APOSTILA DO ALUNO · LINGUAGENS DE PROGRAMAÇÃO (JAVA)

Aula 2 — Arrays de Duas Dimensões (Matrizes)

Organizando dados em linhas e colunas com Java

Unidade: Arrays em Java

Nome do aluno: _____

Turma: _____

Data: _____

1. Objetivos desta aula

Ao final desta apostila, você deverá ser capaz de:

- Diferenciar um array de uma dimensão (vetor) de um array de duas dimensões (matriz).
- Criar e inicializar uma matriz em Java.
- Acessar e alterar um elemento usando matriz[linha][coluna].
- Percorrer uma matriz inteira usando dois laços for aninhados (for duplo).
- Aplicar operações comuns sobre matrizes: soma, contagem de pares/ímpares, maior e menor valor, soma por linha e por coluna, diagonais.
- Resolver um problema prático organizando dados em uma matriz (ex.: notas de alunos).

2. Revisão: arrays de uma dimensão (vetores)

Até agora, você trabalhou com arrays de uma dimensão, também chamados de vetores. Um vetor guarda vários valores em uma única estrutura, organizados em uma linha:

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

Podemos imaginar esse vetor assim:

Índice	0	1	2	3	4
Valores	10	20	30	40	50

Mas e quando precisamos organizar informações em linhas e colunas — como uma tabela de notas? Para isso usamos um array de duas dimensões, normalmente chamado de matriz.

3. O que é uma matriz?

Uma matriz é uma estrutura que organiza dados em linhas e colunas. Podemos representá-la assim:

	Coluna 0	Coluna 1	Coluna 2
Linha 0	10	20	30
Linha 1	40	50	60
Linha 2	70	80	90

Essa matriz possui 3 linhas × 3 colunas, portanto 3 × 3 = 9 elementos.

4. Matriz × vetor

O vetor possui uma dimensão; a matriz possui duas. Podemos pensar assim:

```
VETOR
[10] [20] [30] [40] [50]

MATRIZ
[10] [20] [30]
[40] [50] [60]
[70] [80] [90]
```

> 💡 **Regra para memorizar**
>
> O vetor precisa de um índice. A matriz precisa de dois índices: linha e coluna.

5. Criando uma matriz em Java

Para criar uma matriz usamos dois pares de colchetes:

```
int[][] matriz;
```

Depois reservamos espaço para ela:

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

Ou fazemos tudo em uma única linha:

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

Nesse exemplo temos 3 linhas e 3 colunas, portanto 3 × 3 = 9 posições.

6. Entendendo os índices

Assim como nos vetores, os índices de linha e de coluna começam sempre em zero. Na matriz 3×3, a primeira linha é a linha 0, a segunda é a linha 1 e a terceira é a linha 2 — o mesmo vale para as colunas.

7. Acessando um elemento

Para acessar um elemento da matriz utilizamos:

```
matriz[linha][coluna]
```

Por exemplo, matriz[0][0] significa "linha 0, coluna 0". Na matriz do exemplo:

```
System.out.println(matriz[0][0]); // 10
```

8. Alterando um elemento

Podemos alterar uma posição da matriz da mesma forma que fazemos com vetores:

```
matriz[1][1] = 500;
```

Antes: 10 20 30 / 40 50 60 / 70 80 90 → Depois: 10 20 30 / 40 500 60 / 70 80 90.

9. Inicializando uma matriz diretamente

Podemos criar uma matriz já preenchida. Cada conjunto entre chaves { } representa uma linha:

```
int[][] matriz = {
    {10, 20, 30},
    {40, 50, 60},
    {70, 80, 90}
};
```

10. Percorrendo uma matriz

Esta é uma das partes mais importantes deste conteúdo. Para percorrer um vetor usamos um for; para percorrer uma matriz, precisamos de dois for — um dentro do outro:

```
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        System.out.println(matriz[i][j]);
    }
}
```

> 💡 **Regra para memorizar**
>
> O primeiro for controla as linhas. O segundo for controla as colunas.

11. Exibindo a matriz na tela

Dentro do segundo for usamos System.out.print() para que os elementos da mesma linha fiquem juntos. Depois de terminar uma linha, usamos System.out.println() para quebrar a linha:

```
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        System.out.print(matriz[i][j] + " ");
    }
    System.out.println();
}
```

Resultado: 10 20 30 / 40 50 60 / 70 80 90 (cada linha da matriz em uma linha da tela).

12. Entendendo length em uma matriz

matriz.length representa a quantidade de linhas. matriz[i].length representa a quantidade de colunas da linha i. Por exemplo:

```
int[][] matriz = new int[4][5];
// matriz.length    → 4 (4 linhas)
// matriz[i].length → 5 (5 colunas)
```

13. Preenchendo uma matriz com Scanner

Podemos solicitar os valores ao usuário, identificando linha e coluna na mensagem:

```
Scanner entrada = new Scanner(System.in);
int[][] matriz = new int[3][3];

for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        System.out.print(
            "Digite o valor da linha " + i +
            " e coluna " + j + ": "
        );
        matriz[i][j] = entrada.nextInt();
    }
}
```

14. Matriz com números aleatórios (Random)

Também podemos preencher uma matriz com valores aleatórios, usando a classe Random — útil para testar um programa sem precisar digitar cada valor:

```
Random aleatorio = new Random();
int[][] matriz = new int[3][3];
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        matriz[i][j] = aleatorio.nextInt(100);
    }
}
```

15. Exemplo completo resolvido

Programa que preenche uma matriz 3×3 pelo teclado, identificando linha e coluna, e depois exibe a matriz formatada:

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner entrada = new Scanner(System.in);
        int[][] matriz = new int[3][3];

        for (int i = 0; i < matriz.length; i++) {
            for (int j = 0; j < matriz[i].length; j++) {
                System.out.print(
                    "Digite o valor da linha " + i +
                    " e coluna " + j + ": "
                );
                matriz[i][j] = entrada.nextInt();
            }
        }

        System.out.println("\nMatriz:");
        for (int i = 0; i < matriz.length; i++) {
            for (int j = 0; j < matriz[i].length; j++) {
                System.out.print(matriz[i][j] + "\t");
            }
            System.out.println();
        }
        entrada.close();
    }
}
```

16. Somando todos os elementos

Podemos percorrer toda a matriz e acumular os valores em uma variável de soma, iniciada em zero:

```
int soma = 0;
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        soma += matriz[i][j];
    }
}
System.out.println("Soma: " + soma);
```

17. Contando números pares e ímpares

Podemos combinar o percurso da matriz com uma estrutura condicional. Lembre-se: numero % 2 == 0 significa que o número é par.

```
int quantidadePares = 0;
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        if (matriz[i][j] % 2 == 0) {
            quantidadePares++;
        }
    }
}
```

Para contar ímpares, basta trocar a condição para matriz[i][j] % 2 != 0.

18. Encontrando o maior e o menor valor

Começamos considerando o primeiro elemento da matriz como o maior (ou o menor) e comparamos com os demais durante o percurso:

```
int maior = matriz[0][0];
for (int i = 0; i < matriz.length; i++) {
    for (int j = 0; j < matriz[i].length; j++) {
        if (matriz[i][j] > maior) {
            maior = matriz[i][j];
        }
    }
}
```

A lógica para o menor valor é semelhante, trocando > por <.

19. Soma de cada linha e de cada coluna

Para somar cada linha separadamente, zeramos a variável soma dentro do for externo (a cada nova linha):

```
for (int i = 0; i < matriz.length; i++) {
    int soma = 0;
```

```
    for (int j = 0; j < matriz[i].length; j++) {
        soma += matriz[i][j];
    }
    System.out.println("Soma da linha " + i + ": " + soma);
}
```

Para somar por coluna, invertemos a ordem: o for externo passa a percorrer as colunas (j) e o interno percorre as linhas (i):

```
for (int j = 0; j < matriz[0].length; j++) {
    int soma = 0;
    for (int i = 0; i < matriz.length; i++) {
        soma += matriz[i][j];
    }
    System.out.println("Soma da coluna " + j + ": " + soma);
}
```

20. Exemplo prático — notas dos alunos

Imagine uma turma com 4 alunos, cada um com 3 provas. Podemos representar as notas assim:

	P1	P2	P3
Aluno 0	8	7	9
Aluno 1	6	8	7
Aluno 2	9	9	10
Aluno 3	5	6	7

Cada linha representa um aluno; cada coluna representa uma prova. Podemos calcular a média de cada aluno somando sua linha e dividindo pelo número de provas, e depois decidir se ele foi aprovado (média ≥ 6) ou reprovado:

```
for (int i = 0; i < notas.length; i++) {
    double soma = 0;
    for (int j = 0; j < notas[i].length; j++) {
        soma += notas[i][j];
    }
    double media = soma / notas[i].length;
    if (media >= 6) {
        System.out.println("Aluno " + i + " - Aprovado");
    } else {
        System.out.println("Aluno " + i + " - Reprovado");
    }
}
```

21. Diagonal principal e diagonal secundária

Em uma matriz quadrada existe uma diagonal principal, formada pelos elementos em que linha = coluna (matriz[i][i]):

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

Na matriz 10,20,30 / 40,50,60 / 70,80,90, a diagonal principal é 10, 50, 90 (soma = 150). Já a diagonal secundária vai do canto superior direito ao canto inferior esquerdo (30, 50, 70) e pode ser obtida com:

```
int tamanho = matriz.length;
for (int i = 0; i < tamanho; i++) {
    System.out.println(matriz[i][tamanho - 1 - i]);
}
```

22. Matriz quadrada × matriz retangular

Uma matriz é chamada de quadrada quando possui a mesma quantidade de linhas e colunas (2×2, 3×3, 4×4...). Uma matriz não precisa ser quadrada: new int[2][5] cria uma matriz com 2 linhas e 5 colunas, e new int[3][5] cria uma com 3 linhas e 5 colunas — ambas válidas, mas apenas as quadradas possuem diagonal principal e secundária no sentido tradicional.

23. Exercícios de fixação

Responda no espaço abaixo de cada questão. O gabarito está na seção seguinte.

1) Escreva a declaração e a criação de uma matriz de inteiros com 4 linhas e 2 colunas, em uma única linha de código.

---

2) Dada a matriz int[][] m = {{5, 8}, {2, 9}, {7, 1}};, escreva o valor de m[2][0] e de m[1][1].

---

3) Escreva o trecho de código (for duplo) que soma todos os elementos de uma matriz chamada valores e imprime o resultado.

---

4) Explique, com suas palavras, por que precisamos de dois for aninhados para percorrer uma matriz, e não apenas um.

5) Para uma matriz quadrada notas de tamanho 5×5, escreva o for que percorre e imprime os elementos da diagonal principal.

⚠ **Erros comuns**

- Usar apenas um for para tentar percorrer a matriz inteira — é preciso um for para a linha e outro para a coluna.
- Trocar a ordem dos índices: matriz[coluna][linha] em vez de matriz[linha][coluna].
- Esquecer de zerar a variável de soma dentro do for externo ao somar cada linha separadamente — isso faz o valor acumular entre linhas.
- Confundir matriz.length (número de linhas) com matriz[i].length (número de colunas da linha i).

24. Desafio final — matriz mágica

Uma matriz mágica 3×3 é aquela em que a soma de cada linha, de cada coluna e das duas diagonais é sempre igual. Exemplo:

	Coluna 0	Coluna 1	Coluna 2
Linha 0	8	1	6
Linha 1	3	5	7
Linha 2	4	9	2

Nessa matriz, a soma de cada linha, coluna e diagonal é sempre 15. Crie um programa que: (1) receba uma matriz 3×3; (2) mostre a matriz; (3) calcule as somas das linhas; (4) calcule as somas das colunas; (5) calcule as duas diagonais; (6) informe se a matriz é ou não uma matriz mágica.

25. Gabarito

Questão 1

```
int[][] matriz = new int[4][2];
```

Questão 2

m[2][0] = 7 (linha 2, coluna 0). m[1][1] = 9 (linha 1, coluna 1).

Questão 3

```
int soma = 0;
for (int i = 0; i < valores.length; i++) {
    for (int j = 0; j < valores[i].length; j++) {
        soma += valores[i][j];
    }
}
System.out.println("Soma: " + soma);
```

Questão 4

Uma matriz tem duas dimensões (linhas e colunas), então é preciso um índice para cada uma. Um único for só consegue controlar uma variável de controle (um índice); para visitar todas as combinações de linha e coluna, precisamos de um for dentro do outro — o externo fixa a linha enquanto o interno percorre todas as colunas daquela linha.

Questão 5

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

26. Resumo para estudar antes da prova

- Vetor tem uma dimensão (int[] v); matriz tem duas dimensões (int[][] m).
- Criação: int[][] matriz = new int[linhas][colunas];
- Acesso e alteração: matriz[linha][coluna].
- Índices de linha e coluna sempre começam em zero.
- matriz.length = número de linhas; matriz[i].length = número de colunas da linha i.
- Para percorrer a matriz inteira: for duplo — o externo controla a linha (i), o interno controla a coluna (j).
- Somar por linha: zere a soma dentro do for externo. Somar por coluna: inverta a ordem dos for (externo em j, interno em i).
- Diagonal principal: matriz[i][i]. Diagonal secundária: matriz[i][tamanho - 1 - i].

- Matriz quadrada = mesma quantidade de linhas e colunas.

27. Checklist do aluno

Antes de considerar que você aprendeu matrizes, verifique se consegue:

- Criar uma matriz.
- Identificar linhas e colunas.
- Acessar e alterar um elemento específico.
- Preencher uma matriz usando Scanner.
- Percorrer uma matriz usando dois for (for duplo).
- Calcular a soma de todos os elementos.
- Contar números pares e ímpares.
- Encontrar o maior e o menor elemento.
- Calcular a soma de cada linha e de cada coluna.
- Calcular a diagonal principal e a diagonal secundária.