02

Operadores aritméticos

Operadores aritméticos

Os cálculos sobre os valores das variáveis primitivas numéricas são feitos com os operadores aritméticos. A linguagem Java define operadores para as principais operações aritméticas (soma, subtração, multiplicação e divisão).

```
int dois = 2;
int dez = 10;

// Fazendo uma soma com o operador "+".
int doze = dois + dez;

// Fazendo uma subtração com o operador "-".
int oito = dez - dois;

// Fazendo uma multiplicação com o operador "*".
int vinte = dois * dez;

// Fazendo uma divisão com o operador "/".
int cinco = dez / dois;
```

COPIAR CÓDIGO

Além desses, há um operador para a operação aritmética "resto da divisão".

```
int dois = 2;
int dez = 10;
```

```
// Calculando o resto da divisão de 10 por 2.
int zero = dez % dois;
```

COPIAR CÓDIGO

O resultado de uma operação aritmética é um valor. A dúvida que surge é qual será o tipo dele. Para descobrir o tipo do valor resultante de uma operação aritmética, devem-se considerar os tipos das variáveis envolvidas.

A regra é a seguinte: o resultado é do tipo mais abrangente entre os valores das variáveis envolvidas ou, no mínimo, o `int` .

```
int idade = 15;
long anos = 5;

// ok, o maior tipo era long
long daquiCincoAnos = idade + anos;

// não compila, o maior tipo era long, devolve long
int daquiCincoAnos2 = idade + anos;
```

COPIAR CÓDIGO

Mas devemos lembrar da exceção: o mínimo é um `int` :

```
byte b = 1;
short s = 2;

// devolve no mínimo int, compila
int i = b + s;
```

```
// não compila, ele devolve no mínimo int
byte b2 = i + s;

// compila forçando o casting, correndo risco de perder
// informação
byte b2 = (byte) (i + s);
```

COPIAR CÓDIGO

Divisão por zero

Dividir (ou usar `mod`) um inteiro por zero lança uma `ArithmeticException`. Se o operando for um `float` ou `double`, isso gera infinito positivo ou negativo (depende do sinal do operador). As classes `Float` e `Double` possuem constantes para esses valores.

```
int i = 200;
int v = 0;

// compila, mas exception
System.out.println(i / v);

// compila e roda, infinito positivo
System.out.println(i / 0.0);
```

COPIAR CÓDIGO

Ainda existe o valor `NaN` (*Not a Number*), gerado pela radiciação de um número negativo e por algumas contas com números infinitos.

```
double positivoInfinito = 100 / 0.0;
double negativoInfinito = -100 / 0.0;
```

```
// número não definido (NaN)
System.out.println(positivoInfinito + negativoInfinito);
```

COPIAR CÓDIGO