

A decorative wavy line in yellow and white on the left side of the slide.

AULA 12 – VETORES/ARRANJOS

SORAIA LÚCIA DA SILVA
PUC MINAS
ALGORITMOS E TÉCNICAS DE PROGRAMAÇÃO

MOTIVAÇÃO

- Exercício: Faça um programa que leia n números inteiros, calcule a média desses valores e mostre aqueles que forem maiores que a média.

```
int n, valor, media = 0;
```

```
ler: n;
```

```
for(int i = 0; i < n; i++){
```

```
    ler: valor;
```

```
    media += valor;
```

```
}
```

```
media /= n;
```

```
//Como mostrar os elementos maiores que a média?
```

```
//E agora José?
```

DEFINIÇÃO

O uso de arrays permite a manipulação de uma grande massa de dados a partir de pequenos trechos de código.

São variáveis compostas que correspondem a posições de memória identificadas por um mesmo nome, individualizadas por índices e cujo conteúdo é de mesmo tipo.

DEFINIÇÃO

- Os *arrays* são objetos; portanto, são considerados tipos por referência.
- Quando um array é criado, cada elemento recebe um valor padrão — **zero** para elementos de tipo primitivo numéricos, **false** para elementos booleanos.



SINTAXE

tipo [] nome_array = new **tipo** [tamanho];

- Exemplos:

1) **int** [] vet1 = new **int** [10];

2) **int** [] vet2;

vet2 = new **int** [10];

3) **int** [] vet3 = {5, 6, 20, 40, 2, 34, 87, 3, 1, 4};

Obs.: O valor inicial de **vet1** e **vet2** será zero e de **vet3** será os valores passados entre {}.

Ao inicializar uma variável de array, pode-se omitir a expressão **new** e o tamanho do array. O compilador calcula o tamanho a partir do número de inicializadores.

- Se um vetor tem tamanho ***n***, as posições válidas são de **0** a **(*n*-1)**.
- Tentar acessar posições negativas ou maiores que (*n*-1) ocasionam erros no programa.

```
int n = 10;  
int [] vet1 = new int[n];  
...  
for(int i = n - 1; i >= 0; i--)  
{  
    vet1[i-1] = vet1[i];  
}
```

```
int n = 10;  
int [] vet1 = new int[n];  
...  
for(int i = n; i >= 0; i--)  
{  
    escreva: vet1[i];  
}
```

TAMANHO DO ARRAY

Um objeto array conhece seu comprimento e armazena essas informações em uma **variável de instância** `length`.

`nome-do-vetor.length`

- Exemplo:

```
for (int i =0; i < valores.length; i++)  
{  
    ....;  
}
```

VETORES – ACESSO A POSIÇÕES

```
int [] vet = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 321;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1] --;
```

// Vetor pode ser indexado a partir de variáveis

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++)
```

```
{
```

```
    System.out.println("Digite um número:");
```

```
    vet[i] = sc.nextInt();
```

```
}
```


EXEMPLO

- Faça um programa que leia 10 números e os armazene em um array.

```
int n = 10;  
int[] vet = new int[n];  
  
for (int i = 0; i < n; i++) {  
    ler: vet[i];  
}
```

- ... e, em seguida, mostre cada número na tela

```
for (int i = 0; i < n; i++) {  
    escrever: "vet[" + i + "] = " + vet[i];  
}
```

```

Scanner sc = new Scanner (System.in);
public static void main(String [] args)
{
    int n, maior, i;
    int[ ] vet;

    System.out.println("Digite o tamanho do vetor: ");
    n = sc.nextInt();
    vet = new int[n];
    for (i = 0; i < n; i++)
    {
        System.out.println("Entre com um elemento: ");
        vet[i] = sc.nextInt();
    }
    maior = vet[0];
    for (i = 1; i < n; i++)
    {
        if (vet[i] > maior)
            maior = vet[i];
    }
    System.out.println("\nMaior elemento: " + maior);
}

```

Faça um programa que leia os elementos de um array de tamanho n e mostre o maior elemento do array.

MÉTODOS E PASSAGEM DE ARRAYS

- Um array pode ser passado como argumento a um método.
- Como o array é um tipo **reference**, será passada sua referência e alterações no array que é o parâmetro, no método chamado, são refletidas no respectivo array, que é o argumento, no método chamador.
- Para passar um argumento array para um método, especifique o nome do array sem usar colchetes.
- Exemplo:

```
int [ ] temperaturas = new int[20];
```

a chamada ao método:

```
conversaoTemp(temperaturas);
```

- Para que um método receba um array por meio de uma chamada a método, a lista de parâmetros do método deve especificar que um array será recebido.
- Exemplo:

```
conversaoTemp(int[ ] t);
```

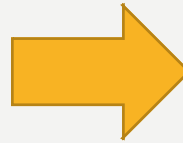
- Quando um argumento para o método for um array inteiro ou um elemento de array individual que não seja do tipo primitivo, a passagem é por REFERÊNCIA e NÃO por valor, ou seja, passa a referência à posição de memória do objeto.
 - As alterações realizadas no array dentro do método refletem no programa principal.
 - Ex.:
`int [] temperaturas = new int[20];`
A chamada ao método: `conversaoTemp(temperaturas);`
- Quando um argumento para o método for um elemento de array individual de um tipo primitivo, a passagem é por valor, ou seja, passa a CÓPIA daquele valor.
 - As alterações realizadas no elemento de array dentro do método NÃO refletem no programa principal.
 - Ex.:
`int [] temperaturas = new int[20];`
a chamada ao método: `conversaoTemp(temperaturas[3]);`

LAÇO “FOR DE ESTILO” MELHORADO

- Percorre o array inteiro automaticamente, obtendo um elemento de cada vez, em sequência, do início ao fim. Percorre do índice menor para o maior.

Ex.:

```
int [ ] nums = {1,2,3,4,5,6,7};  
int soma = 0;  
for (int i = 0; i<7; i++)  
    soma += nums [i];
```



Ex.:

```
int [ ] nums = {1,2,3,4,5,6,7};  
int soma = 0;  
for (int x : nums)  
    soma += x;
```

- Neste tipo de for, os elementos são somente “leitura”. Não podemos alterar o conteúdo do array.

```
Scanner sc = new Scanner (System.in);
```

```
class Somar
```

```
{
```

```
    public static int somar(int [ ] valores)
```

```
    {
```

```
        int somat = 0;
```

```
        for (int i =0; i < valores.length; i++)
```

```
        {
```

```
            somat = somat + valores[i];
```

```
        }
```

```
        valores[3] = 99;
```

```
        return somat;
```

```
    }
```

```
    public static void main (string[ ] args)
```

```
    {
```

```
        int [ ] pares = {2,4,6,8} ;
```

```
        int result;
```

```
        result = somar(pares);
```

```
        System.out.println("Total da soma: " + result);
```

```
        System.out.println("Valor na posicao pares[3]: " + pares[3]);
```

```
    }
```

```
}
```

Exemplo de array inteiro passado por referência.

```
Total da soma: 20  
Valor na posicao pares[3]: 99
```

```
Scanner sc = new Scanner (System.in);  
class Somar
```

```
{  
    public static void dobrar(int elemento)  
    {  
        elemento *=2;  
        System.out.println("Resultado no método" + elemento);  
    }  
    public static void main (string[ ] args)  
    {  
        int [ ] pares = {2,4,6,8} ;  
        System.out.println("Resultado antes da chamada" + pares[3]);  
        dobrar(pares[3]);  
        System.out.println("Valor na posicao pares[3]: " + pares[3]);  
    }  
}
```

**Exemplo de elemento
de array passado por
valor.**


```
class MainClass
```

```
{
```

```
    public static int[ ] multiplicar (int n)
```

```
    {
```

```
        int [ ] valores1 = {2,4,6,8} ;
```

```
        for (int i =0; i < valores1.length; i++)
```

```
        {
```

```
            valores1[i] = valores1[i] * n;
```

```
        }
```

```
        return valores1;
```

```
    }
```

```
    public static void main (string[] args)
```

```
    {
```

```
        int [ ] valores2;
```

```
        valores2 = multiplicar (3);
```

```
        for (int i =0; i < valores2.Length; i++)
```

```
        {
```

```
            System.out.println ("Valores2: " + valores2[i]);
```

```
        }
```

```
    }
```

```
}
```

Arrays como retorno

```
Valores2: 6
```

```
Valores2: 12
```

```
Valores2: 18
```

```
Valores2: 24
```

```
Press any key to continue...
```

```
public static int[] soma_Vet (int[] X, int[] Y)
```

```
{  
    int i;  
    int[] vet = new int[10];  
  
    for (i = 0; i < 10; i++)  
        vet[i] = X[i] + Y[i];  
    return vet;  
}
```

Passagem de parâmetro



```
static void main(String[] args)
```

```
{  
    int[] vetA = new int[10];  
    int[] vetB = new int[10];  
    int[] vetC = new int[10];  
    int i;  
  
    for (i = 0; i < 10; i++)  
    {  
        System.out.println("Digite o elemento" + i + 1 + "de A:");  
        vetA[i] = sc.nextInt();  
    }  
    for (i = 0; i < 10; i++)  
    {  
        System.out.println (" Digite o elemento" + i + 1 + "de B:");  
        vetB[i] = sc.nextInt();  
    }  
    vetC = soma_Vet(vetA, vetB);  
    for (i = 0; i < 10; i++)  
        System.out.println("Elemento" + i + 1 + "de C é", vetC[i]);  
}
```

EXERCÍCIOS

- 1) Faça um programa que leia um vetor A de 10 inteiros e crie um método que receba este vetor e retorne a soma dos elementos elevados ao quadrado.
- 2) Faça um programa que leia os elementos de um array de tamanho n e mostre o maior e o menor elementos do array.
- 3) Faça um programa para ler um número inteiro N e N elementos de um *array*. Em seguida, encontre a posição do menor elemento.