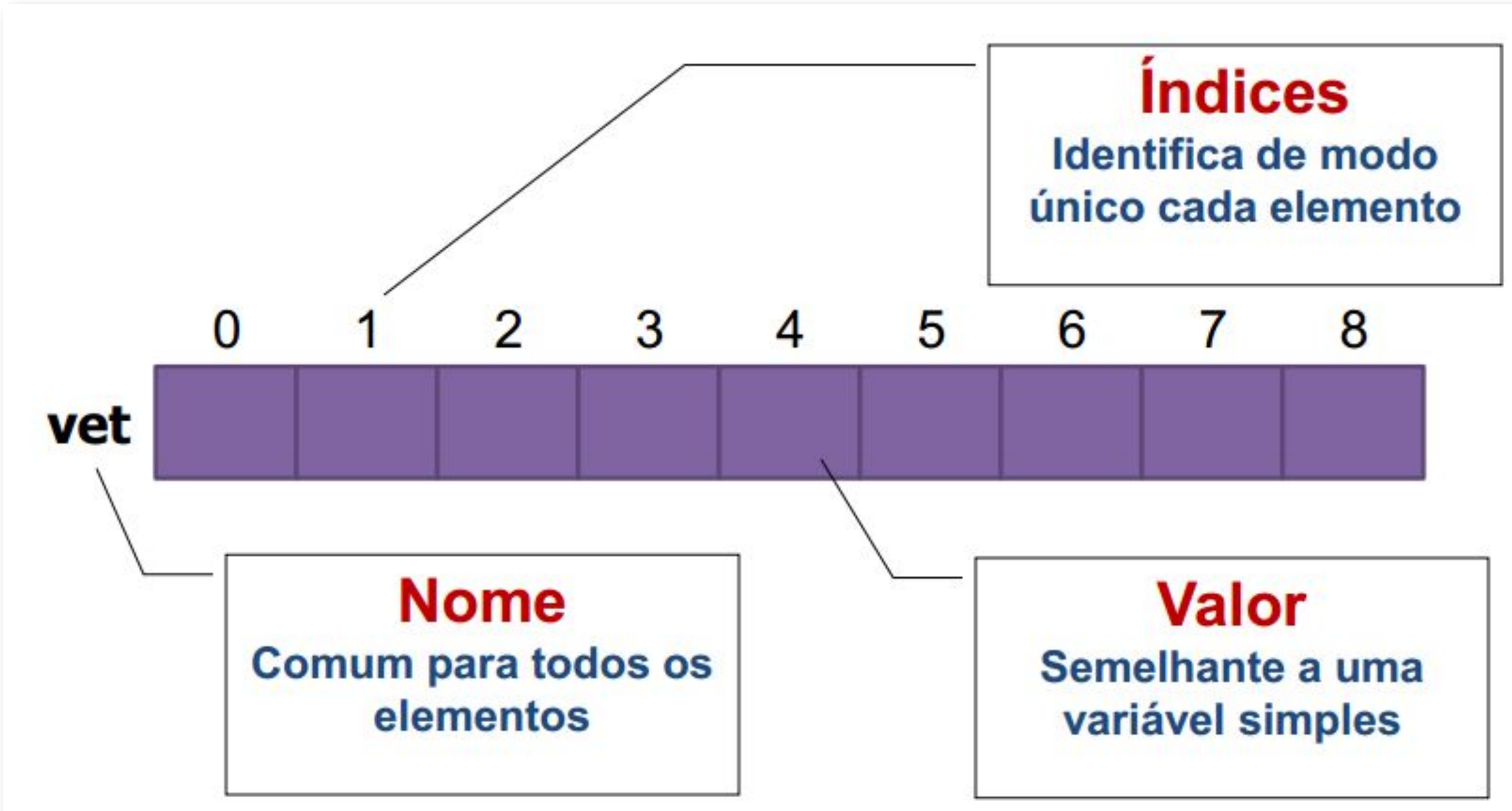
- Unidimensionais
  - Vetores ou Arrays

Na aula de hoje!

- Multidimensionais
  - Matrizes, cubos, etc.



ARRANJOS UNIDIMENSIONAIS (VETORES OU ARRAYS)



DECLARAÇÃO DE VETORES

- Forma geral

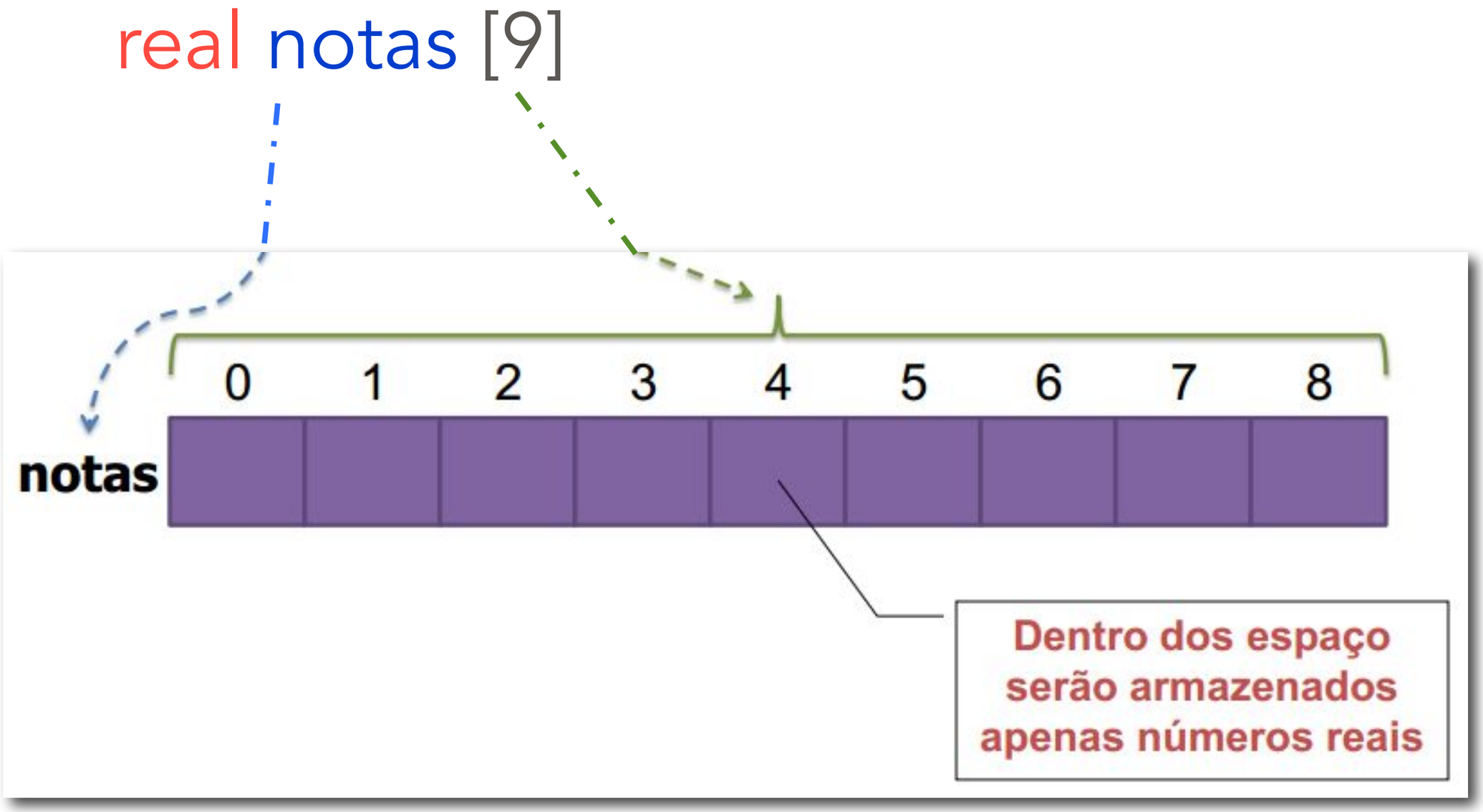
tipo nome [tamanho];

TIPO DE DADO
(REAL, INTEIRO...)

NOME DA VARIÁVEL
(ESCOLHIDO PELO PROGRAMADOR)

TAMANHO VETOR
(QUANTOS VALORES SÃO ARMAZENADOS)

DECLARANDO UM VETOR DE 9 NOTAS DE ALUNOS



INICIALIZAÇÃO DE VETORES

- Vetores, como as demais variáveis, ao serem criados não limpam a área da memória onde serão alocados os valores
- Vetores podem ser inicializados de 3 formas:

1. Na declaração;
2. Por atribuição, em algum momento da execução;
3. Por leitura.

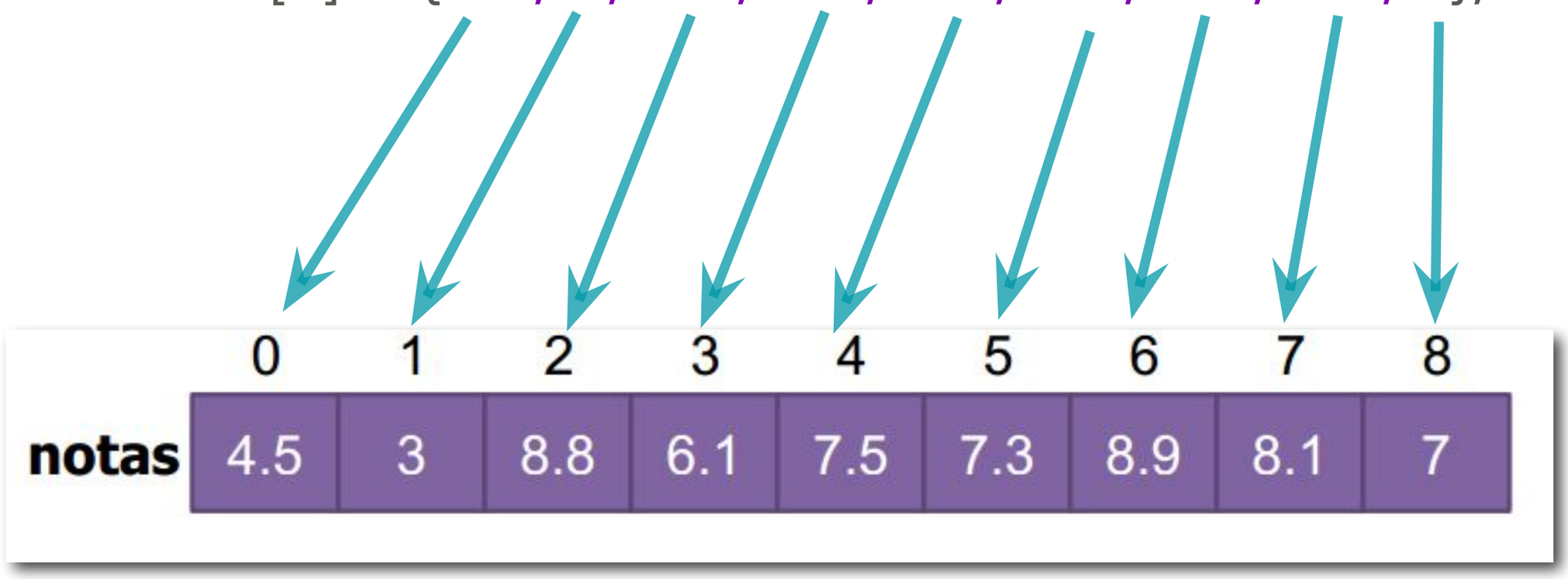
Declaração

INICIALIZAÇÃO NA DECLARAÇÃO

Forma geral

```
tipo nome[tam] = {valor0, valor1, ... valortam-1};
```

```
real notas [9] = {4.5, 3, 8.8, 6.1, 7.5, 7.3, 8.9, 8.1, 7};
```

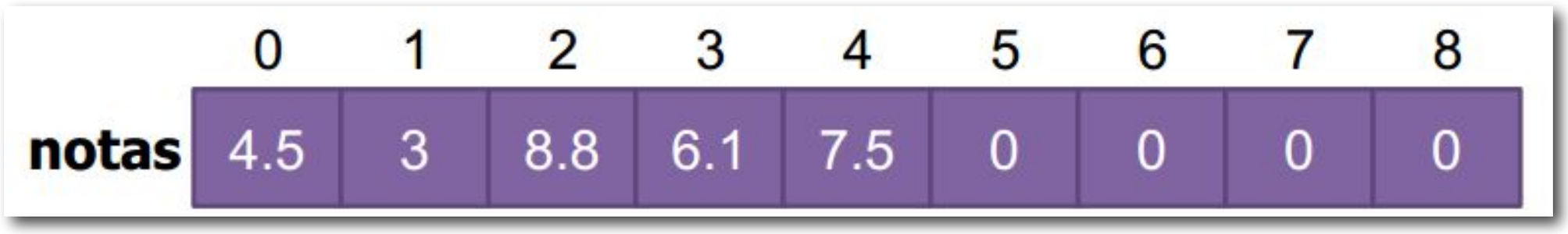


Declaração

INICIALIZAÇÃO NA DECLARAÇÃO

Posições não inicializadas são preenchidas com zero

```
real notas [9] = {4.5, 3, 8.8, 6.1, 7.5};
```



Atribuição

INICIALIZAÇÃO POR ATRIBUIÇÃO

Inicializar um vetor na declaração assim...

```
inteiro numeros [9] = {4, 3, 8, 1, 5};
```

é o mesmo que

```
inteiro numeros[6]
  numeros[0] = 4
  numeros[1] = 3
  numeros[2] = 8
  numeros[3] = 1
  numeros[4] = 5
  numeros[5] = 3
```

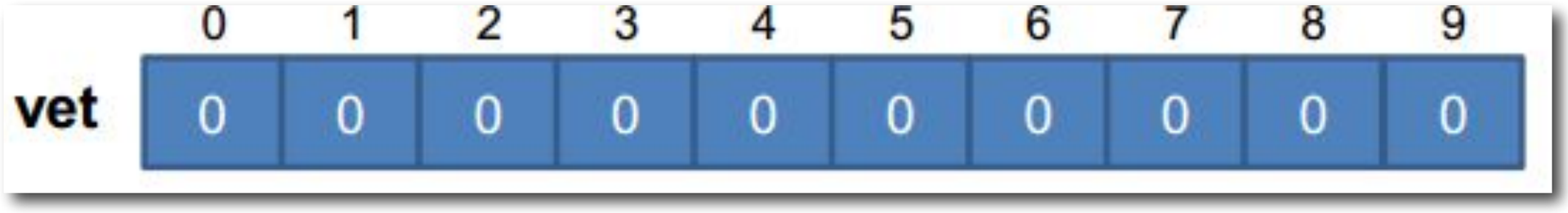
Inicialização por atribuição, durante a execução

Atribuição

INICIALIZAÇÃO POR ATRIBUIÇÃO

```
ALGORITMO EXEMPLO
INICIO
    INTEIRO VET[10]
    INTEIRO I
    PARA(I=0;I<10;I++)FACA
        VET[I] = 0
    FIMPARA
FIMALGORITMO
```

Zerar todo o conteúdo de um vetor de 10 posições



Atribuição

INICIALIZAÇÃO POR ATRIBUIÇÃO

```
ALGORITMO EXEMPLO
INICIO
    INTEIRO VET[10]
    INTEIRO I
    PARA(I=0;I<10;I++)FACA
        VET[I] = I + 1
    FIMPARA
FIMALGORITMO
```

Inicializar um vetor de 10 posições com valores consecutivos a partir de 1

leitura

INICIALIZAÇÃO POR LEITURA

O usuário informa os valores que devem ser armazenados no vetor

```
ALGORITMO EXEMPLO
INICIO
    INTEIRO VET[10]
    INTEIRO I
    PARA(I=0;I<10;I++)FACA
        ESCREVA("DIGITE O VALOR DA POSICAO ", I)
        LEIA(VET[I])
    FIMPARA
FIMALGORITMO
```

Voltando

VOLTANDO AO NOSSO ENUNCIADO

- Ler 30 valores, calcular a média aritmética dos mesmos e imprimir a lista de valores que ficarem acima da média.
- Perguntas:
  - Quantas posições precisa ter o nosso arranjo?
    -
  - De que forma devemos iniciar os valores do arranjo?
    -
  - De que forma imprimimos a lista?

Voltando

VOLTANDO AO NOSSO ENUNCIADO

- Ler 30 valores, calcular a média aritmética dos mesmos e imprimir a lista de valores que ficarem acima da média.
- Perguntas:
  - Quantas posições precisa ter o nosso arranjo?
    - 30 valores (vamos assumir inteiros)
  - De que forma devemos iniciar os valores do arranjo?
    - Por leitura, os valores são informados pelo usuário
  - De que forma imprimimos a lista?
    - Percorrendo e comparando os valores com a média

Solução

SOLUÇÃO

```
ALGORITMO EXEMPLO
CONSTANTE TAMANHO 5
INICIO
    INTEIRO VET[30], SOMA  = 0, POSICAO
    REAL MEDIA
    PARA(POSICAO=0;POSICAO<10;POSICAO++)FACA
        ESCREVA("DIGITE O PRIMEIRO VALOR: ")
        LEIA(VETOR[POSICAO])
        SOMA = SOMA + VETOR[POSICAO]
    FIMPARA
    MEDIA = SOMA/TAMANHO
    PARA(POSICAO=0;POSICAO<10;POSICAO++)FACA
            SE(VETOR[POSICAO]>=MEDIA)ENTAO
                ESCREVA("FICOU ACIMA DA MÉDIA", VETOR[POSICAO])
            FIMSE
    FIMAPARA
FIMALGORITMO
```

Solução

SOLUÇÃO

Welcome | C aula.c

home > dominico > C aula.c > main()

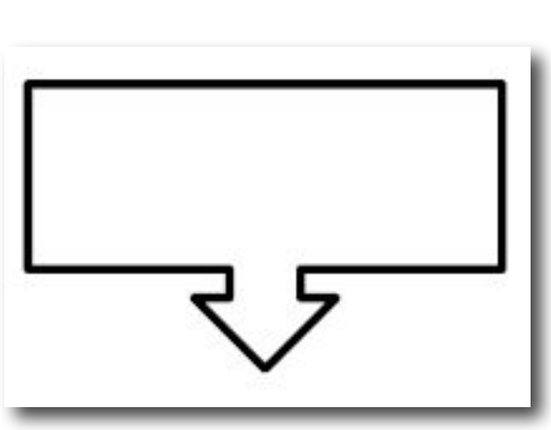
```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size], posicao, soma=0;
    float media;
    for ( posicao = 0; posicao < size; posicao++)
    {

        printf("Digite a nota: ");
        scanf("%d", &vet[posicao]);
        soma = soma + vet[posicao];
    }
    media = soma/size;
    for ( posicao = 0; posicao < size; posicao++)
    {
        if (vet[posicao]>=media)
        {
            printf("Acima da meedia: %d", vet[posicao]);
        }
    }

    printf("\n");
    return 0;
}
```

GRUPO EDUCACIONAL Opet

VETORES UNIDIMENSIONAIS - TESTE DE MESA E EXEMPLOS

Teste de mesa

SEJA UM VETOR, DE NOME VALOR, COM 5 ELEMENTOS



INICIALIZAÇÃO, POR LEITURA, DO VETOR

Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

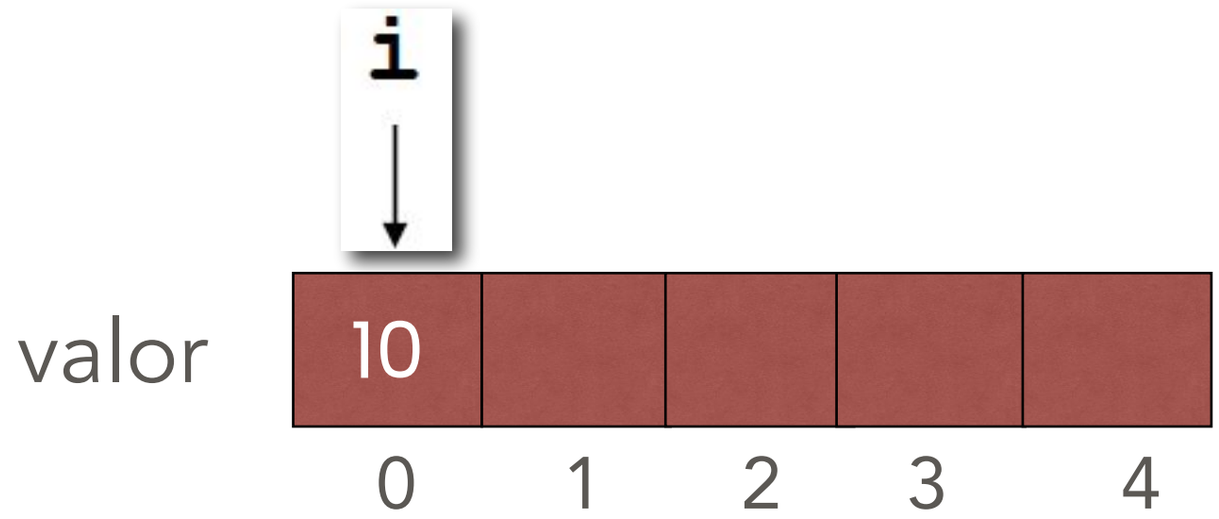
home > dominico > aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 1

i = 0

Usuário digitou: 10



Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

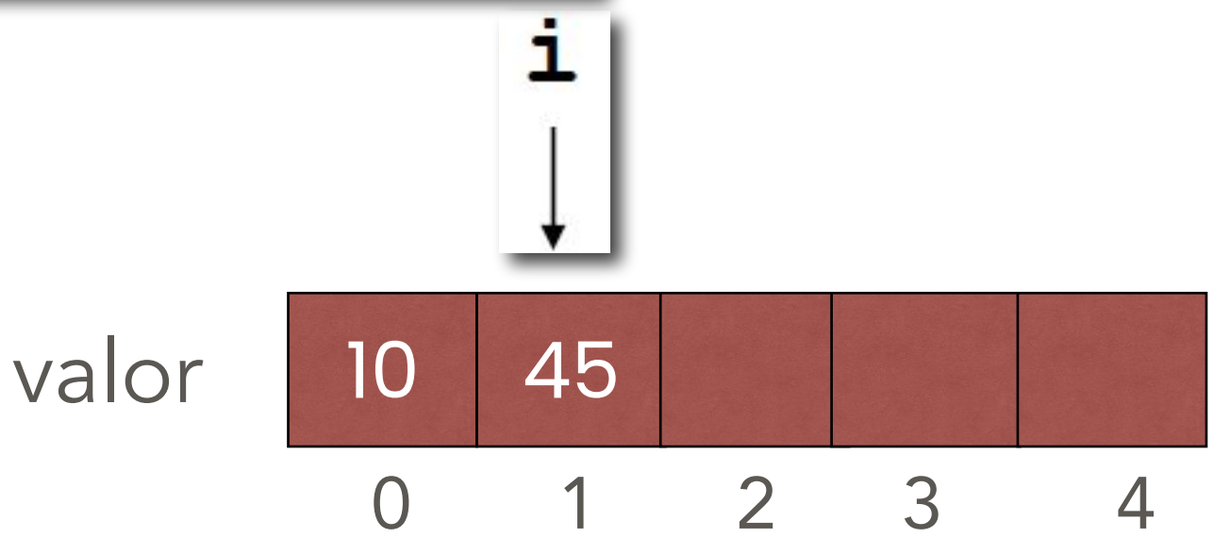
home > dominico > C aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 2

i = 1

Usuário digitou: 45



Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

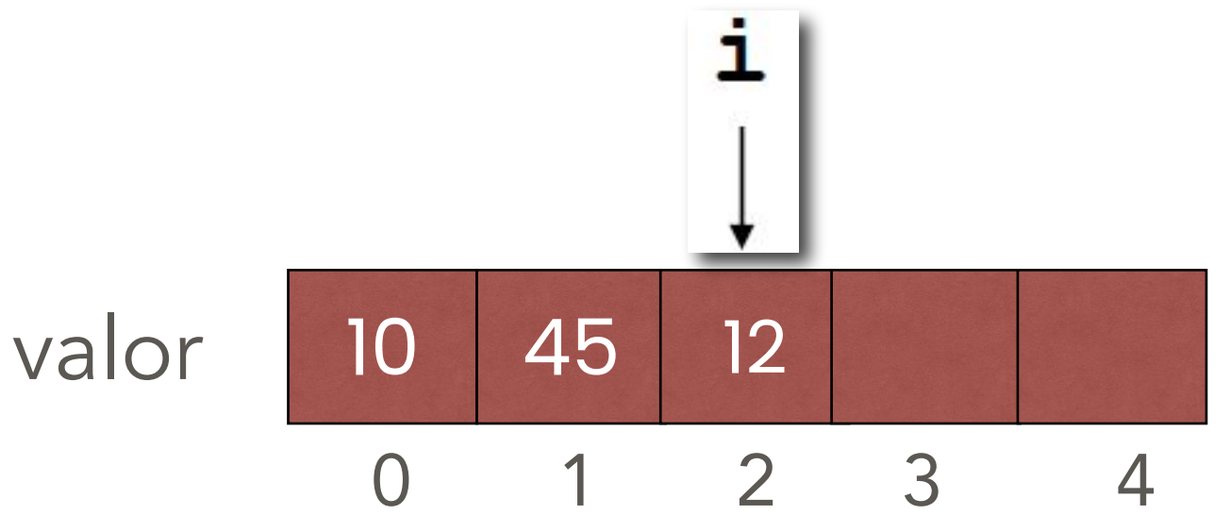
home > dominico > C aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 3

i = 2

Usuário digitou: 12



Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

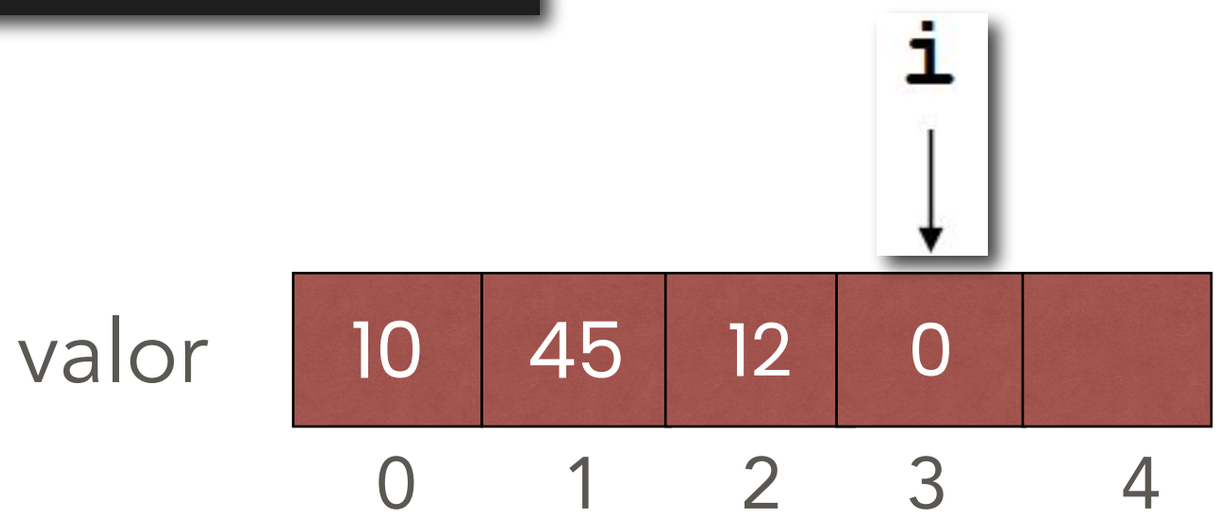
home > dominico > aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 4

i = 3

Usuário digitou: 0



Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

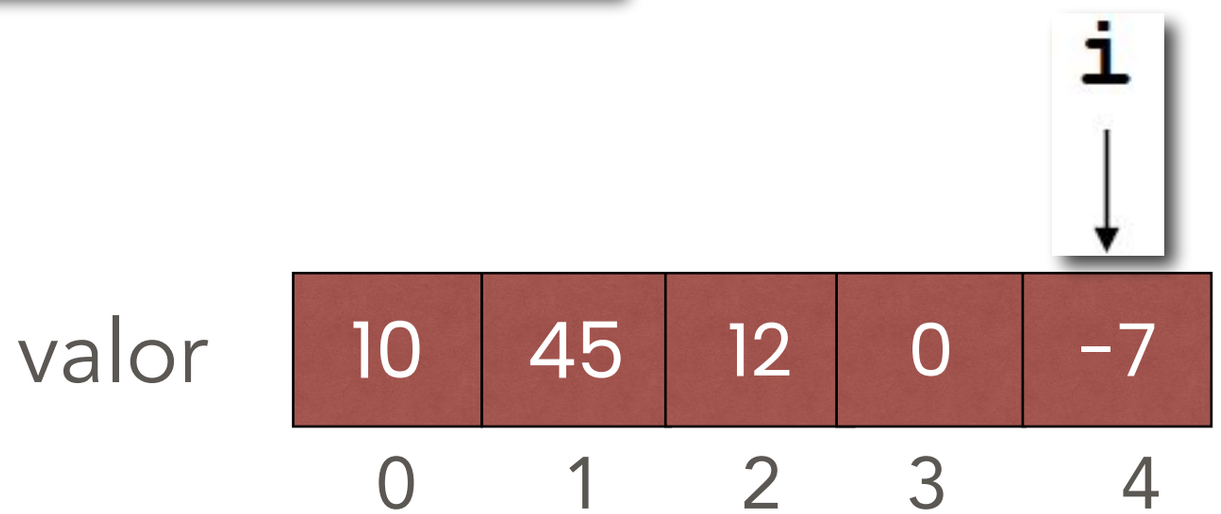
home > dominico > C aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 5

i = 4

Usuário digitou: -7



Teste de mesa

INICIALIZAÇÃO, POR LEITURA, DO VETOR

home > dominico > C aula.c > main()

```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("Digite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    return 0;
}
```

Iteração 6

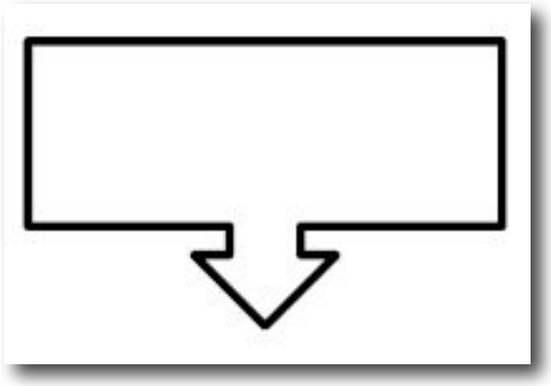
i = 5

FORA DO LAÇO

valor	10	45	12	0	-7
	0	1	2	3	4

Teste de mesa

SEJA UM VETOR, DE NOME VALOR, COM 5 ELEMENTOS



APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Teste de mesa

APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

home > dominico > C aula.c > main()

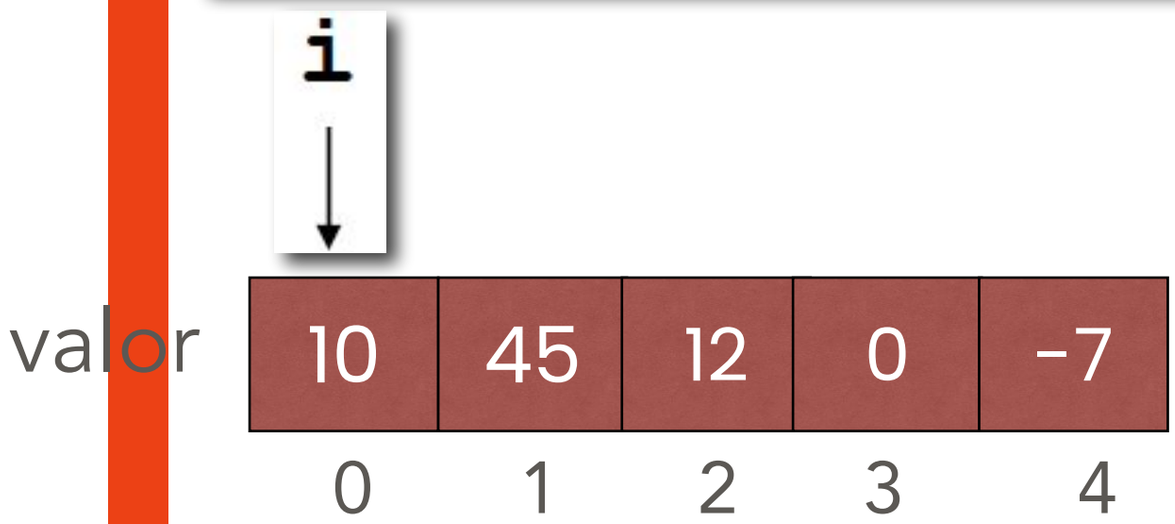
```c
#include<stdio.h>
#define size 5


int main(){
    int vet[size],i;
    //preenche o vetor
    for ( i = 0; i < size; i++)
    {
        printf("\nDigite o valor do vet[ %d ]", i);
        scanf("%d", &vet[i]);
    }
    //apresentar valores em tela
    for ( i = 0; i < size; i++)
    {
        printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
    }

    return 0;
}
```

Teste de mesa

APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Obs. Tem código antes

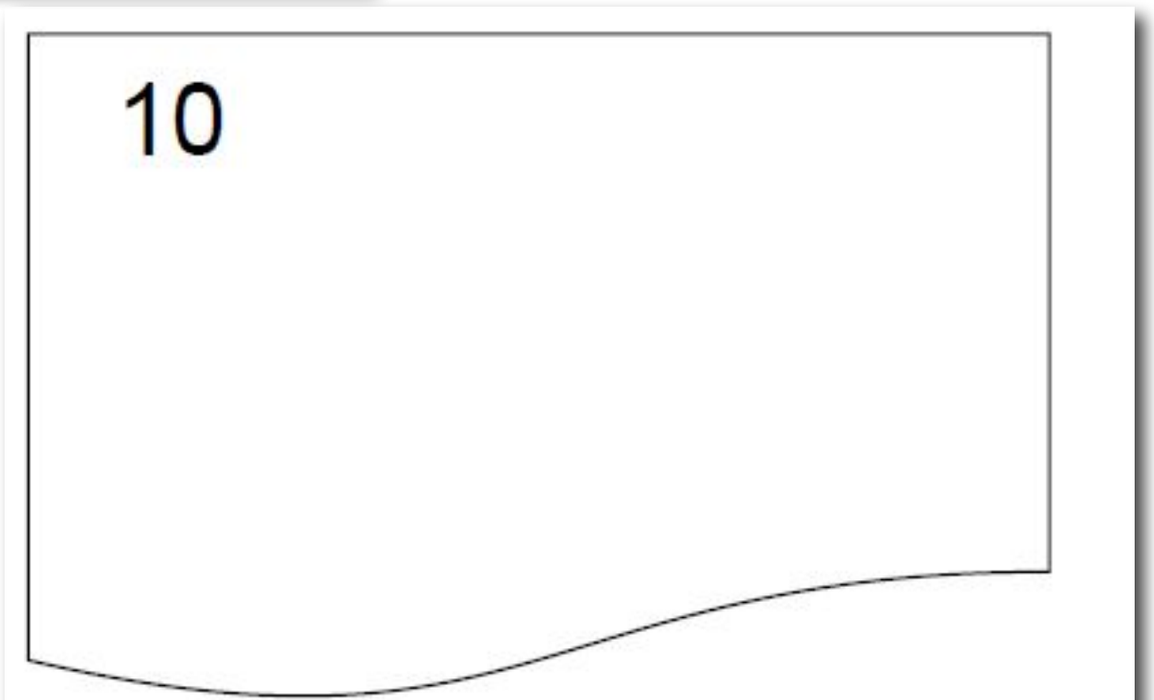
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```



Iteração 1

i = 0

Tela



Teste de mesa

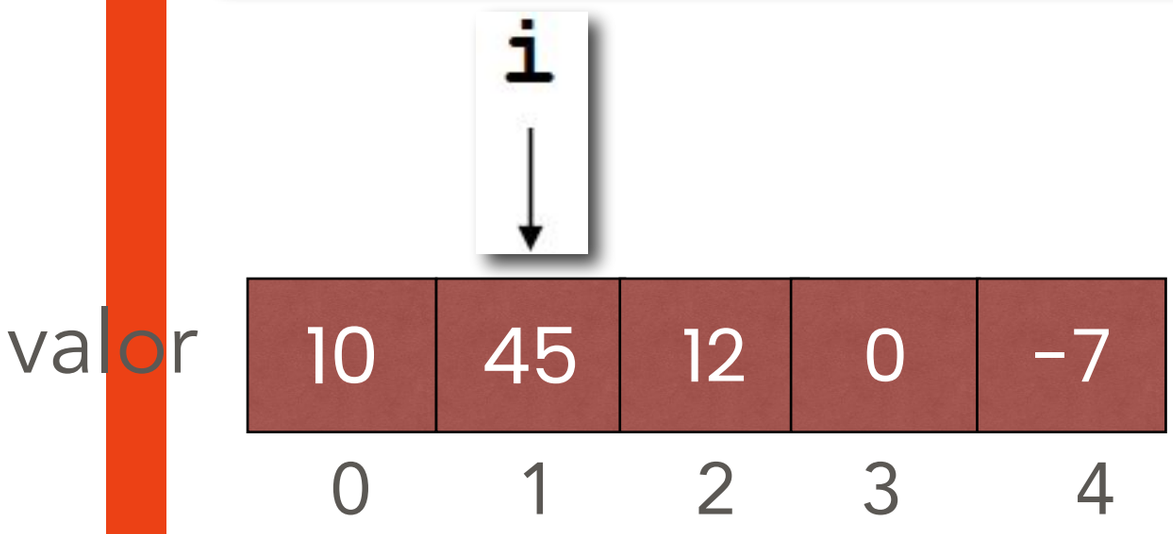
APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Obs. Tem código antes

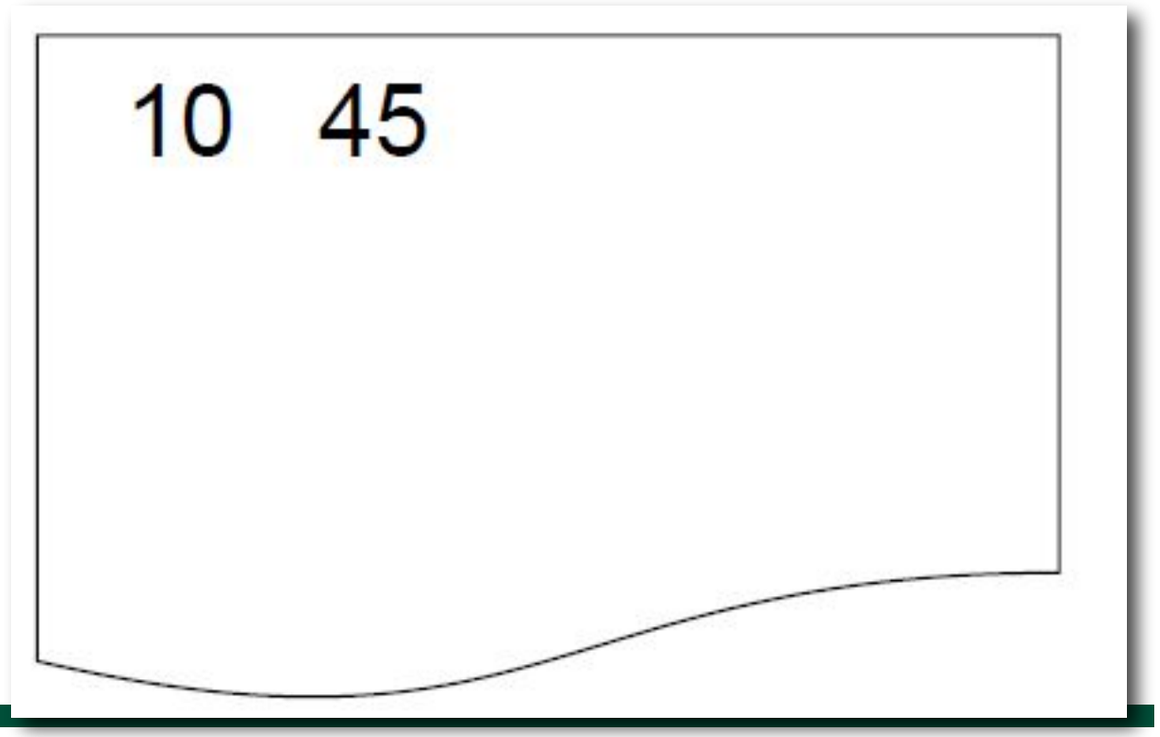
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```

Iteração 2

i = 1



Tela



Teste de mesa

APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

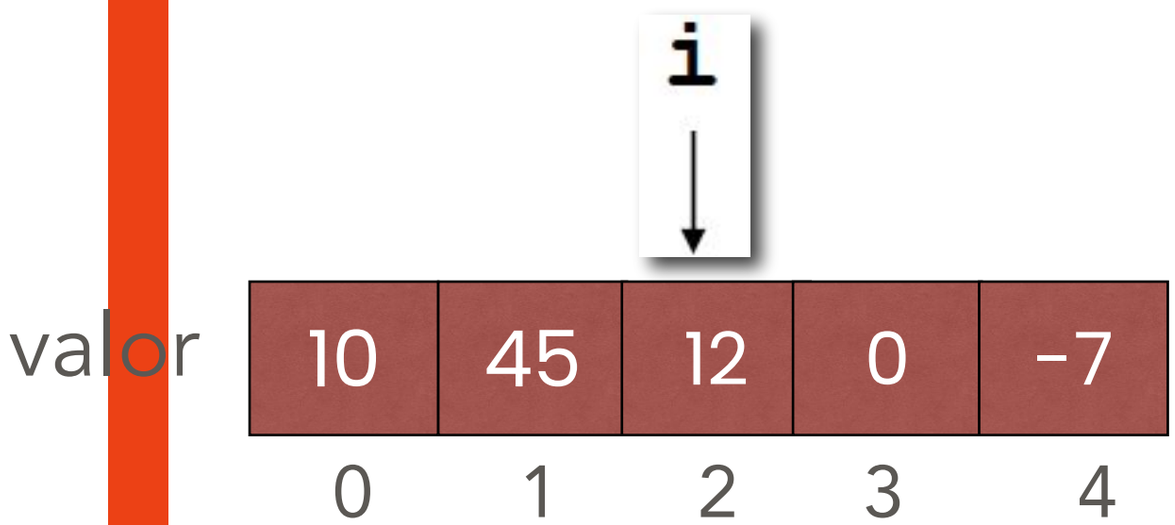
Obs. Tem código antes

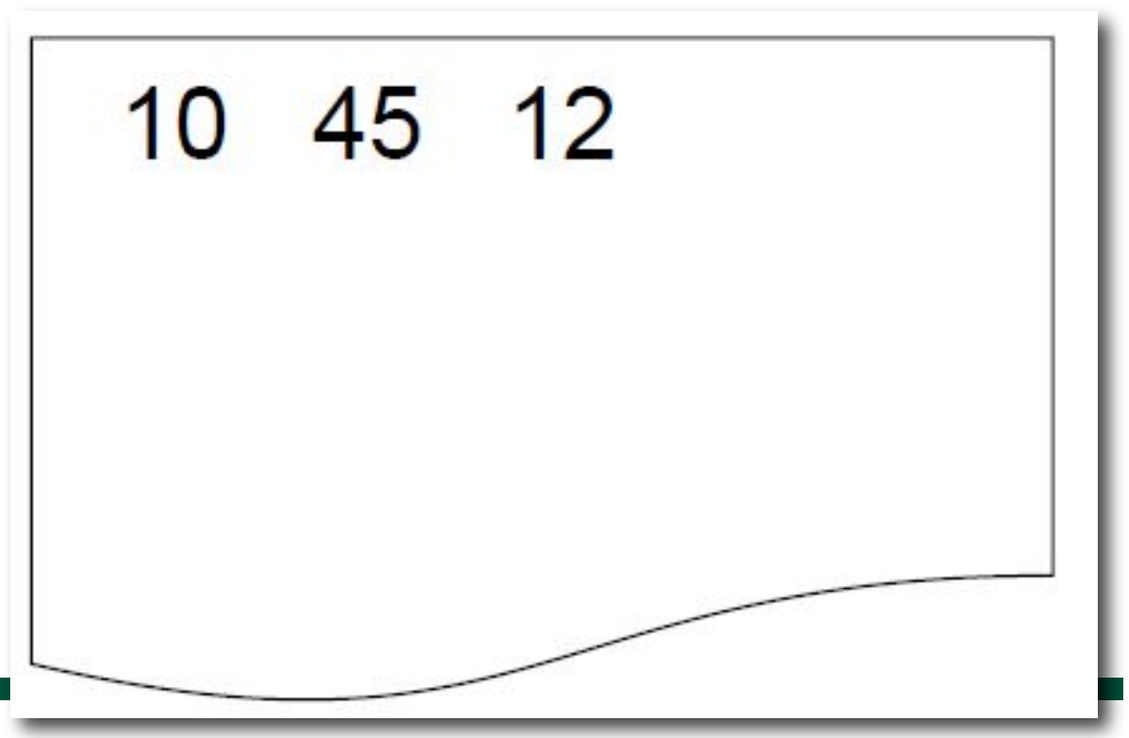
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```

Iteração 3

i = 2

Tela





Teste de mesa

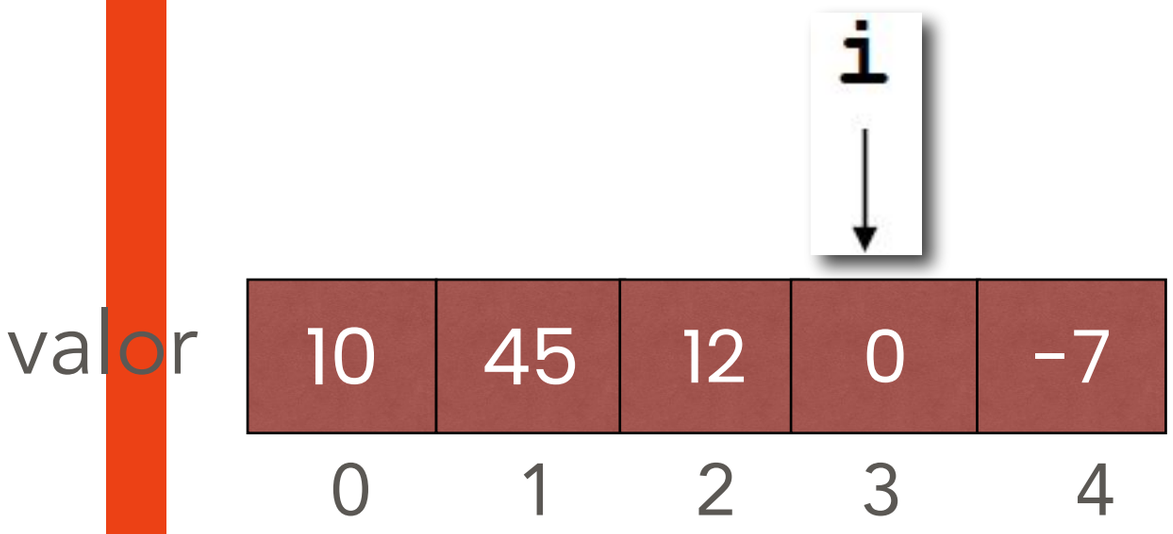
APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Obs. Tem código antes

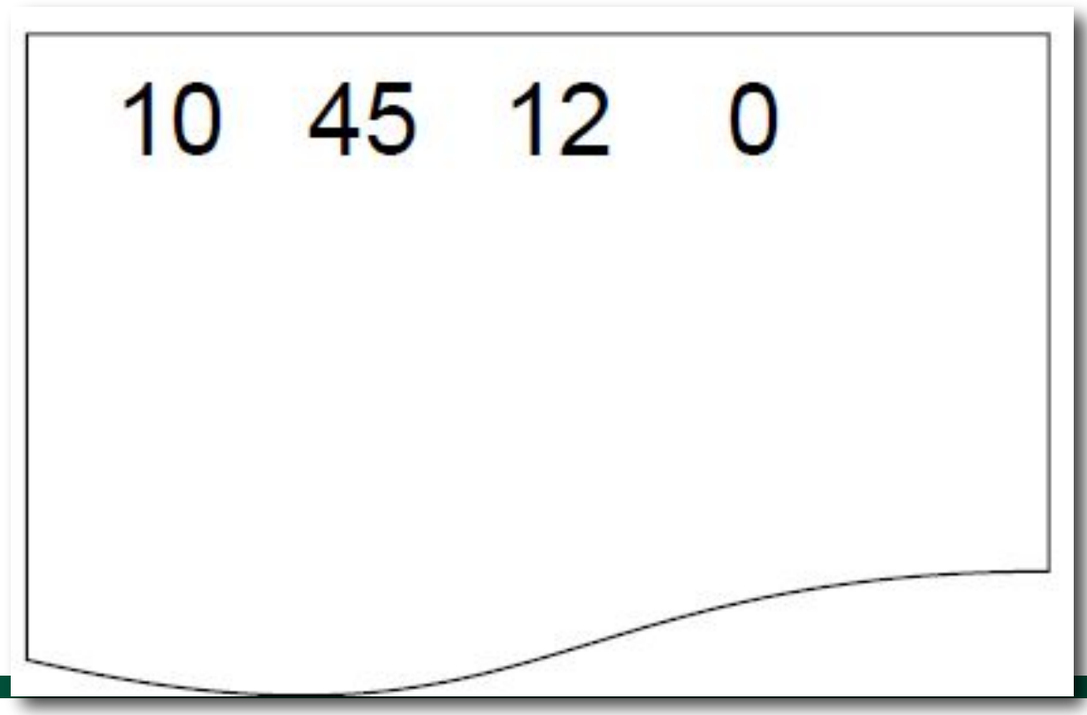
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```

Iteração 4

i = 3



Tela



Teste de mesa

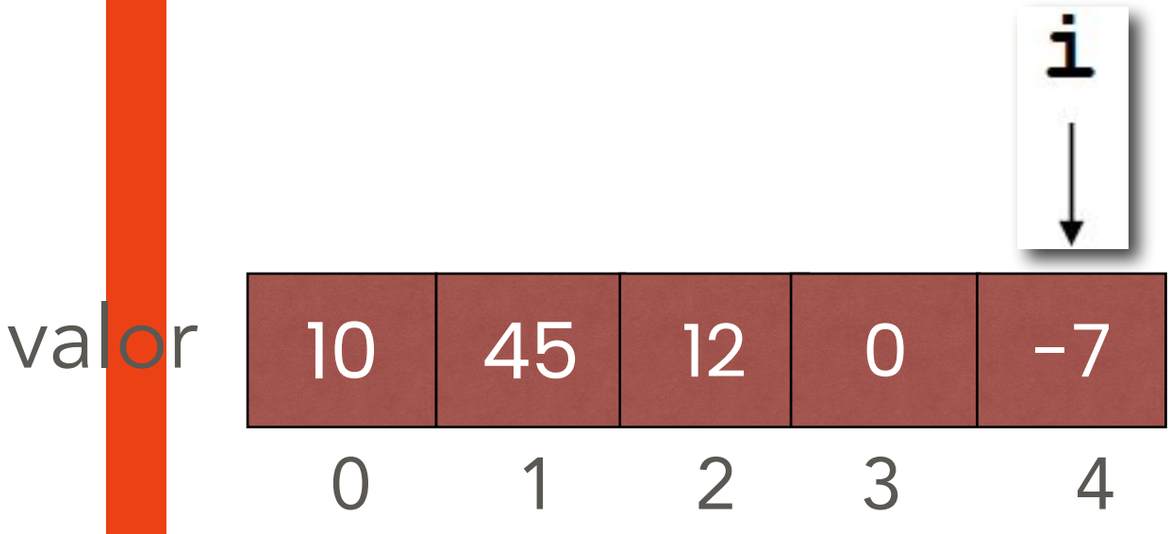
APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Obs. Tem código antes

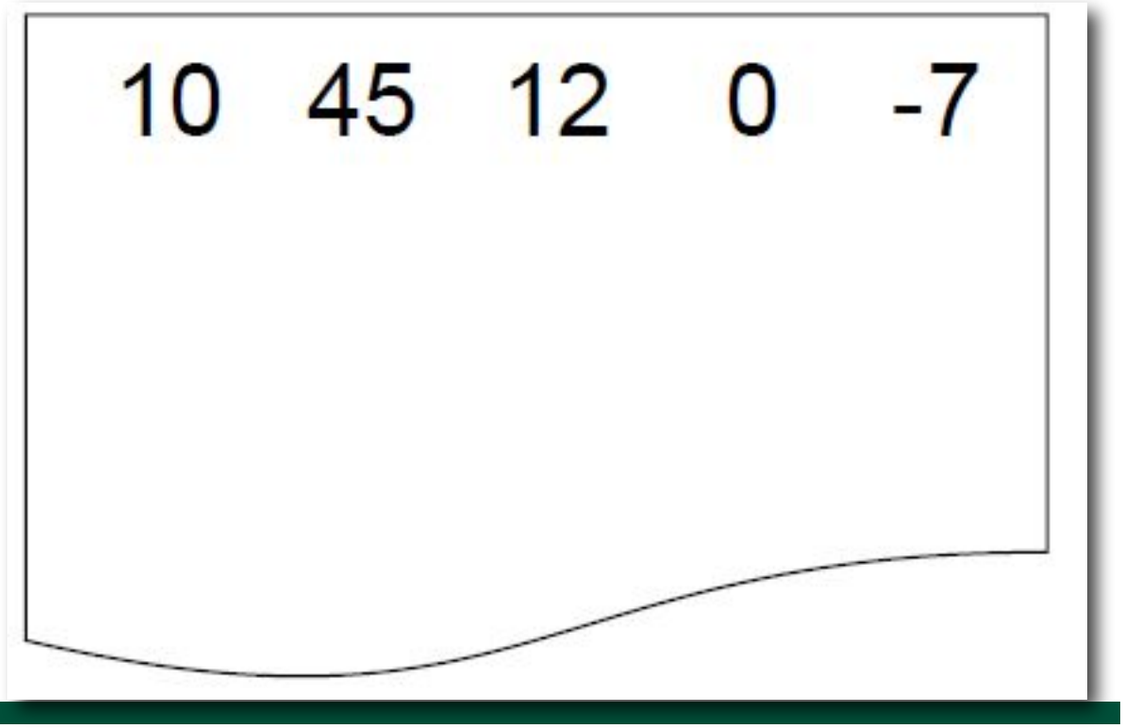
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```

Iteração 5

i = 4



Tela



Teste de mesa

APRESENTAÇÃO NA TELA DO CONTEÚDO DO VETOR

Obs. Tem código antes

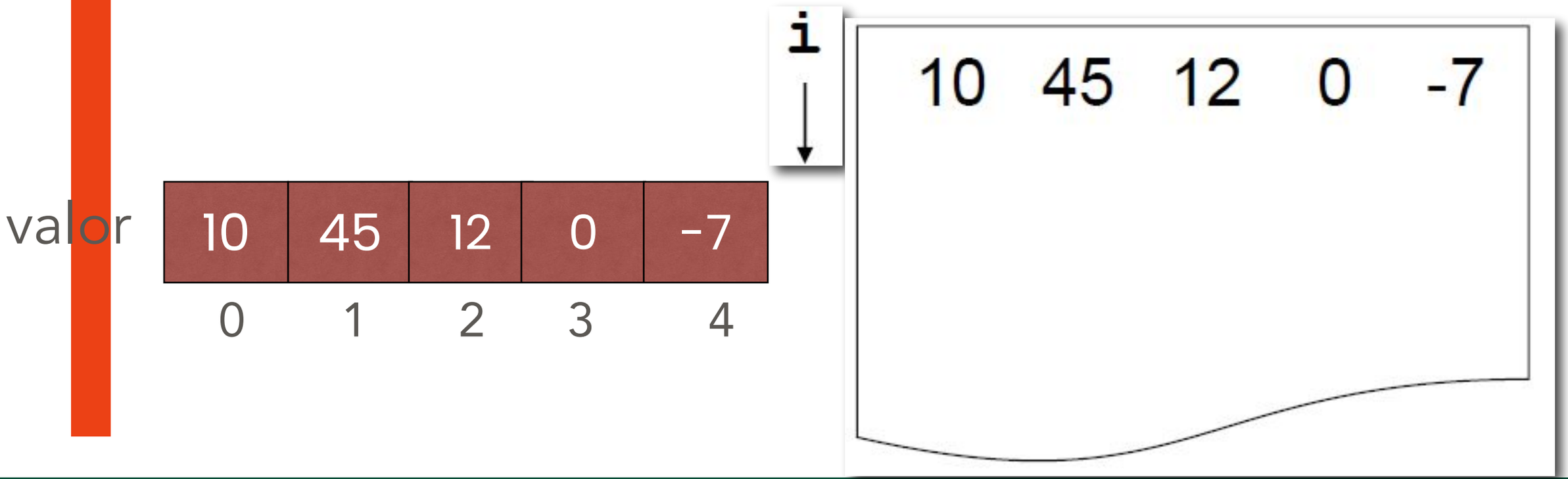
```
}
//apresentar valores em tela
for ( i = 0; i < size; i++)
{
    printf("\nVetor v[ %d ] da posição: %d.", vet[i],i);
}
```

Iteração 6

i = 5

FORA DO LAÇO

Tela



valor

10	45	12	0	-7
0	1	2	3	4