CIÊNCIAS DA COMPUTAÇÃO

FACULDADE DE CIÊNCIAS | UNIVERSIDADE AGOSTINHO NETO

FUNDAMENTOS DE PROGRAMAÇÃO

Docente:

- ✓ Lufialuiso Sampaio Velho, MSc.

Monitor

- ✓ João Pedro

Conteúdo

Cap. V

Arrays(vectores e matrizes)

CAP. V - Arrays

❑ Motivação

- O número de variáveis que declaramos reflecte de certa forma a quantidade de elementos que desejamos manipular. Imagine que deseja armazenar três notas de um estudante?

- Neste caso o mais comum é declarar uma variável do tipo String para o Nome e três variáveis do tipo Float para as notas. Agora imagine o que seria para armazenar dados de 50 estudantes cada um com três notas? Declararias 150 variáveis?

❑ Motivação (cont.)

- Ou ainda como resolveria a leitura de 20 números inteiros e que se deseja obter o maior e o menor destes? Declararias 20 variáveis?

- Claro que não 150 ou 20 variáveis normais. Mas sim **vectores ou matrizes**.

CAP. V - Arrays

❑ Definição

- Array é uma colecção de variáveis do mesmo tipo, acessíveis com um único nome e armazenadas de forma contínua na memória.

- Em Java os índices de um vector com n elementos variam sempre entre 0 e n – 1.

v1 =

0	1	2	3
Lukau	Paulino	Valdanio	Solange

Representação de um Array de String com 4 elementos.

v2 =

0	1	2	3
20	13	9	27

Representação de um Array de Inteiro com 4 elementos.

CAP. V - Arrays

❑ Conceitos em Java

- Arrays em Java são objectos, portanto são considerados tipos por referência.
- Os elementos de um Array podem ser de tipos primitivos ou tipos por referencia.
- Para referenciar a um elemento do Array é necessário colocar o nome da variável seguido de um ou mais índices entre colchetes [ ].
- O índice de um Array deve ser um valor inteiro não negativo.

❑ Arrays podem ter uma ou várias dimensões:

- Vector = 1 dimensão
- Matriz = mais do que uma dimensão

CAP. V. Arrays - Declaração

- Em Java, um vector é declarado de várias formas, obedecendo a seguinte sintaxe:

```
tipo_dado [ ] nome_vector = new tipo_dado [dimensão];
```

  - tipo_dado – o tipo de dado do vector (tipo primitivo ou tipo por referencia).
  - nome_vector – o nome da variavel Array.
  - Dimensão – o tamanho do vector definido em valor inteiro positivo.

- Ex: `int [] num = new int [5];`

  - Implica que a variável **num** tem a capacidade para armazenar 5 elementos; a sua primeira posição é 0 e a última é 4.
  - Com esta opção, todas as posições de **num** são inicializadas com Zero (0) tendo em conta o seu tipo (int).

- Pode ser igualmente declarado e inicializado da seguinte maneira:

```
int [] num;
num = new int [5];
```

CAP. V. Arrays - Declaração

- Inicialização explícita: a inicialização é feita atribuindo valores de forma explícita ao vector utilizando { }.

```
Ex1: int num[] = {2,6,8,7,5};
```

- O vector num possui a dimensão 5.
- Na sua primeira posição num[0] encontramos o valor 2, num[1]=6, num[2]=8, num[3]=7 , num[4]=5.

```
Ex2: double[] m = {};
```

- Para o exemplo 2, não existe alocação de espaço; isto implica que não podemos aceder a nenhuma posição do vector.

CAP. V. Arrays - Manipulação

❑ Atribuição Direita

```
String nomes[]=new String [2];
nomes[0]="Eliana";
nomes[1]="Francisco";
```

❑ Leitura de dados:

```
int numeros = new int [5];
for (int i=0;i< numeros.length;i++){
     numeros[i]= teclado.nextInt();
}
```

❑ Escrita:

```
for (int i=0; i < numeros.length; i++){
     System.ou.println(numeros[i]);
}
```

OU:

```
for (int numero : numeros ){
     System.ou.println(numero);
}
```

- O método length retorna a dimensão do vector. (Ex: `numeros.length`)

CAP. V. Arrays - Manipulação

- EX: Programa em Java que lê os 10 primeiros números digitados pelo utilizador e mostra na tela.

Resolução

```
public class Fundamentos {
    public static void main(String[] args) {

        Scanner teclado = new Scanner(System.in);
        final int MAX = 10;
        float numeros[] = new float[MAX];

       /*Ler o números */
        for (int i = 0; i < numeros.length; i++) {
            System.out.print("\nDigite o número: ");
            numeros[i] = teclado.nextFloat();
        }

        /*Mostrar os números armazenados no vector */
        for (int i = 0; i < numeros.length; i++) {
            System.out.print("\nPosição " + i + ": " + numeros[i]);
        }
    }
}
```

CAP. V. Arrays - Exercícios

1. Faça um programa que recebe uma quantidade de números determinados pelo utilizador e imprime-os de forma inversa.

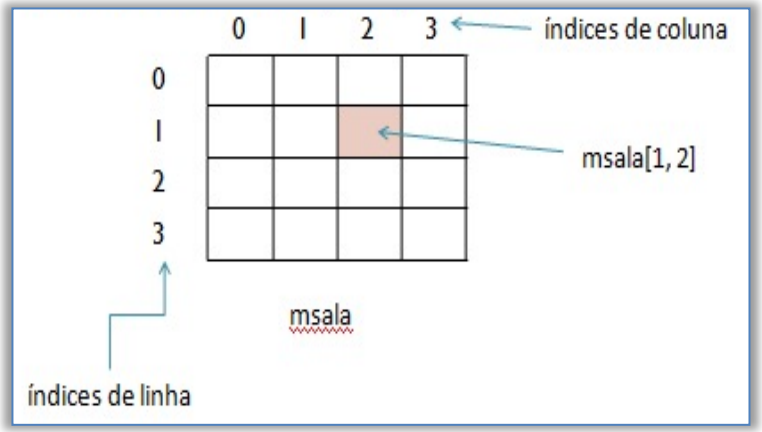
2. Faça um programa em Java que lê 20 números inteiros e mostra o maior e o menor.

3. Faça um programa que lê 30 números e mostra o vector ordenado em forma decrescente.

Até a próxima Aula

CAP. V. Arrays - Multidimensional

- Os Arrays multidimensional também denominado de matriz comportam mais do que uma dimensão organizada na forma linhas e colunas.



- Sintaxe

```
tipo_dado [][] nome_matriz = new tipo_dado [L][C];
```

- tipo_dado – o tipo de dado do array (tipo primitivo ou tipo por referencia).
- nome_matriz – o nome da variavel Array.
- L– a quantidade de linhas da matriz.
- C– a quantidade de colunas da matriz.

CAP. V. Arrays - Multidimensional

- Atribuição Direita

```
int num [][] = new int[2][2];
num[0][0] = 23;
num[0][1] = 33;
```

- Leitura de dados:

```
int num[][] = new int[2][2];

for (int linha = 0; linha < num.length; linha++) {
     for (int coluna = 0; coluna < num[linha].length; coluna++) {
          num[linha][coluna] = teclado.nextInt();
     }
}
```

- Escrita:

```
for (int linha = 0; linha < num.length; linha++) {
     for (int coluna = 0; coluna < num[linha].length; coluna++) {
          System.out.println(num[linha][coluna]);
     }
}
```

CAP. V. Arrays - Multidimensional

- EX: Programa em Java que armazena numa matriz 2x2 os 4 primeiros numeros inteiro digitado, e de seguida exibe os números na tela.

Resolução

```
public class Fundamentos {
    public static void main(String[] args) {

        Scanner teclado = new Scanner(System.in);
        int num[][] = new int[2][2];

        /*ler os números */
        for (int linha = 0; linha < num.length; linha++) { //ciclo da linha
            for (int coluna = 0; coluna < num[linha].length; coluna++){ //ciclo da coluna
                System.out.println("Digite o número: ");
                num[linha][coluna] = teclado.nextInt();
            }
        }

        /*mostrar os números armazenados na matriz */
        for (int linha = 0; linha < num.length; linha++) { //ciclo da linha
            for (int coluna = 0; coluna < num[linha].length; coluna++){ //ciclo da coluna
                System.out.println(num[linha][coluna]);
            }
        }
    }
}
```

CAP. V. ArrayList

- Os arrays(Vectores e Matrizes) possuem um tamanho fixo, ou seja não alteram o seu tamanho em tempo de execução.
- ArrayList: é uma classe de colecção em java que permite armazenar diversos valores de forma dinâmica.
- Sintaxe:

```
ArrayList <Tipo> nome_do_ArrayList = new ArrayList<Tipo>();
```

  - Tipo: é o tipo do ArrayList, não pode ser tipo primitivo, mas pode ser um tipo por referencia(String, Integer, Float, Double, Character, entre outros)
  - nome_do_ArrayList: corresponde ao nome da variavel que armazenará o conjunto de valores do tipo definido.

- Exemplo

```
ArrayList <String> nomes = new ArrayList<String>(); // ArrayList de String
ArrayList <Double> pesos = new ArrayList<Double>(); // ArrayList de float
ArrayList <Integer> idades = new ArrayList<Integer>(); // ArrayList de int
```

CAP. V. ArrayList – Métodos

- Para a inserir bem como para remover elementos existente em um ArrayList faz-se recursos de métodos:

```
ArrayList <String> nomes = new ArrayList();
```

Métodos	Descrição	Exemplo

add()	Adiciona um novo elemento no fim ou numa posição do ArrayList.	`nomes.add("Nsimba");`
		`nomes.add("Gabriel");`
		`nomes.add(0,"Malengue");`

set()	Altera o elemento encontrado na posição especificada por um novo elemento.	`nomes.set(1,"Mariano");`
contains()	Retorna true se ArrayList contiver o elemento, caso não retorna false.	`nomes.contains("Nsimba);`
get()	Retorna o elemento localizado no índice especificado.	`nomes.get(1);`
indexOf()	Retorna o índice da primeira ocorrência do valor especificado.	`nomes.indexOf("Gabriel");`
remove()	Remove a primeira ocorrência do valor especificado ou o elemento localizado no índice especificado.	`nomes.remove(1);` `nomes.remove("Malengue");`
size()	Retorna o numero de elementos armazenados.	`nomes.size();`

CAP. V. ArrayList – Manipulação

❑ Adicionar elementos- add():

```
ArrayList <String> nomes = new ArrayList();
nomes.add("Nsimba");
nomes.add("Anacleto");
nomes.add(0, "Malengue");
```

❑ Alterar elementos- set():

```
nomes.set(0, "Lukau");
nomes.set(1, "Paulino");
nomes.add(2, "Kondo");
```

❑ Imprimir elementos – get():

```
for (int i = 0; i < nomes.size(); i++) {
    System.out.println(nomes.get(i));
}
```

```
for (String nome : nomes ) {
    System.out.println( nome);
}
```

OU:

❑ A utilização da colecção ArrayList requer uma importação previa:

- `import java.util.ArrayList;`

CAP. V. ArrayList – Manipulação

- EX: Programa em Java que lê infinitamente números digitados pelo utilizador e mostra na tela. O programa termina se for digitado o numero -1.

Resolução

```
import java.util.Scanner;
import java.util.ArrayList;
public class Fundamentos {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        ArrayList<Integer> numeros = new ArrayList<Integer>();
        int num;
        while (true) {
            System.out.println("Digite um numero: ");
            num = input.nextInt();
            if (num == -1)
                break;
            else
                numeros.add(num);
        }

        for (Integer numero : numeros) {
            System.out.println( numero );
        }
    }
```

Até a próxima Aula