Linguagem Java

Explore os fundamentos da linguagem de programação Java, abrangendo conceitos essenciais como variáveis, tipos de dados, entrada e saída de dados, e operadores.

by Alexandre de Souza Jr.

Variáveis e Tipos de Dados

1 Declaração de Variáveis

A declaração de variáveis é o primeiro passo para armazenar e manipular dados em um programa Java. Cada variável possui um nome único e um tipo de dado associado.

2 Tipos Primitivos

Java possui tipos de dados primitivos básicos, como inteiros, decimais, caracteres e booleanos, que fornecem a base para a construção de aplicativos mais complexos.

3 Tipos de Referência

Além dos tipos primitivos, Java também possui tipos de referência, como arrays, strings e classes personalizadas, que permitem a representação de estruturas de dados mais sofisticadas.

Declaração e Inicialização de Variáveis

1

Declaração

Criação da variável com um nome único.

2

Inicialização

Atribuição de um valor inicial à variável.

3

Regras

Respeitar convenções de nomenclatura e boas práticas (padrão *camelcase*).

Em Java, as variáveis devem ser declaradas antes de serem utilizadas. Essa declaração inclui a definição do tipo de dado que a variável irá armazenar. Após a declaração, é possível atribuir um valor inicial a essa variável, um processo chamado de inicialização. É importante seguir as regras de nomenclatura e boas práticas para garantir a legibilidade e manutenibilidade do código.

Tipos Primitivos e Tipos de Referência

Tipos Primitivos

Os tipos primitivos são os blocos de construção básicos da linguagem Java, como **int**, **double**, **boolean** e **char**. Eles são valores simples, ocupam pouco espaço na memória e não possuem métodos ou atributos.

Tipos de Referência

Já os tipos de referência, como **String**, **Array** e classes criadas pelo desenvolvedor, são objetos mais complexos que possuem métodos e atributos. Eles são alocados dinamicamente na memória e permitem maior flexibilidade e funcionalidade.

Alocação de Memória

Os tipos primitivos são armazenados na pilha de execução, enquanto os tipos de referência são alocados na memória *heap*. Isso afeta a forma como eles são manipulados e gerenciados pelo sistema.

Entrada e Saída de Dados

1

Leitura de Dados do Usuário

Em Java, podemos usar a classe Scanner para ler dados digitados pelo usuário, como números, textos e datas.

2

Exibição de Dados na Tela

O comando System.out.println() permite exibir mensagens, valores de variáveis e resultados de expressões na tela do console.

3

Formação de Saída

Podemos formatar a saída de dados usando recursos como concatenação de strings, formatação de números e alinhamento de colunas.

Leitura de Dados do Usuário

Entrada de Dados

Para ler dados do usuário em Java, você pode utilizar a classe **Scanner**. Essa classe permite que você capture diferentes tipos de dados, como números inteiros, decimais, strings e muito mais.

Métodos da Classe Scanner

- nextInt(): Lê um valor inteiro.
- nextDouble(): Lê um valor decimal.
- next(): Lê uma string até o próximo espaço em branco.
- nextLine(): Lê uma linha inteira de texto, incluindo espaços.

Exibição de Dados na Tela

Janela de Saída

A saída de dados em Java é exibida em uma janela de console ou terminal, onde os valores das variáveis e expressões são mostrados de forma clara e organizada.

Métodos de Impressão

Os principais métodos de exibição de dados são o System.out.print() e System.out.println(), que permitem imprimir valores, textos e quebras de linha na tela.

Formatação de Saída

Pode-se utilizar recursos de formatação, como alinhamento, casas decimais e espaçamento, para apresentar os dados de forma mais organizada e legível.

Operadores Aritméticos, Relacionais e Lógicos

Operadores Aritméticos

Utilizam-se para realizar cálculos matemáticos básicos, como adição (+), subtração (-), multiplicação (*), divisão (/) e módulo (%). Permitem fazer operações entre variáveis e valores numéricos.

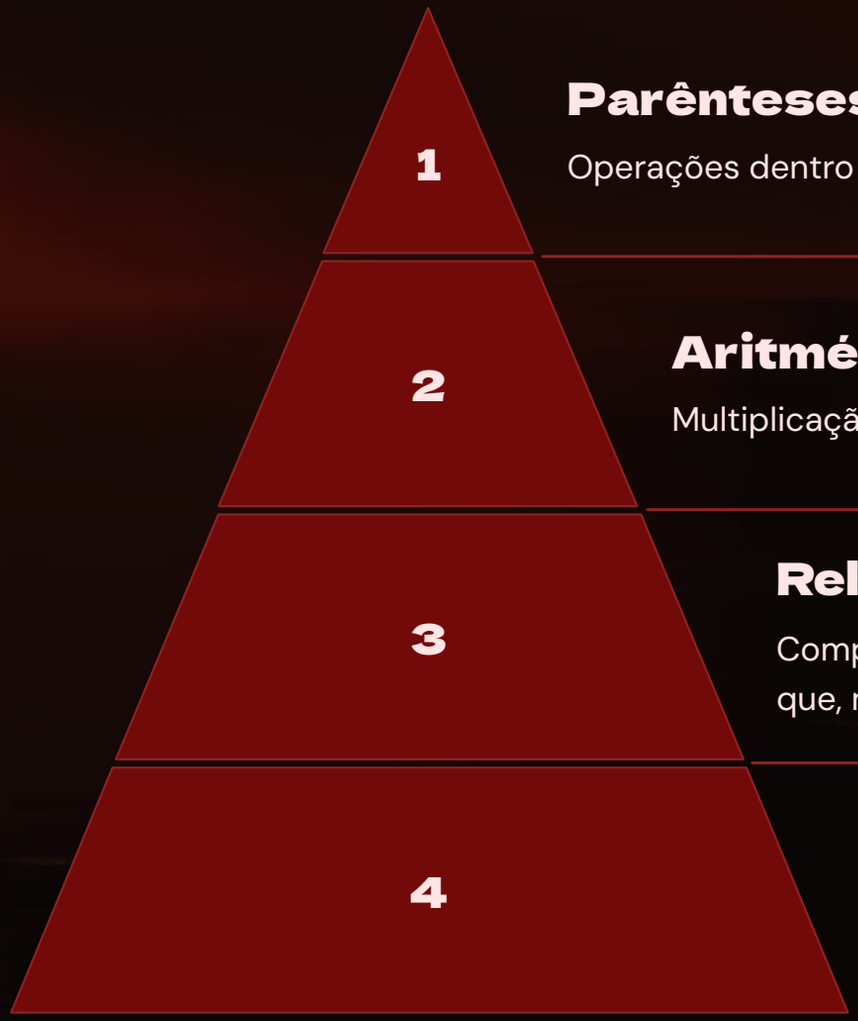
Operadores Relacionais

Comparam dois valores e retornam um resultado booleano (verdadeiro ou falso). Exemplos: igual (==), diferente (!=), maior que (>), menor que (<), maior ou igual (>=), menor ou igual (<=).

Operadores Lógicos

Combinam expressões booleanas para criar condições mais complexas. Exemplos: E (&&), OU (||), NÃO (!). Úteis em estruturas de controle, como if-else e loops.

Precedência de Operadores



Parênteses

Operações dentro de parênteses são executadas primeiro.

Aritmética

Multiplicação, divisão, adição e subtração têm precedência.

Relacionais

Comparações como maior que, maior ou igual a, menor que, menor ou igual a, e igualdade vêm depois.

Lógicos

Os operadors lógicos "e", "ou" e "não" são os últimos a serem executados.

É importante entender a ordem de precedência dos operadores em Java para escrever expressões aritméticas e lógicas corretamente. Primeiro são executadas as operações dentro de parênteses, seguido pela aritmética, comparações e, por fim, os operadores lógicos.

Conclusão e Próximos Passos

Chegamos ao fim desta jornada introdutória sobre a linguagem Java. Aprendemos sobre variáveis, tipos de dados, entrada, saída e os diferentes operadores. Agora, é hora de colocar esses conceitos em prática e avançar para tópicos mais avançados, como estruturas de decisão, *loops* e tratamento de exceções.