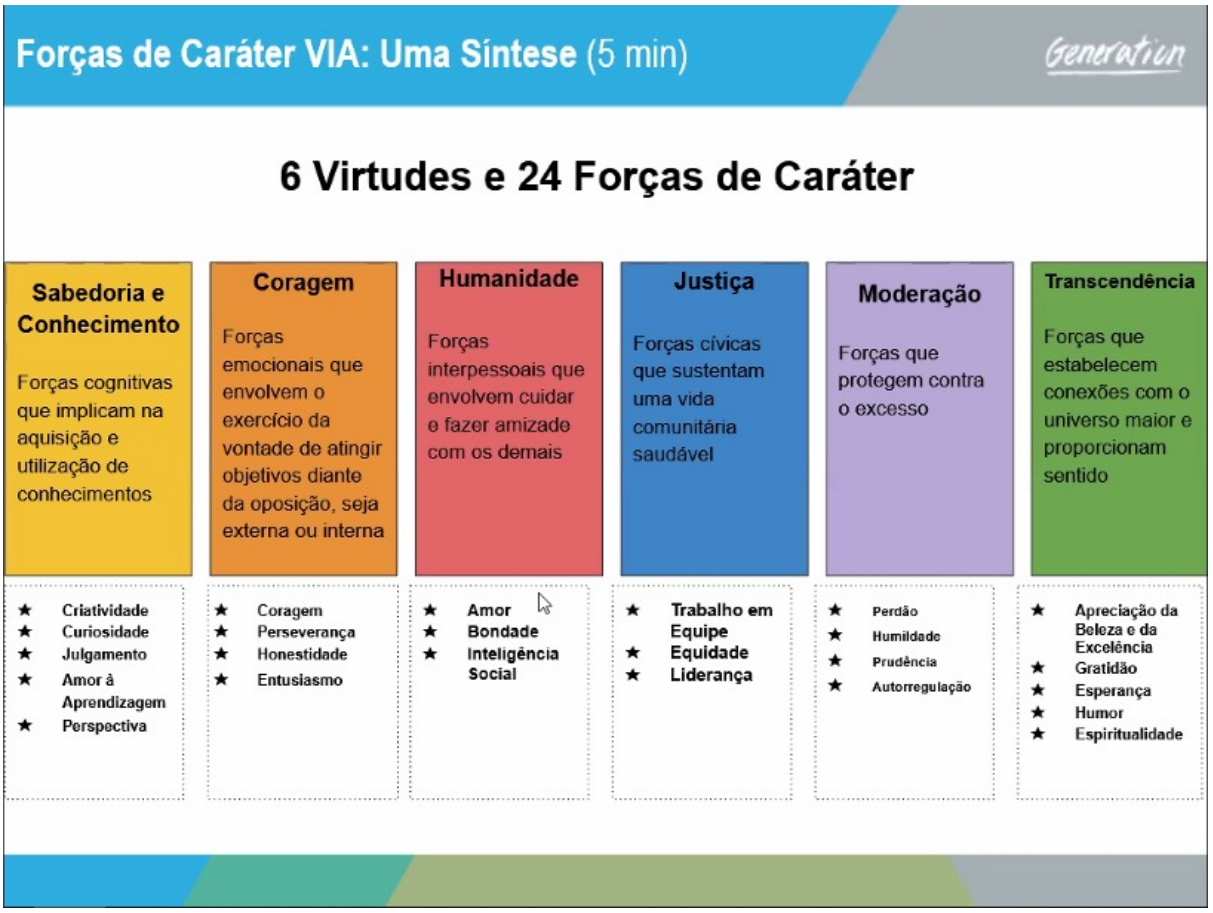
Índice

Assuntos: Fazendo mais com suas forças, JAVA variáveis, JAVA operadores lógicos.

Abertura Alunos:
Reflexão diária: Preconceito contra as mães no mercado de trabalho
Ellen Cristina e Luciana Rocha.

- É possível pedir quebra de contrato em caso de assédio.
- Ao processar, pedir segredo de justiça para RH não ter acesso aos detalhes do processo.

Forças de Caráter VIA



Java - Variáveis

VARIÁVEL: Uma variável é um tipo de armazenamento de dados em memória, que possui o conteúdo variável durante a execução de um algoritmo ou programa. Uma variável pode assumir vários valores diferentes ao longo da execução do programa, mas, em um determinado momento, possui apenas um valor.

CONSTANTE: Uma constante é um tipo de armazenamento de dados em memória, que possui um valor fixo e imutável, durante a execução de um algoritmo ou programa.

Toda a Variável deve possuir um tipo de dado. No caso da Linguagem Java, por se tratar de uma Linguagem de Programação, existem diversos tipos de dados, onde cada um tem uma indicação de uso de acordo com as suas características.

Toda a variável possui um **Identificador**, que representa o nome escolhido para rotular a variável.

Tipos de dados

(Para informações mais específicas, vá ao **cookbook md3**, na aba de links.)

Os tipos de dados estão divididos em duas categorias:

1. **Tipos de dados primitivos** (também chamados de tipos intrínsecos ou integrados), que correspondem a dados mais simples, como números inteiros e reais, caracteres simples e lógicos

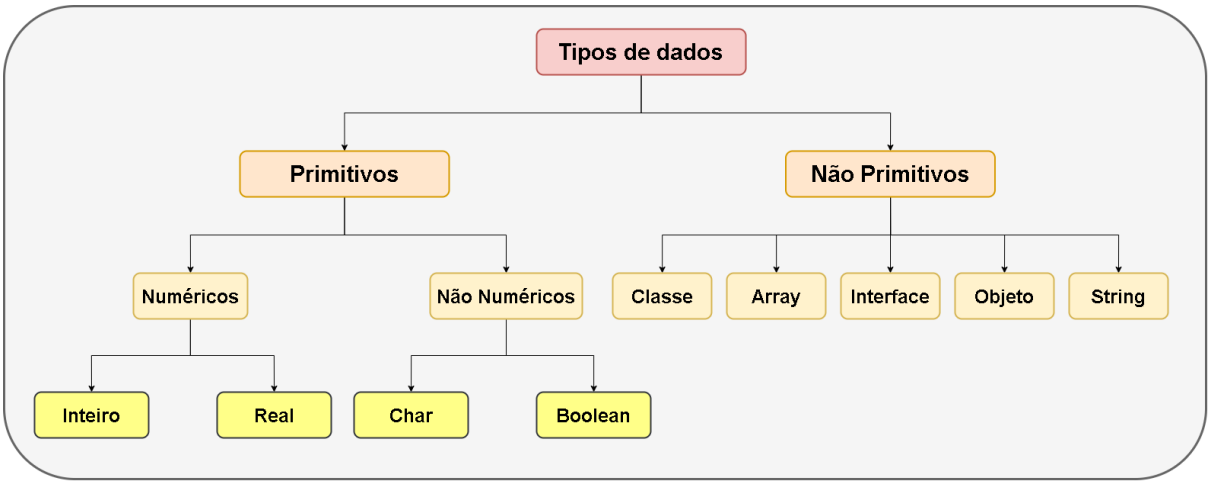
2. **Tipos de dados não primitivos** (também chamados de tipos de dados derivados ou de referência), que consistem em arrays (vetores), classes, objetos, interfaces e Strings, armazenam valores de tipos variados ou um grupo de valores definidos em uma Classe.

São os 8 primitivos:

1. **byte**
2. **short**
3. **int**
4. **long**
5. **char**
6. **float**
7. **double**
8. **boolean (true ou false)**

E 5 não primitivos:

1. **classe**
2. **array**
3. **interface**
4. **objeto**
5. **string**



Dados inteiros

Os tipos de dados inteiros em Java representam valores numéricos inteiros ou números sem partes fracionárias ou pontos decimais, (ponto flutuante). Por exemplo, 225, -56524, 0, 1045, entre outros, estão nesta categoria.

Os tipos de dados inteiros são subdivididos em quatro tipos: **byte, short, int e long.**

Dados reais

Os tipos de dados Reais ou de ponto flutuante são úteis para armazenar números contendo casas decimais ou partes fracionárias. Por exemplo: 3,14, -2,567, 0,00034, entre outros. Eles são chamados de números de ponto flutuante.

Existem dois tipos de tipos de dados de ponto flutuante: **float e double.**

Não numéricos

Os tipos de dados não numéricos são classificados em duas categorias: **char e boolean**.

Identificadores

- **CamelCase:** O uso de separação de palavras por letras maiúsculas ao invés de espaços. Exemplo: **formadoEmProgramaçãoJava**.

Restrições

- Deve-se sempre começar com uma letra, sublinhado (**_**), ou cifrão (**$**), jamais com dígito.
- Não pode ser **true**, **false** ou **null**.
- Pode ter qualquer comprimento, mas indica-se que não sejam muito extensos.
- Deve ser exclusivo e auto descritivo. Exemplo: **valorFinal** .
- Não se pode usar palavras reservadas. Exemplo: **class**, **for**, **while**, **public**. (Lista de palavras reservadas no cookbook)

Convenções

- Todas as variáveis e métodos de instância pública começam com uma palavra com todas as letras minúsculas. Exemplo: **media**, **soma**.
- Se for composta, a próxima palavra deve-se iniciar em maiúscula, com o camel-case. Exemplo: **mediaFina**l, **somaCompleta**.
- Classes e interfaces devem começar com maiúscula inicial. Exemplo: **Produto**, **HelloJava**.

Observações importantes:

- Ao atribuir um valor para uma variável do tipo long, **no final do valor deve ser acrescentada a letra l minúscula**, para indicar que se trata de um dado do tipo int;

- Ao atribuir um valor para uma variável do tipo float, as casas decimais devem ser informadas, mesmo que seja zero e **no final deve ser acrescentada a letra f minúscula**, para diferenciar de um dado do tipo double;

- Diferente da língua portuguesa que utiliza a vírgula como separador de casas decimais, o Java utiliza o ponto (padrão americano);

- Ao atribuir um valor para uma variável do tipo char, o valor deve estar entre aspas simples (' ');

Constante

No Java a declaração de uma constante é bem parecida com a declaração de uma variável. Basta colocar a palavra reservada **final** antes do tipo da variável:

```
final byte BIT = 127;
final short VALOR = 254;
```

Quando declaramos uma constante, o nome da constante deve ser escrito em letras maiúsculas e o valor da constante deve ser obrigatoriamente atribuído.

String

O tipo **String**, com **S** maiúsculo, é um tipo de dado não primitivo e um objetos Java mais utilizados, o tipo String é semelhante ao tipo **cadeia** no Portugol.

As Strings em Java não são tratadas como uma mera sequência de caracteres. Strings são Objetos ou Instâncias da classe **java.lang.String**, portanto, devem ser declarados e instanciados.

Observe que:

- Ao atribuir um valor para uma String, o valor deve estar entre **aspas duplas**;
- Quando adicionamos um valor numérico em uma String, o Java interpretará o valor como texto e não como um número.

Java - Operadores

(Para informações mais específicas, vá ao **cookbook md5**, na aba de links)

Em relação aos tipos de dados, os Operadores são classificados como:

- Operadores Aritméticos
- Operadores de Atribuição
- Operadores Unários
- Operadores relacionais
- Operadores Lógicos

Os operadores de atribuição, aritméticos, relacionais e lógicos no Java são utilizados principalmente na etapa de processamento - para a construção da lógica.

Operadores Aritméticos

Os operadores aritméticos realizam as operações fundamentais da matemática entre duas variáveis e retornam o resultado.

Caso seja necessário escrever operações maiores ou mais complexas, podemos combinar esses operadores e criar expressões, o que nos permite executar todo tipo de cálculo de forma pragmática.

Ao fazer o resto da divisão, aparecem os números que restam após a vírgula. para saber se é par ou ímpar, se divide por 2. Ele dará 1 ou 0.

Operador	Operação	Exemplo
+	Soma	

-	Subtração	
*****	Multiplicação	
/	Divisão	
%	Módulo ou Resto	x%2 Se a operação tiver resto o resultado será 1. Se não, será 0.

Operadores de Atribuição

O operador de atribuição é utilizado para definir o valor inicial ou sobrescrever o valor de uma variável.

Em seu uso, o operando à esquerda representa a variável para qual desejamos atribuir o valor informado à direita.

Operador	Operação	Exemplo	Equivalente
=	Atribuição simples		
+=	Atribuição com soma	x += y	x = x + y
-=	Atribuição com subtração	x -= y	x = x - y
***=**	Atribuição com multiplicação	x * = y	x = x * y
/=	Atribuição com divisão	x /= y	x = x / y
%=	Atribuição com módulo	x %= y	x = x % y

Operadores Unários

Se o sinal de soma está antes do valor, é incremento, se está após é decremento.

Operador	Operação
- **Número**	Número negativo

++ x	Pré Incremento
- - x	Pré Decremento
x ++	Pós Incremento
x - -	Pós Decremento

Incremento é o ato de somar ou subtrair 1 do valor.
Exemplo: x = (x + 1) >> x = ++x
Se for constante será de um em um.
Muito útil para loops crescentes ou decrescentes.

Pré (x = ++y)

- Primeiro o y é incrementado em 1
- x vai receber o valor de y já incrementado
- Só depois o x vai receber esse valor já incrementado

Pós (x = y++)

- Primeiro o x recebe o valor de y não incrementado
- x vai receber o valor de y ainda NÃO incrementado
- y é incrementado em 1

Pré Incremento (x = ++y)
1º) Y é incrementado em 1
2º) X recebe o valor de Y já incrementado
- ++y : x vai receber o valor de Y já incrementado

Pós Incremento (x = y++)
1º) X recebe o valor de Y ainda não incrementado
2º) Y é incrementado em 1
- y++ : x vai receber o valor de Y ainda NÃO incrementado

--- Outra forma de explicar (x = ++y // x = y++):
- Em ambos os casos, o Y será incrementado em 1.
- A diferença é qual valor o X vai receber (se será o valor Y antes ou depois do incremento).
  - ++y: Se o ++ vem antes do Y, X vai receber valor de Y incrementado
  - y++: Se o ++ vem depois do Y, X vai receber valor de Y NÃO incrementado

---

Operadores Relacionais

Os operadores relacionais, assim como os de igualdade, avaliam dois operandos.

Neste caso, mais precisamente, define se o operando à esquerda é maior, maior ou igual, menor, menor ou igual, igual, ou diferente ao da direita, retornando um valor booleano.

Operador	Descrição
>	Maior do que
>=	Maior do que ou igual
<	Menor do que
<=	Menor do que ou igual
==	Igual
!=	Diferente

Booleano: Um valor que só se classifica como true ou false (verdadeiro ou falso).

Operadores Lógicos

Os operadores lógicos representam o recurso que nos permite criar expressões lógicas maiores a partir da junção de duas ou mais expressões.

Operador	Operação
&&	“E” Se ambas as condições forem verdadeiras, faça
||	“Ou” Se uma ou outra condição forem verdadeiros
!	“Não” Reverter o resultado de um boolean. Se for verdadeiro, se torna falso, e vice-versa.

Links e afins

- **Cookbook MD3, variáveis e constantes, tipos de dados:**
  https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/03.md

- **Cookbook, MD5, operadores:**
  https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/05.md

---

Dúvidas
