02

Operador ternário, de referência e de concatenação

Operador ternário - Condicional

Há também um operador para controle de fluxo do programa, como um `if`. É chamado de **operador ternário**. Se determinada condição acontecer, ele vai por um caminho, caso contrário vai por outro.

A estrutura do operador ternário é a seguinte:

variável = teste_booleano **?** valor_se_verdadeiro **:** valor_se_falso;

```
int i = 5;
System.out.println(i == 5 ? "verdadeiro": "falso");//
verdadeiro
System.out.println(i != 5 ? 1: 2);                    // 2

String mensagem = i % 2 == 0 ? "é par" : "é ímpar";
```

COPIAR CÓDIGO

O operador condicional sempre tem que retornar valores que podemos usar para atribuir, imprimir etc.

Operador de referência

Para acessar os atributos ou métodos de um objeto precisamos aplicar o operador `.` (ponto) em uma referência. Você pode imaginar que esse operador navega n…

referência até chegar no objeto.

```
String s = new String("Caelum");

// Utilizando o operador "." para acessar um
// objeto String e invocar um método.
int length = s.length();
```

COPIAR CÓDIGO

Concatenação de Strings

Quando usamos Strings, podemos usar o `+` para denotar concatenação. É a única classe que aceita algum operador fora o ponto.

Em Java, não há sobrecarga de operadores como em outras linguagens. Portanto, não podemos escrever nossas próprias classes com operadores diversos.

StringBuilder

A concatenação de Strings é um *syntax sugar* que o próprio compilador resolve. No código compilado, na verdade, é usado um `StringBuilder`.

Precedência

Não é necessário decorar a precedência de todos operadores do Java, basta saber o básico, que primeiro são executados pré-incrementos/decrementos, depois multiplicação/divisão/mod, passando para soma/subtração, depois os *shifts* (`<<` , `>>` , `>>>`) e, por último, os pós-incrementos/decrementos.

As questões da certificação não entram em mais detalhes que isto.

Pontos importantes

- Na atribuição de um valor para uma variável primitiva, o valor deve ser do mesmo tipo da variável ou de um menos abrangente.

EXCEÇÃO À REGRA: Para os tipos `byte`, `short` e `char`, em atribuições com literais do tipo `int`, o compilador verifica se o valor a ser atribuído está no range do tipo da variável. *Toda variável não primitiva está preparada somente para armazenar referências para objetos que sejam do mesmo tipo dela.* Toda comparação e toda operação lógica devolve `boolean`. *O resultado de toda operação aritmética é no mínimo `int` ou do tipo da variável mais abrangente que participou da operação.* A comparação de valores numéricos não considera os tipos dos valores. *As referências e os valores `boolean` só podem ser comparados com `==` ou `!=`.* Toda atribuição é por cópia de valor.

Observação: O recurso do *autoboxing* permite fazer algumas operações diferentes envolvendo variáveis não primitivas. Discutiremos sobre autoboxing adiante.