FIAP

Lógica de Programação - Java

Aula 09 – Vetores e matrizes

Alexandre Russi Jr.

profalexandre.junior@fiap.com.br

Variáveis compostas e motivação

FIAP

- Nossas variáveis só armazenam **1 informação** de texto, caractere, número inteiro ou real.
- Mas se quiséssemos **armazenar mais de uma informação**? Aí que entram as variáveis compostas.
- Considere um programa que receba o número n de alunos de uma turma e receba a nota de cada um deles para, posteriormente, realizar a média da turma. A partir do que sabemos até agora, o correto seria utilizar uma **estrutura de repetição** para somar essas notas e, assim, calcular a média.
- Entretanto, desta forma, **as notas não são armazenadas individualmente**, são apenas totalizadas em uma variável `soma`.
- Para isso, utilizaremos os vetores.

Vetores

FIAP

- Um vetor ou array é uma estrutura de dados **homogênea indexada**, sua função dentro de um programa é a de representar um conjunto de dados. Vejamos algumas características:
  - capacidade de armazenamento finita
  - armazena um único tipo de dado definido em sua declaração
  - acessa os elementos do conjunto através de um índice
- Exemplo de declaração de um vetor com valores inteiros com 10 posições:

```
// lado esquerdo: declaração do vetor de num inteiros
// lado direito: instanciamos o vetor com 10 posições
int[] vetInt = new int[10];

vetInt[0] = 10; // colocando o valor 10 na posição 0 do vetor
vetInt[1] = 1999; // colocando o valor 1999 na posição 1 do vetor

// atribuindo o valor da posição 0 do vetor na variável anoNasc
int anoNasc = vetInt[1];
```

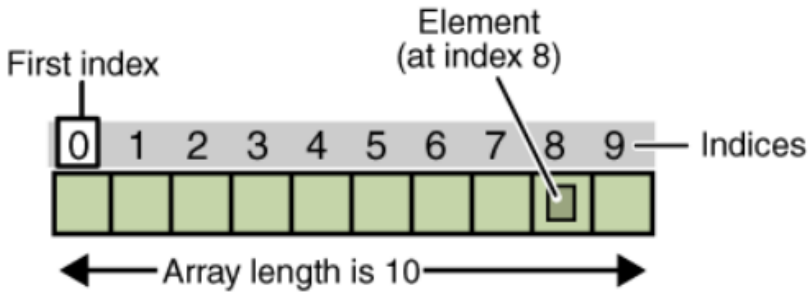
Vetores

```
int[] vetInt = new int[10];
vetInt[0] = 10;
vetInt[1] = 1999;
int anoNasc = vetInt[1];
```

- Os índices ou posições são números inteiros que começam do 0 e vai até 9
- **`vetInt[0]`** acessa a primeira posição e **`vetInt[9]`** acessa a última posição
- Podemos usar uma variável inteira para representar a posição/índice
- Representação do vetor **`vetInt`**:

0	1	2	3	4	5	6	7	8	9
10	1999	0	0	0	0	0	0	0	0

Vetores



- O atributo length armazena a capacidade do vetor, ou seja, o tamanho e quantas informações ele pode armazenar.
- Através do índice é possível acessar qualquer valor de uma determinada posição do vetor.
- Não é possível aumentar ou diminuir posições, se isso for necessário será preciso criar um outro vetor.

Vetores em Java

- Sintaxe: `<tipo>[] <nome> = new <tipo>[<tamanho>];`

- Onde:
  - <tipo> é o tipo de dado que o vetor armazena
  - <nome> é a referência/nome do objeto
  - <tamanho> a capacidade de armazenamento
- Também é possível inicializar um vetor diretamente:

```
String[] diasSemana = {"seg", "ter", "qua", "qui", "sex", "sab", "dom"};

int[] numeros = {10, 22, 13, 64, 35, -3, -87, 0, 48};
```

Vetores em Java – Dicas

- Desenhe um exemplo do vetor
- Na maioria das vezes use uma variável inteira para representar o índice (i, j, k)
- Não deixe essa variável ultrapassar a capacidade do vetor

  `(0 <= i <= vetor.length)`
- Faça o teste de mesa do seu algoritmo auxiliado pelo desenho do vetor
- Cada posição do vetor funciona como uma variável individual que foi declarada no seu programa

Exibindo valores do vetor

- Para percorrer todo o vetor, podemos utilizar um for

```
int[] numeros = {0, 5, 11, 4};

// FOR INDEXADO
for (int i = 0; i < numeros.length; i++) {
    System.out.println("Pos: " + i + " -- Valor: " + numeros[i]);
}

// FOR EACH --> percorre a lista e acessa os valores das posições
for (int num: numeros) {
    System.out.println("Valor: " + num);
}
```

Obs.: `numeros.length = 4`

Atividade 1: Duplas -- Repetições encadeadas

- Dado um conjunto de nomes de pessoas, escreva um algoritmo que imprima todas as possíveis duplas que podem ser formadas.
- Primeiro, crie um vetor e coloque alguns nomes nele.
- A seguir, exiba as possíveis duplas.

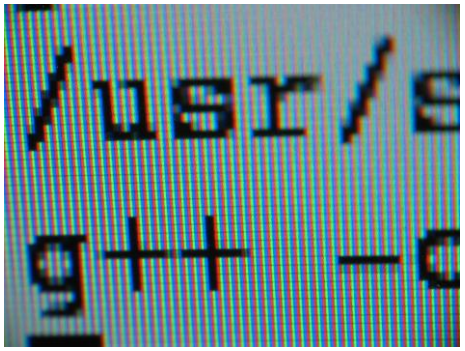
Atividade 1: Duplas – Resolução

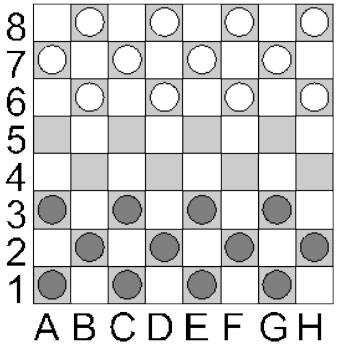
- Dado um conjunto de nomes de pessoas, escreva um algoritmo que imprima todas as possíveis duplas que podem ser formadas.

```
String nomes[] = {"Ana", "Bia", "Rodrigo", "Davi", "John"};

for (int j = 0; j < nomes.length - 1; j++) {
    String str = nomes[j];

    for (int i = j + 1; i < nomes.length; i++) {
        System.out.println(str + ", " + nomes[i]);
    }
}
```

Matrizes: motivação

- Planilha eletrônica (células)
- Tela de LCD (pixels)
- Tabuleiro de dama (casas)
- Tabela de banco de dados (dados do registro)



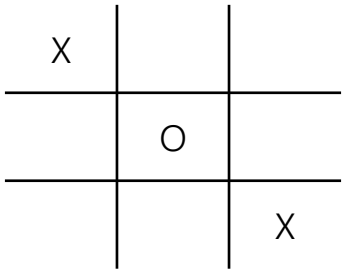


- Características em comum: possuem **linhas** e **colunas**.
- São representados por matriz na linguagem de programação.
- Matriz é utilizada para armazenar informações na forma de tabelas.

Matrizes: outras aplicações

- Visão computacional: representação de imagens, rotação e expansão.
- Criptografia: uma matriz pode ser a chave no processo de criptografia
- Banco de dados: tabela representa o conceito de matriz

- Jogo da velha:



Matrizes em Java

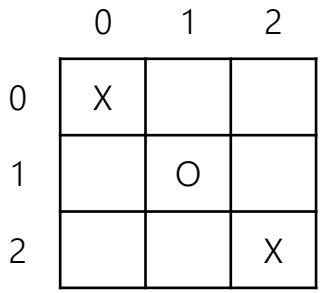
- Sintaxe: `<tipo>[][] <nome> = new <tipo>[<linha>][<coluna>];`

- Onde:
  - <tipo> é o tipo de dado que a matriz armazena
  - <nome> é a referência/nome do objeto
  - <linha> é o número de linhas da matriz
  - <coluna> é o número de colunas da matriz

- Exemplo no próximo slide.

Jogo da velha – Colocando dados

- Vamos imaginar que nossa tarefa é construir um jogo da velha!! O primeiro passo é a criação do tabuleiro (matriz onde serão armazenadas as jogadas).



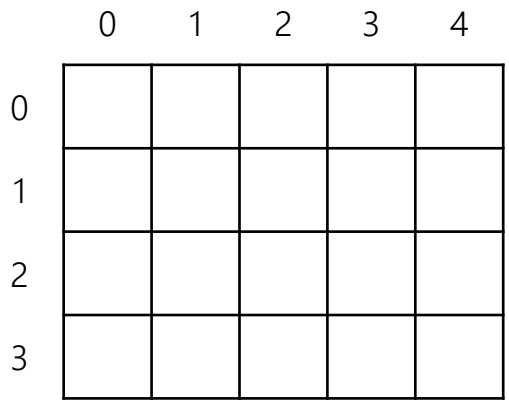
- Note que a matriz do nosso jogo deverá ter 3 linhas e 3 colunas e armazenar caracteres. Abaixo segue o código para sua criação na linguagem Java:

```
char[][] tabuleiro = new char[3][3];
tabuleiro[0][0] = 'X';
tabuleiro[1][1] = 'O';
tabuleiro[2][2] = 'X';
```

Matriz em Java – Exemplo

- O número 4 representa o número de linhas e o 5 o número de colunas, assim dizemos que nossa matriz possui uma dimensão **4x5**. Abaixo segue uma figura representando a matriz anterior:

```
int [][] matriz = new int[4][5];
```



Atividade 2

- Preencher uma matriz 4x5 de números inteiros com os números de 1 até 20.

- Matriz solução da atividade:

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20

Atividade 2 – Resolução

- Vamos tentar resolver por partes nosso problema:
  - pegando o número de linhas e colunas da matriz:

```
int [][] matriz = new int[4][5];

// num de linhas da matriz
int lin = matriz.length;
System.out.println(lin);

// num de colunas da matriz
int col = matriz[0].length;
System.out.println(col);
```

  - Para preencher a primeira linha da matriz:

```
int num = 1;

for (int j = 0; j < col; j++) {
      matriz[0][j] = num;
      num++;
}
```

Atividade 2 – Resolução

- Como a matriz possui 4 linhas, teríamos que repetir mais três vezes o bloco **for** apenas mudando o índice das linhas. Porém a matriz pode ter dimensões variáveis, logo a alternativa é colocar o bloco **for** dentro de um outro bloco **for**:

```
int num = 1;
for (int i = 0; i < lin; i++) {
      for (int j = 0; j < col; j++) {
            matriz[i][j] = num;
            num++;
      }
}
```

- Para exibir a matriz:

```
for (int i = 0; i < lin; i++) {
      for (int j = 0; j < col; j++) {
            System.out.print(matriz[i][j] + "\t");
      }
      System.out.println(""); // pular linha
}
```

Desafios e exercícios

Vetores e matrizes

Exercício 1

- Considerando o trecho de código Java representado a seguir:

```
int[] v = {2, 0, 3, 9};
v[v[2]] = v[v[1]];
for (int i: v){
     System.out.print(i);
}
```

- O que será impresso na tela?

a) 2 0 3 2

b) 2 0 3 9

c) 2 0 0 9

d) 2 3 3 2

Exercício 2

- Execute o trecho Java a seguir e marque a opção que contém o valor que será exibido na tela:

```
String s = "FIAPTOKIO";
char[] r = new char[9];

for (int i = s.length() - 1; i >= 0; i--) {
     r[i] = s.charAt(i);
}

for (char letra: r) {
     System.out.print(letra);
}
```

a) OIKOTPAIF

b) FIAPTOKIO

c) 1110162068068

d) 150,251,02

Exercício 3

FIAP

- Escreva um algoritmo que recebe um número inteiro **n** > 0, cria um vetor de números reais com n posições e preenche o vetor com **n** números aleatórios reais.
- **Depois** de preenchido o vetor, imprima na tela todos os números gerados.

Exercício 4

- Considere uma turma de **n** alunos onde desejamos calcular a média das notas da prova semestral e saber quantas notas estão iguais, acima e abaixo dessa média.
- Escreva um algoritmo que lê um inteiro **n** representando a quantidade de alunos e cada uma das **n** notas e mostra a média da turma, quantas notas são iguais, acima e abaixo da média da turma.

Exercício 5

- Faça um programa que tenha 2 vetores. Um vetor para os meses e outros para a quantidade de dias para cada mês.

- Seu programa deve exibir mensagens da seguinte forma:
  - O Mês de Jan tem 31 dias ao todo.
  - O mês de Fev tem 28 dias ao todo.
  - O mês de Mar tem 31 dias ao todo.
  - ...
  - O mês de Dez tem 31 dias ao todo.

Exercício 6

FIAP

- Escreva um algoritmo que lê um número inteiro **n**, cria um vetor de inteiros de tamanho **n**, faz a leitura de um conjunto de n números inteiros armazenando-os no vetor e depois calcula a somatória dos números contidos no vetor.
- Dica: note que a somatória deverá ser feita **após** o vetor estar preenchido.

Exercício 7

- Escreva um algoritmo que recebe uma lista de nomes e imprime os nomes na ordem inversa a da leitura.
- A lista termina quando o usuário aperta o Enter sem que nenhum nome tenha sido digitado.

Exercício 8

- Escreva um algoritmo que lê um número inteiro **n** > 0 e preenche um vetor de caracteres de **n** posições.
- Depois de preencher o vetor, você deverá inverter o seu conteúdo, ou seja, trocar o conteúdo da primeira posição (**0**) com a ´ultima (**n – 1**) a segunda com a penúltima e assim por diante até que o vetor esteja invertido.

Exercício 9 – A partir daqui, matrizes

- Crie um programa com uma matriz 3x4
  - 3 linhas
  - 4 colunas

- Atribua valores aleatórios à todas posições da matriz.
- Exiba essa matriz.

Exercício 10

- Faça um programa que realize a soma de duas matrizes, com **mesmas dimensões**. Seu programa deve ter 2 matrizes A e B de números inteiros. A terceira matriz deve ser a soma de A com B.

- Exemplo de soma:

$$\begin{bmatrix} -3 & 5 & 2 \\ 1 & 6 & 4 \end{bmatrix} + \begin{bmatrix} 7 & 2 & 0 \\ 9 & -2 & 3 \end{bmatrix} = \begin{bmatrix} 4 & 7 & 2 \\ 10 & 4 & 7 \end{bmatrix}$$

Alguma Dúvida?

Entre em contato por e-mail ou via LinkedIn

/alexandrerussi