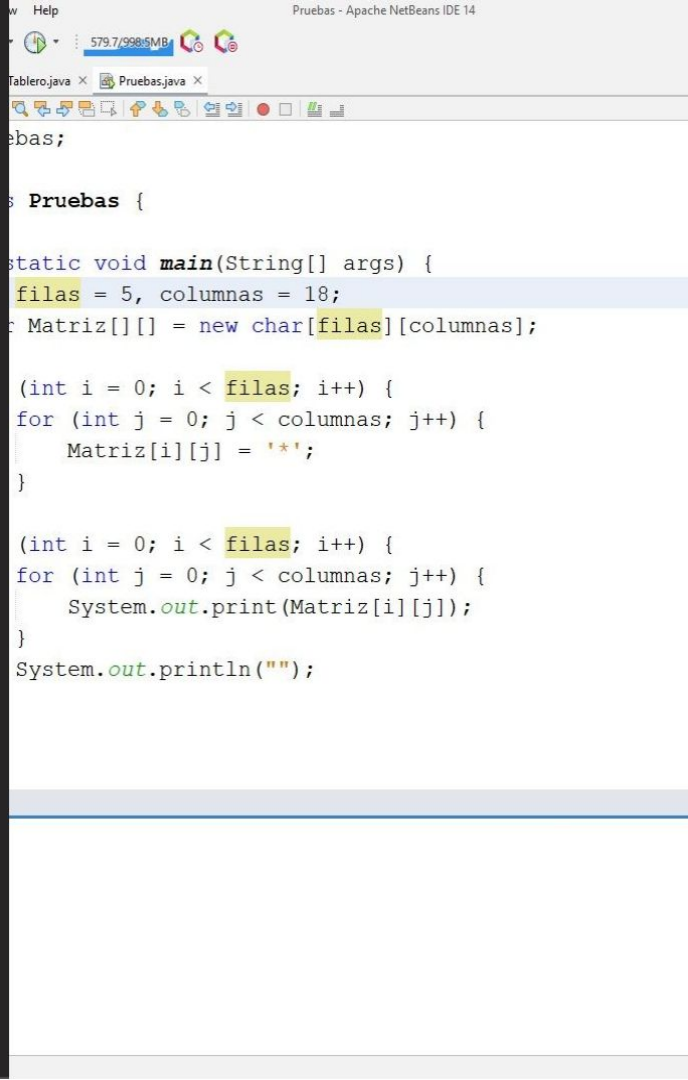
MATRIZ E VETORES EM JAVA

PAULO TELLI – VINICIUS KOGLINSKI – JOÃO BATISTA

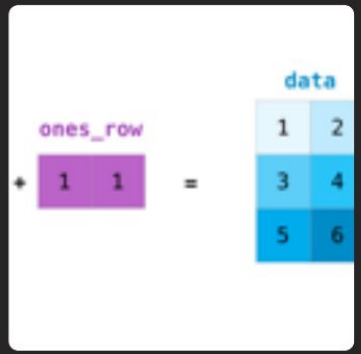
Matriz em Java

Nesta apresentação, vamos explorar o poderoso recurso das matrizes em Java - seus conceitos fundamentais, criação, inicialização, acessos e aplicações.



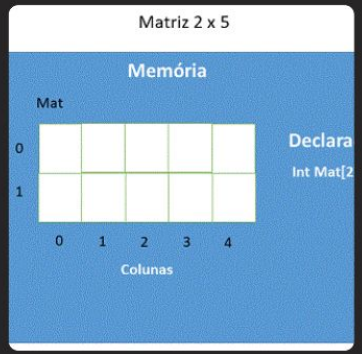
Introdução à Matriz em Java

Uma matriz é uma estrutura de dados retangular composta de elementos dispostos em linhas e colunas. É um recurso muito importante em programação para armazenar e manipular grandes quantidades de dados de forma organizada e eficiente.



Criação de Matrizes

Criar uma matriz é fácil, seja declarando-a ou instanciando-a em tempo de execução. Existem muitas maneiras de criar uma matriz em Java.



Inicialização de valores em Matrizes

Os valores de uma matriz podem ser inicializados em branco ou com valores predefinidos. Vamos discutir algumas maneiras de fazer isso.

Acesso aos elementos de uma Matriz

Acesso aos elementos de uma matriz é uma operação essencial para modificar ou ler esses elementos. Vamos ver como acessar os elementos no Java.

Acesso Direto

O acesso a elementos em matrizes é feito através das posições da linha e da coluna, e pode ser realizado diretamente por meio de índices.

Acesso com Percorrida Simples

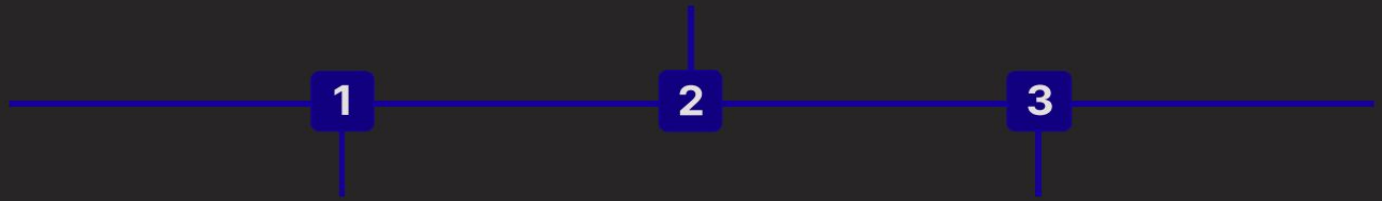
Percorrer matrizes com loops pode ser muito útil em muitos aplicativos como pesquisar, filtrar e visualizar dados.

Operações Básicas em Matrizes

Muitas operações podem ser executadas com matrizes. Vamos ver as principais operações que podemos fazer com matrizes em Java.

Multiplicação de Matrizes

A multiplicação de matrizes é uma operação crítica e é muito importante em muitas aplicações de computação científica.



Adição de Matrizes

Para adicionar matrizes, simplesmente adicionamos os elementos correspondentes de cada matriz.

Transposição de Matrizes

A transposição é uma operação que troca linhas por colunas em uma matriz.

Aplicações Práticas de Matrizes em Java

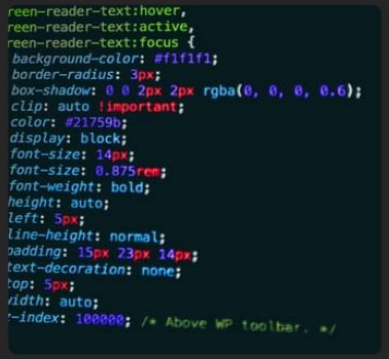
As matrizes são amplamente utilizadas nas áreas de engenharia, ciência de dados e redes de computadores. Vamos ver algumas de suas aplicações em Java.

Análise de Dados

Gráficos de tendência e de barras são criados a partir de matrizes que contêm informações sobre dados do mundo real.

Redes de Computadores

As matrizes são usadas em algoritmos de roteamento para encaminhar pacotes entre dispositivos em uma rede.

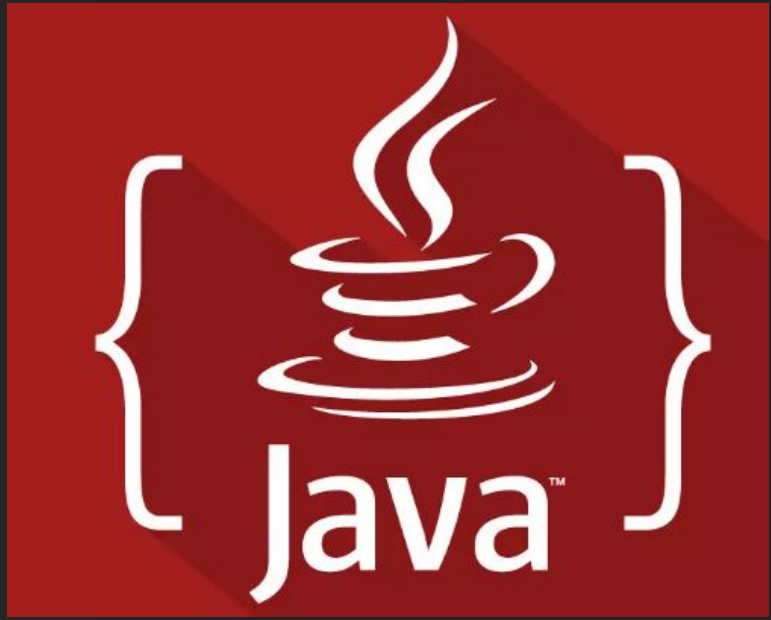


Desenvolvimento Web

As matrizes são usadas para armazenar elementos em uma página web, como linhas de uma tabela ou informações de um formulário.

Introdução ao Vetor em Java

O vetor é uma estrutura que permite armazenar vários valores em uma única variável. É uma ferramenta poderosa e flexível para processar dados em Java.



Declarando um Vetor em Java

Criando um Vetor

Podemos criar um vetor em Java usando a sintaxe: `tipo[] nomeDoVetor = new tipo[tamanhoDoVetor];`

Inicializando um Vetor

Também podemos inicializar um vetor durante a criação usando uma lista de valores: `tipo[] nomeDoVetor = {valor1, valor2, ..., valorN};`

Validação

Devemos garantir que o tamanho do vetor seja apropriado para o número de elementos que queremos armazenar nele. O Java lançará uma exceção ArrayIndexOutOfBoundsException se tentarmos acessar um elemento fora do intervalo válido.

Manipulando Elementos do Vetor

Adicionar Elementos

Podemos adicionar elementos a um vetor usando seu índice. O índice começa em 0.

Alterar Elementos

Podemos alterar um elemento de um vetor usando seu índice e atribuindo um novo valor a ele.

Remover Elementos

Os elementos de um vetor são imutáveis, mas podemos simular a remoção definindo o elemento como `null`.

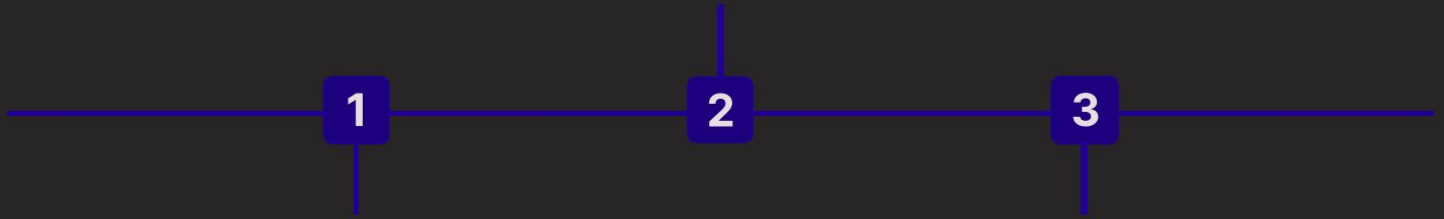
Ordenar Elementos

Podemos ordenar os elementos de um vetor usando a classe `Arrays`.

Acessando Elementos do Vetor

Acessar todos os Elementos

Podemos acessar todos os elementos de um vetor usando um loop for.



Acessar um Elemento

Podemos acessar um elemento de um vetor usando seu índice. O índice começa em 0.

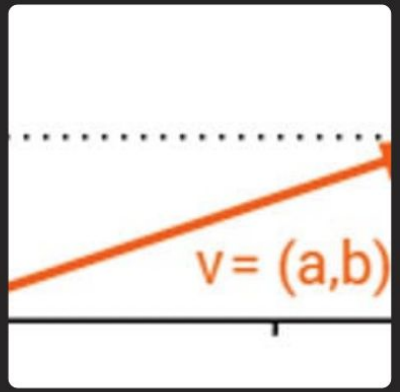
Acessar o Último Elemento

Podemos acessar o último elemento de um vetor subtraindo 1 do seu comprimento.

Comprimento do Vetor

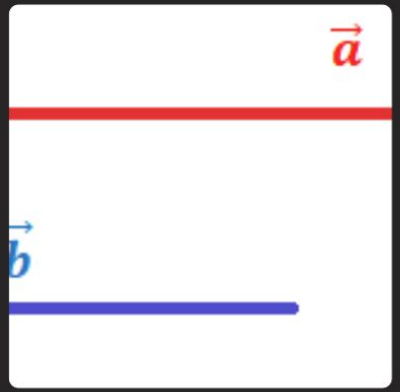
Comprimento de um Vetor

O comprimento de um vetor é a quantidade de elementos que ele contém. Podemos acessar o comprimento de um vetor usando a propriedade `length`.



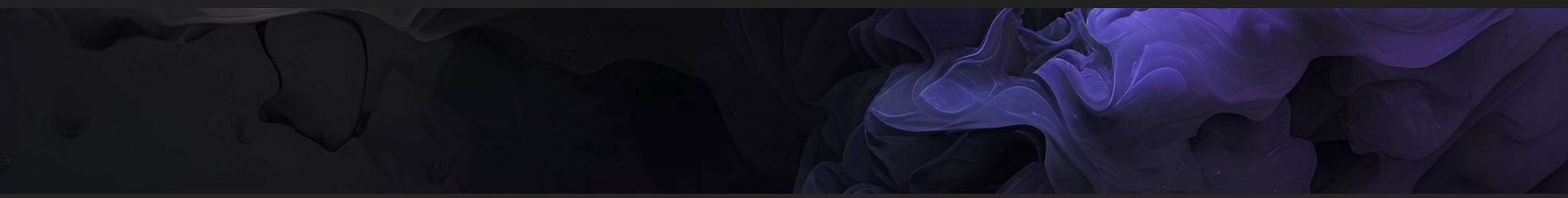
Usando o Comprimento

O comprimento de um vetor é frequentemente usado para criar loops `for` que percorrem todos os seus elementos.



Não Alterar o Comprimento

O comprimento de um vetor é imutável. Devemos criar um novo vetor se quisermos adicionar ou remover elementos.



ArrayIndexOutOfBoundsException

1 Causa

O erro ArrayIndexOutOfBoundsException acontece quando tentamos acessar um elemento de um vetor usando um índice que não existe.

2 Resolução

Podemos evitar este erro garantindo que o índice esteja dentro do intervalo válido do vetor.

3 Dicas

Devemos verificar sempre se o índice do vetor está em um intervalo válido antes de usá-lo. Também devemos evitar usar números mágicos para índices.

Exemplos Práticos de Uso de Vetores em Java

Armazenando Dados de Uma Pesquisa

Podemos usar um vetor para armazenar os resultados de uma pesquisa de opinião. Cada elemento do vetor seria uma resposta e o índice seria a opção escolhida pelos participantes da pesquisa.

Exemplo: `int[] resultados = {12, 27, 8, 53};`

Armazenando Notas de Um Aluno

Podemos usar um vetor para armazenar as notas de um aluno em diferentes disciplinas. Cada elemento do vetor seria uma disciplina e o valor seria a nota do aluno.

Exemplo: `double[] notas = {7.5, 8.0, 6.5, 9.0};`

Armazenando Números em Ordem Crescente

Podemos usar um vetor para armazenar uma lista de números em ordem crescente. Isso facilita a busca e a ordenação posterior dos números.

Exemplo: `int[] numeros = {1, 3, 5, 7, 9};`

Armazenando Dados em Uma Tabela

Podemos usar um vetor bidimensional para armazenar os dados de uma tabela. Os elementos do vetor seriam as células da tabela.

Exemplo: `String[][] tabela = {{"João", "20"}, {"Maria", "22"}, {"Pedro", "19"}};`

OBRIGADO A TODOS PELA ATENÇÃO