



Universidade Federal do ABC

Bacharelado em Ciência e Tecnologia
Processamento da Informação

Vetores

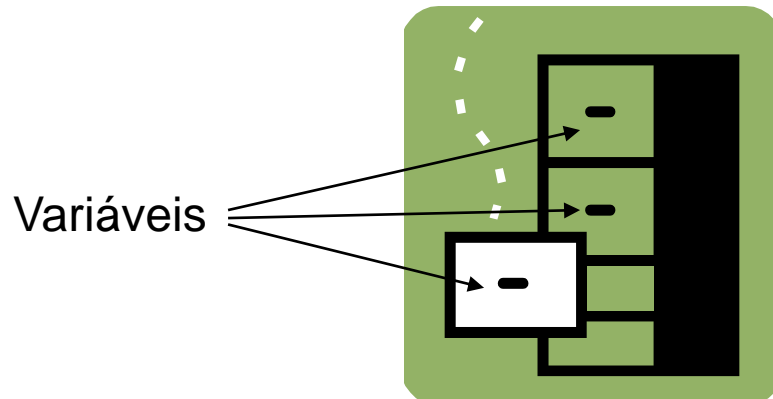
Vetores (unidimensionais)

Objetivos

- Entender a importância e a necessidade do uso de Vetores
- Definição de Vetores Unidimensionais
- Manipulação de Vetores
 - Inserir elementos em um vetor (usando laços ou não)
 - Acessar elementos de um vetor (usando laços ou não)

Vetores

Memória



```
0:000> dd 0x01f8b890
01f8b890 00000000 0000000a 00000032 000000fa
01f8b8a0 000002ee 000004c9 00000000 5e3d2a78
01f8b8b0 00000000 00000000 00000000 00000000
01f8b8c0 00000000 00000000 00523874 00000000
01f8b8d0 00000000 00000000 00000000 00000000
01f8b8e0 00000000 00000000 00000000 00000000
01f8b8f0 00000000 00000000 00000000 00000000
01f8b900 00000000 00000000 00000000 00000000
```

Vetores

Address	Content	Name	Type	Value
90000000	00	sum	int (4 bytes)	000000FF (255 ₁₀)
90000001	00			
90000002	00			
90000003	FF			
90000004	FF	age	short (2 bytes)	FFFF (-1 ₁₀)
90000005	FF			
90000006	1F	average	double (8 bytes)	1FFFFFFFFFFFFFFFFF (4.45015E-308 ₁₀)
90000007	FF			
90000008	FF			
90000009	FF			
9000000A	FF			
9000000B	FF			
9000000C	FF			
9000000D	FF			
9000000E	90	ptrSum	int* (4 bytes)	90000000
9000000F	00			
90000010	00			
90000011	00			

Note: All numbers in hexadecimal

Vetores

Var Tipo

-

A0 - Inteiro

-

A1 - Inteiro

-

A2 - Inteiro

-

A3 - Inteiro

-

A4 - Inteiro

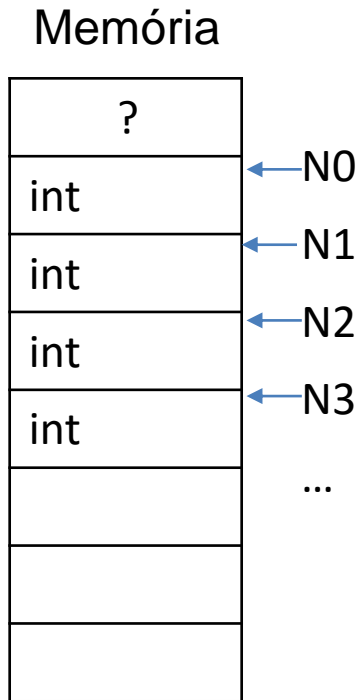
-

A5 - Inteiro

-

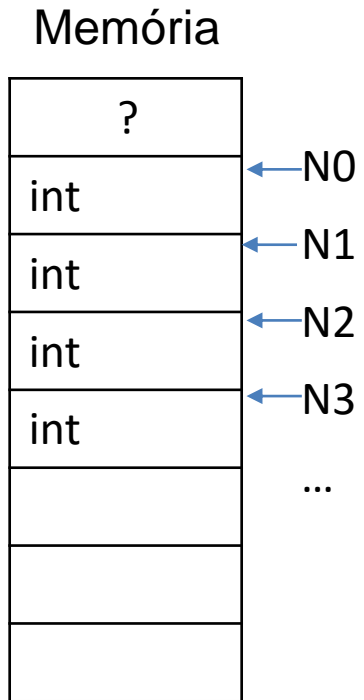
A6 - Inteiro

Vetores

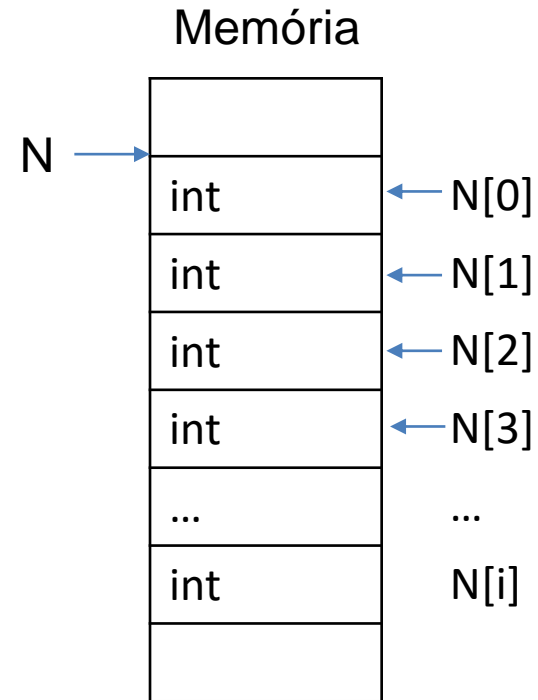


Posição de memória
Variável **escalar**

Vetores



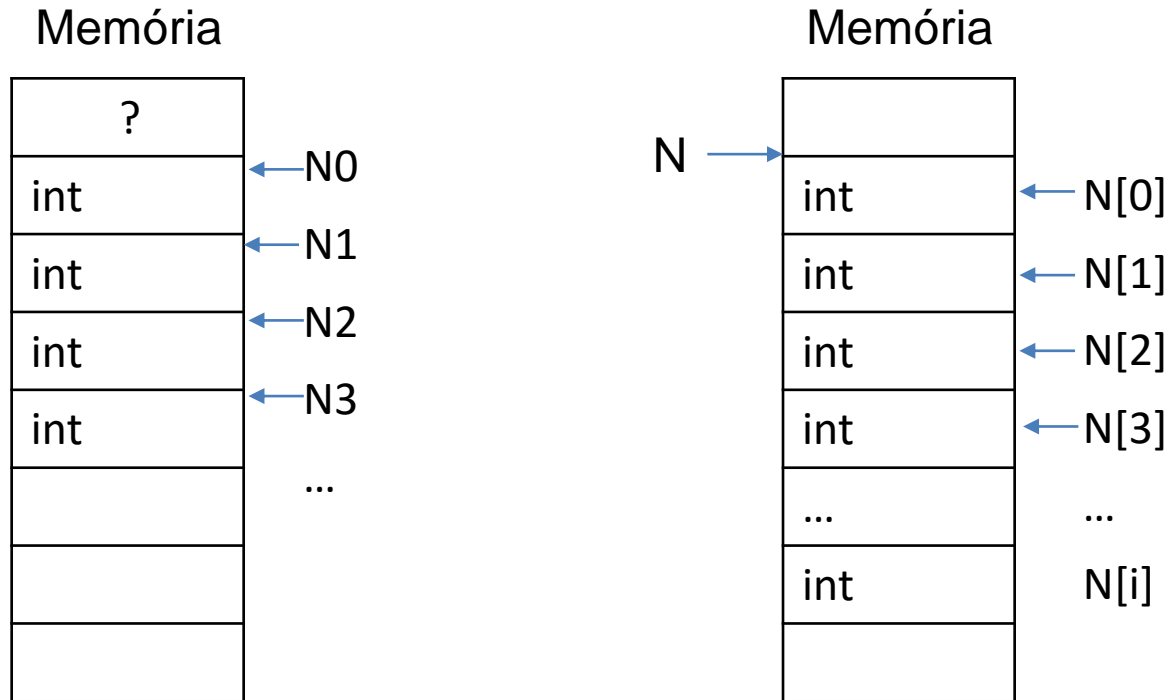
Posição de memória
Variável **escalar**



Posição de memória
Vetor de variáveis **escalares**

$$\text{Posição de memória} = \text{pos}(N) + \text{bytes}(\text{tipo}) * i$$

Vetores



Posição de memória = $\text{pos}(N) + \text{bytes}(\text{tipo}) * i$

EXEMPLO: $\text{bytes}(\text{int}) = 4$ (32 bits)

$\text{Posição_dado}(N[4]) = \#22 + 4 * 4 = \#32$

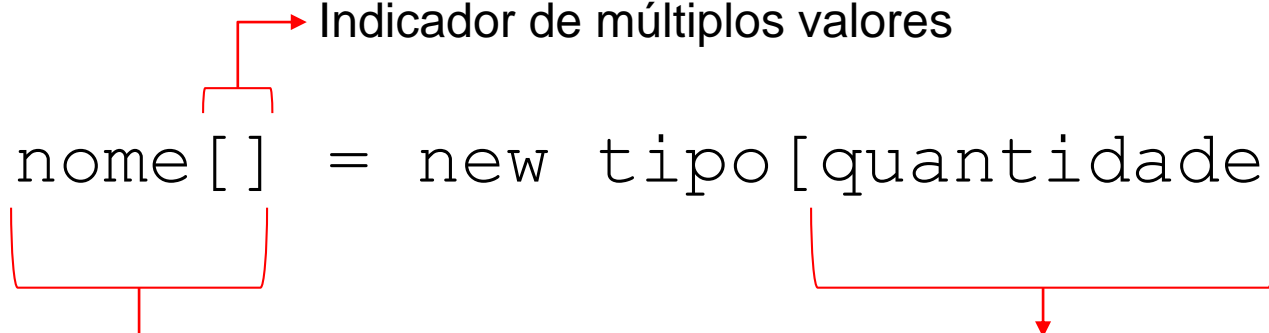
Sintaxe: regra geral

`tipo nome[] = new tipo[quantidade];`

Indicador de múltiplos valores

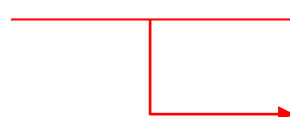
Vetor

Quantos valores do mesmo tipo



`nome[índice] = valor;`

Posição no vetor



Índice

- Índice
 - Pode ser um número inteiro ou uma variável inteira
 - Começa a contagem sempre no 0
 - Última posição = quantidade-1
 - Permite o uso de laços para variar o índiceIndica endereço de memória dentro do vetor
 - Ex: Tipo int tem 32 bits ou 4 bytes
Dado: **int A[] = new int[5];**
Posição de **A** na memória = **&A;**
&A[3] = &A + 3 * [tamanho de int]
 - No vetor, a manipulação é feita unicamente pelas variáveis internas do vetor, no caso A[posição].

Índice

- Exemplos:

```
int Z[] = new int[10]; // Aloca 10 valores inteiros para 'Z'
Z[0] = -10; // certo → posição 0 da memória de Z recebe -10
Z[7] = 21; // certo → posição 21 da memória de Z recebe 21
Z[10] = 0; // errado! → não há posição 10
Z=0; // errado! → Z não é um número, é um vetor
int i = 0; // Variável índice
Z[i++] = 11; // certo → posição 0 da memória de Z recebe 11
Z[i++] = 12; // certo → posição 1 da memória de Z recebe 12
Z[i] = 15; // certo → posição 3 da memória de Z recebe 15
Z[i] = 16; // certo → posição 3 da memória de Z recebe 16
```

Vetores em Java

- Alocação de memória == Declaração de variáveis
 - Formato geral para vetores[]:

```
tipo nome[] = new tipo[quantidade];
```

- Ex:

```
int x[] = new int[100]; // Aloca 100 valores inteiros
float r[] = new float[5]; // Aloca 5 valores reais
```

- Declaração e inicialização (aloca e inicializa as variáveis):

```
int j[] = {3, 2, 1, 0};
float q[] = {5f, 2f, 2.1f, -.5f, 0f};
String cor[] = {"vermelho", "azul", "verde", "preto"};
```

Exemplos

```
float q[] = {-4f, 2f, 2.1f, -.5f, 0f};  
int nElem= q.length;  
System.out.println(nElem);
```

5

```
String cor[]={"vermelho", "azul", "verde"};  
System.out.println(cor);
```

[Ljava.lang.String
;@28d93b30

```
for (int i=0;i<cor.length;i++)  
    System.out.println(cor[i]);
```

vermelho
azul
verde

Leitura e impressão

```
String nome[] = new String[5];
int idade[] = new int[5];

for (int i=0;i<5;i++) {
    nome[i] = leia("Entre com o nome " + i + ": ");
    idade[i] = leiaInt("Entre com a idade "+i + ": ");
}

System.out.printf("%32s%15s%15s\n",
    "Nome do candidato", "idade", "nascimento");

for (int i=0;i<5;i++)
    System.out.printf("%32s%15d%15d\n", nome[i],
        idade[i], (2014-idade[i]));
```

Métodos

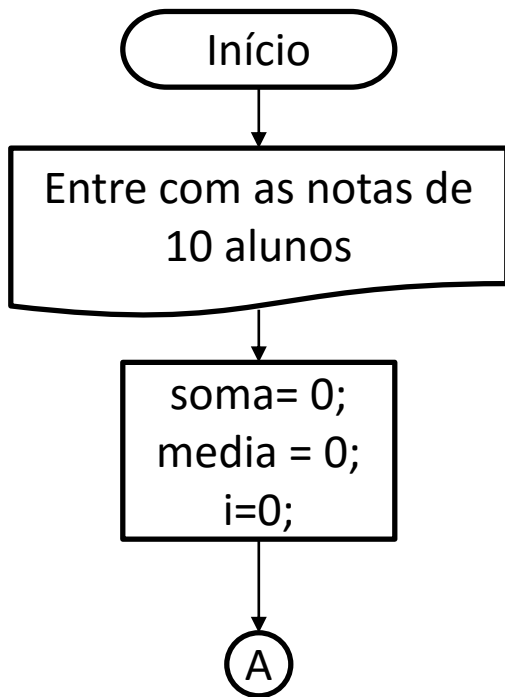
```
public static String leia(Object m) {  
    System.out.print(m);  
    return new java.util.Scanner(System.in).nextLine();  
}
```

```
public static int leiaInt(Object m) {  
    return Integer.parseInt(leia(m));  
}
```

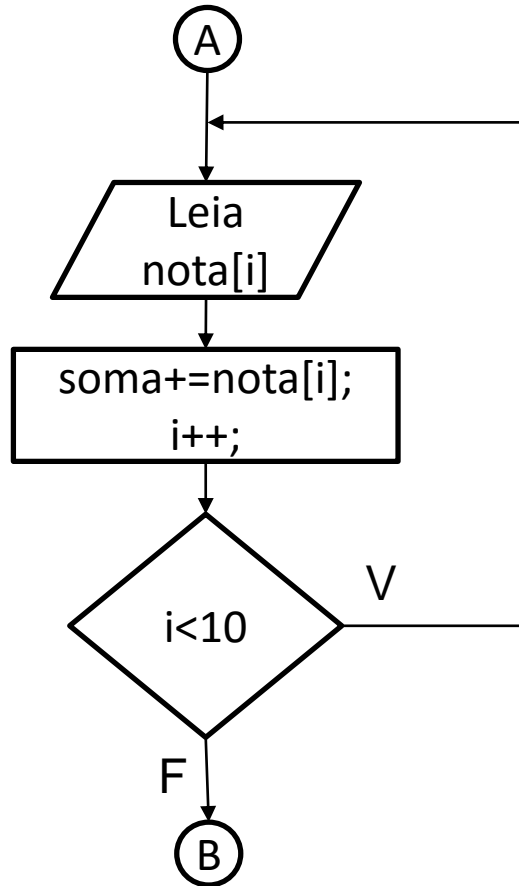
Prática

1. Média de 10 alunos

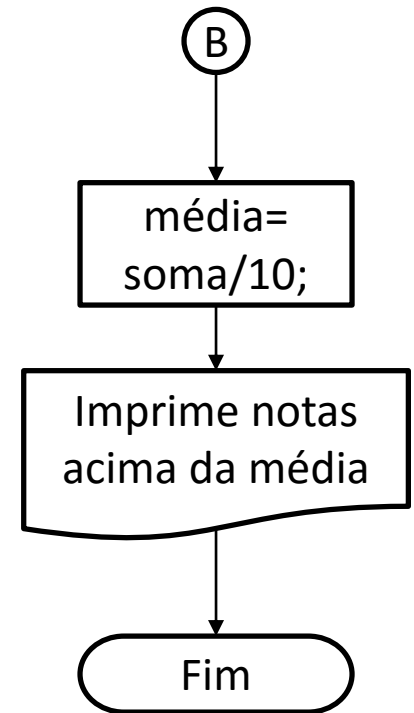
Fazer utilizando console



Sequencial



Repetição



Sequencial

2. Contando ocorrências

Faça um programa que receba um número do tipo `long` (inteiro longo) e conte quantos dígitos de zero a nove tem o número entrado.

Saída: quantidade de zeros a noves

Entrada: número inteiro longo

```
run:
Entre com um número: 123123123450
0: 1 ocorrências
1: 3 ocorrências
2: 3 ocorrências
3: 3 ocorrências
4: 1 ocorrências
5: 1 ocorrências
6: 0 ocorrências
7: 0 ocorrências
8: 0 ocorrências
9: 0 ocorrências
BUILD SUCCESSFUL (total time: 2 minutes 5
seconds)
```

Processamento:

```
digito = n%10;
n=n/10;
contador[digito]++;
```

3. Organizar

Faça um programa que leia 10 números inteiros em um vetor e, posteriormente imprima-os em uma linha só de pares, uma linha só de ímpares, o maior e o menor número entrados.

```
run:  
Entre com 10 números: 50 23 25 22 11 8 12  
Pares: 50 22 8 12  
Ímpares: 23 25 11  
Máximo: 50  
Mínimo: 8  
BUILD SUCCESSFUL (total time: 2 minutes 5  
seconds)
```