Programação Orientada a Objetos

Aula 04 – Estruturas de Dados Homogêneas: Vetores e Matrizes

Prof. Luiz Mário Lustosa Pascoal

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)
  - Imprimir elementos de um vetor (usando laços ou não)

Problema 1

- Calcular a média de uma classe a partir da nota de seus 10 alunos.

- LÓGICA (SOLUÇÃO)
  - Para cada um dos 10 alunos:
    - Ler a nota N
    - Acumular a nota (somar com as anteriores)
  - Media = soma / 10

ATENÇÃO: a cada nota digitada vai acumulando com a anterior. Pode-se usar a mesma variável "N" para ler a nota do aluno seguinte

N	N	N	N	N	N	N	N	N	N
5.5	6.5	8.0	3,0	7.5	2.5	7.5	6.0	4.5	10.0

SOMA	MEDIA
61,0	6,1

Problema 2

- Calcular a média de uma classe a partir da nota de seus 10 alunos **e verificar quantos conseguiram nota acima da média da classe:**

- **LÓGICA (SOLUÇÃO)**
  - Ler as 10 notas
  - Somar as 10 notas e dividir por 10
  - **Para cada uma das 10 notas faça:**
    - **SE nota > media**
      ENTÃO

      contar

ATENÇÃO: Note que é preciso armazenar os 10 valores, para que depois de calculada a média se possa verificar se cada uma das 10 notas estão acima da média

N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
5,5	6,5	8,0	3,0	7,5	2,5	7,5	6,0	4,5	10,0

SOMA
61,0

MEDIA
6,1

- SE N1 > media ENTÃO cont = cont + 1
- SE N2 > media ENTÃO cont = cont + 1
- SE N3 > media ENTÃO cont = cont + 1
- ...
- SE N9 > media ENTÃO cont = cont + 1
- SE N10 > media ENTÃO cont = cont + 1

Acima da Média
N2, N3, N5, N7, N10

5 notas acima da média

Problema

E se fosse uma turma de 500 alunos ? 500 variáveis ?
500 "ifs" ?

A maioria das linguagens implementam variáveis do tipo CONJUNTO, chamadas de VETOR ou ARRAY

VETOR

(ARRAY)

Arrays – Declaração

- Um array em Java é uma estrutura de dados que permite o armazenamento de um conjunto de variáveis (elementos) de um mesmo tipo (ou referências para instâncias de uma mesma classe) e que são acessadas individualmente por um índice

```
char[] letrasAlfabeto = {'a', 'b', 'c', 'd', 'e', 'f', 'g',
                         'h', 'i', 'j', 'k', 'l', 'm', 'n',
                         'o', 'p', 'q', 'r', 's', 't', 'u',
                         'v', 'w','x', 'y', 'z'};

letrasAlfabeto[0] ➔ a

letrasAlfabeto[1] ➔ b

...

letrasAlfabeto[25] ➔ z
```

Array – Usos mais comuns

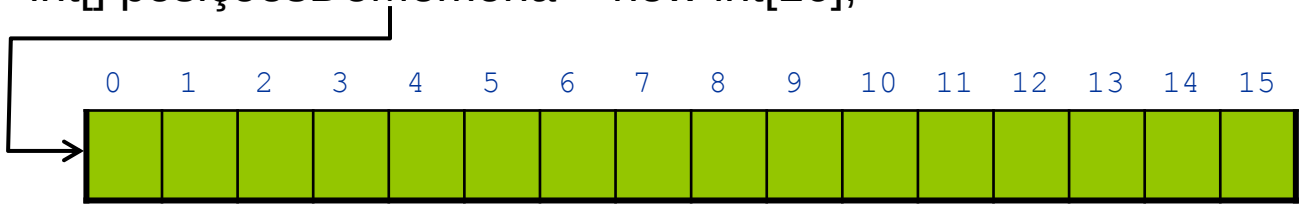
- Armazenar grandes quantidades de dados de um mesmo tipo ou classe
  - 1440 amostras do tipo double da temperatura de um paciente internado, medida a cada minuto, durante um dia
  - As 125 instâncias da classe Aluno, componentes de uma turma
- Utilizar variáveis individuais e com nomes distintos em casos como estes é extremamente trabalhoso e sujeito a erros
  - O uso de arrays permite usar um único nome para denotar um conjunto homogêneo de variáveis, que são acessadas individualmente através de **índices**

Declaração e alocação de arrays unidimensionais

- Java adota a notação de colchetes para declarar arrays
  - Uma variável de um tipo (classe) específico seguido de um par de colchetes declara uma referência a um array de elementos desse tipo
    - int[] posiçõesDeMemória;
    - char letrasAlfabeto[];
    - double[] medidaTemperatura;
  - A declaração acima define apenas referências para arrays, mas não cria nenhuma variável do tipo array na memória
    - É preciso criar (alocar) o array na memória com um número predeterminado de posições
    - Opcionalmente, pode-se inicializá-lo na cláusula de declaração

Declaração e alocação de arrays unidimensionais

- Alocação de arrays
  - Cria a variável correspondente ao array na memória com para um número de elementos predeterminado
  - Comando realizado pela palavra-chave new, seguida pelo tipo do dado do array e do número de elementos a alocar, entre colchetes
    - int[] posiçõesDeMemória = new int[16];



    - int tamanho = 32768;
    - posiçõesDeMemória = new int[tamanho];
    - double[] medidasTemperatura = new int[24 * 60 * 60];

Declaração e alocação de arrays unidimensionais

- Inicialização
  - Pode ser realizada na cláusula de declaração, para pequeno número de elementos conhecidos a priori
    - int[] ultimoDiaMes = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    - String[] universidades = {“FASAM", “UFG", “DELTA", "PUC"};
  - Pode ser realizada após a alocação/criação do array
    - String[] universidades = new String[4];
    - universidades[0] = “FASAM";
      universidades[1] = "UFG";
      universidades[2] = “DELTA";
      universidades[3] = "PUC";

Arrays: acesso a elementos

- O índice para acesso a um elemento deverá ser um valor do tipo inteiro entre 0 e o tamanho do array subtraído de 1
  - nome_do_array[indice]
- Qualquer tentativa de acesso com valor de índice fora dessa faixa resulta em erro em tempo de execução, com a interrupção da execução do programa, indicado pela exceção
  - ArrayIndexOutOfBoundsException
- O tamanho – quantidade de elementos – de um array pode ser lido através do atributo **length**, existente em todo array
  - universidades.length -> 4
  - ultimoDiaMes.length -> 12

Cópia de arrays

- Cópia de Arrays – cópia de referência
  - int[] primos = { 2, 3, 5, 7, 11, 13};
  - int[] cunhados = primos;
  - cunhados[3] = 100;
  - System.out.println(“primos[3] = “, primos[3]);
- Cópia de Arrays – cópia de conteúdo
  - System.arraycopy(from, fromindex, to, toindex, count);

Retomando o problema das notas...

Para o problema de armazenar 10 notas, pode-se definir uma variável de tamanho 10 do tipo float.

```
double[] nota = new double [10];
```

	NOTA									
CONTEÚDO	5,5	6,5	8,0	3,0	7,5	2,5	7,5	6,0	4,5	10,0
ÍNDICE	0	1	2	3	4	5	6	7	8	9

A declaração acima cria em memória uma variável com 10 posições do tipo float

As 10 posições são automaticamente numeradas de 0 a 9 (índice);

Para acessar cada posição deve-se usar o nome da variável e a sua posição ou índice (entre colchetes)

- A instrução abaixo imprime a nota que está na posição 3 da variável.

```
System.out.println("A nota armazenada na posição 3 é: "+ nota[3]);
```

Vetores

- Como percorrer um vetor:

Uma forma prática de percorrer um vetor é usando um laço:

```
for (int i = 0; i < 10; i++)
{
    System.out.printf("A nota na posicao %d é: %.2f", i , nota[i]);
}
```

Problema 2

- Calcular a média de uma classe a partir da nota de seus 10 alunos e verificar quantos conseguiram nota acima da média da classe:

N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
5,5	6,5	8,0	3,0	7,5	2,5	7,5	6,0	4,5	10,0

SOMA	MEDIA
61,0	6,1

- SE N1 > media ENTÃO cont = cont + 1
- SE N2 > media ENTÃO cont = cont + 1
- SE N3 > media ENTÃO cont = cont + 1
- ...
- SE N9 > media ENTÃO cont = cont + 1
- SE N10 > media ENTÃO cont = cont + 1

Acima da Média
N2, N3, N5, N7, N10

5 notas acima da média

Solução do Problema 2

```
double[] notas = new double [10];
double soma = 0.0, mediaTurma;
int i, cont = 0;
Scanner leitura = new Scanner(System.in);

for (i = 0; i < notas.length; i++) {
    System.out.println("Digite a nota do Aluno " + (i + 1) + " : ");
    notas[i] = leitura.nextDouble();

    soma = soma + notas[i];
}

mediaTurma = soma / notas.length;

for (i = 0; i < notas.length; i++) {
    if (notas[i] > mediaTurma)
        cont++;
}

System.out.println("Media da turma: "+ mediaTurma);
System.out.println("Existem "+ cont + " alunos com notas acima da media da turma");
```

E se fosse uma turma de 500 alunos ?

Arrays multidimensionais

- Elementos são acessados por um número arbitrário de índices
- Matrizes matemáticas

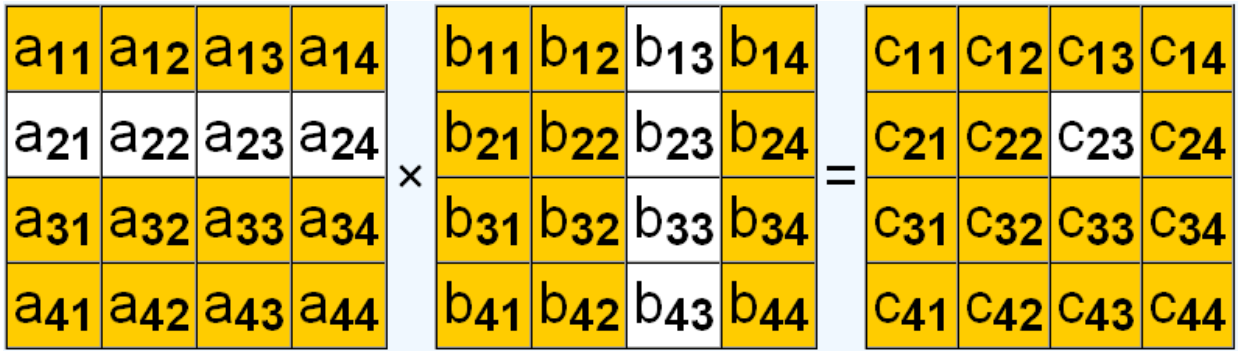
```
double[][] matriz = new double[5][10];
double maior = maiorValor(matriz);
...

 public double maiorValor(double[] m) {
   double maiorAtéAgora = m [0][0];
   for (int lin= 0;lin < m.length; lin++)
      for (int col = 0; col < m [lin].length; col++)
        if (m [lin][col] > maiorAtéAgora)
          maiorAtéAgora = m [lin][col];
   return maiorAtéAgora;
 }
```

Exemplo: Multiplicação de matrizes

- Suponha duas matrizes A(x,y) e B(m,n)
  - A . B só é possível se e somente se y = m
    - A matriz resultante seria do tipo x,n
  - B . A só é possível se e somente se n = x
    - A matriz resultante seria do tipo m,y

$$C_{ij} = \sum_{k=0}^{N-1} A_{ik} . B_{kj}$$



Multiplicação Matrizes em Java

```
double[][] matrixA = {{3.0,2.0,-1.0},{0.0,4.0,6.0}};
double[][] matrixB = {{1.0,0.0},{5.0,3.0},{6.0, 4.0}};
double[][] matrixC = new double[2][2];

for(int i=0; i<2; i++){
    for(int j=0; j<2; j++){
        for(int k=0; k<3; k++){
            matrixC[i][j] += matrixA[i][k]*matrixB[k][j];
        }
    }
}
```

Repetição: for tradicional

```
public void somaVetor(){
    int[] vetor = {87, 68, 94, 100, 68, 39, 10};

    int total = 0;
    for (int i=0; i < vetor.length; i++){
        total = total + vetor[i];
    }
    System.out.println("Total = " + total);
}
```

Repetição: for alterado

```
public void somaVetor(){
    int[] vetor = {87, 68, 94, 100, 68, 39, 10};

    int total = 0;
    for (int valor: vetor){
        total = total + valor;
    }
    System.out.println("Total = " + total);
}
```

EXERCÍCIOS

Pra você Resolver!

Exercício 1

- Preencher um vetor com números inteiros (8 unidades); solicitar um número do teclado. Pesquisar se esse número existe no vetor. Se existir, imprimir em qual posição do vetor. Se não existir, imprimir MSG que não existe.

Solução do Exercício 1.

```
public static void main(String[] args) {
    // TODO Auto-generated method stub


    int vetor[] = new int[8], num;
    boolean achei = false;

    Scanner leitura = new Scanner(System.in);

    for (int i = 0; i < vetor.length; i++) {
        System.out.printf("\nDigite um número[%d]: ", i);
        vetor[i] = leitura.nextInt();
    }

    System.out.print("\n\n Digite o valor a ser pesquisado no vetor: ");
    num = leitura.nextInt();

    for (int i = 0; i < vetor.length; i++) {
        if (vetor[i] == num)
        {
            System.out.printf("\n O numero %d está no vetor na posicao %d", num, i);
            achei = true;
            break;
        }
    }

    if(!achei)
        System.out.println("O numero " + num + " não está no vetor");

}
```