Operadores

Operadores

- Java fornece um ambiente rico em operadores. Um operador é um símbolo que solicita ao compilador que execute uma operação matemática ou lógica específica.
- Java tem quatro classes gerais de operadores:
  - aritmético;
  - bitwise;
  - relacional;
  - lógico.

Operadores

- Também define alguns operadores adicionais que tratam certas situações especiais.
- Aqui falaremos sobres operadores aritméticos, relacionais e lógicos. Também examinaremos o operador de atribuição.
- O operador bitwise e outros operadores especiais serão explorados em outro momento.

Operadores Aritméticos

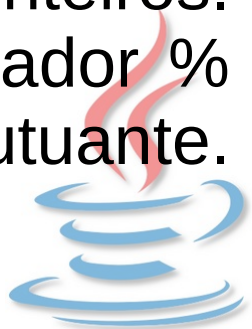
Operador	Significado
+	Adição (também mais unário)
-	Subtração (também menos unário)
*	Multiplicação
/	Divisão
%	Módulo
++	Incremento
--	Decremento

Operadores Aritméticos

- Os operadores +, -, * e / funcionam em java da mesma maneira que em qualquer outra linguagem de computador(ou em álgebra).
- Eles podem ser aplicados a qualquer tipo de dado numérico interno. Também podem ser usados em objetos de tipo **char**.
- Embora as ações dos operadores aritméticos sejam conhecidas por todos os leitores, algumas situações especiais pedem explicação.

Operadores Aritméticos

- Primeiro, lembre-se de que quando / é aplicado a um inteiro, o resto da divisão é truncado; por exemplo 10/3 será igual a 3 na divisão de inteiros.
- Você pode obter o resto dessa divisão usando o operador de módulo %.
- Ele funciona em Java da mesma forma que em outras linguagens: gerando o resto de uma divisão de inteiros. Por exemplo, 10%3 é igual a 1. Em Java, o operador % pode ser aplicado a tipos inteiros e de ponto flutuante. Logo 10.0%3.0 também é igual a 1.



Demonstrando o operador

```
//Demonstra o operador %
public class ModDemo {
   public static void main(String[] args) {
      int iresult, iresMod;
      double dresult, dresMod;
      iresult = 10/3;
      iresMod = 10%3;
      dresult = 10.0/3.0;
      dresMod = 10.0%3.0;
      System.out.println("O resultado e o resto de
10/3 : " + iresult + " e " + iresMod);
      System.out.println("O resultado e o resto de
10.0/3.0 : " + dresult + " e " + dresMod);
   }
}
```

Incremento e Decremento

- ++ e – – são operadores Java de incremento e decremento. Eles tem algumas propriedades especiais que os tornam muito interessantes.
- O operador de incremento adiciona 1 a seu operando:

x = x + 1;

É o mesmo que

X++;

- O operador de decremento subtrai 1 a seu operando:

x = x – 1;

É o mesmo que

X– –;

Incremento e Decremento

- Tanto o operador de incremento quanto o de decremento podem preceder(prefixar) ou vir após(posfixar) o operando.

x = x + 1;

Pode ser escrito como

++x; // forma prefixada

ou como

– –x; // forma posfixada

Incremento e Decremento

- No exemplo anterior, não há diferença se o incremento é aplicado como um prefixo ou um posfixo.
- No entanto quando um incremento ou decremento é usado como parte de uma expressão maior, há uma diferença importante.
- Quando um operador de incremento ou decremento precede seu operando, Java executa a operação correspondente antes de obter o valor do operando a ser usado pelo resto da expressão.

Incremento e Decremento

- Se o operador vier após seu operando, Java obterá o valor do operando antes de ele ser incrementado ou decrementado:

X = 10;

Y = x++;

Nesse caso, y será configurado com 11.

Incremento e Decremento

No entanto se o código for escrito como:

X = 10;

Y = y++;

então y será configurado com 10. Nos dois casos, x é configurado com 11; a diferença é quando isso ocorre. Há vantagens significativas em podermos controlar quando a operação de incremento ou decremento deve ocorrer.

Relacionais e Lógicos

- Nos termos *operador relacional* e *operador lógico, relacional* se refere aos relacionamentos que os valores podem ter uns com os outros e *lógico* se refere às maneiras como os valores verdadeiro e falso podem estar conectados.
- Já que os operadores relacionais produzem resultados verdadeiros ou falsos, com frequência trabalham com os operadores lógicos.

Operadores Relacionais

Operador	Significado
==	Igual a
!=	Diferente de
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a

Operadores Lógicos

Operador	Significado
&	AND
|	OR
^	XOR(exclusive OR)
||	OR de curto-circuito
&&	AND de curto-circuito
!	NOT

Operadores Lógicos

- Quanto aos operadores lógicos, os operandos devem ser de tipo **boolean** e o resultado de uma operação lógica é de tipo **boolean.**
- Os operadores lógicos **&, |, ^** e **!** dão suporte às operações lógicas básicas AND, OR, XOR, NOT, de acordo com a tabela verdade a seguir:

Operadores Lógicos

P	Q	P & Q	P | Q	P ^ Q	!P
FALSO	FALSO	FALSO	FALSO	FALSO	VERDADEIRO
VERDADEIRO	FALSO	FALSO	VERDADEIRO	VERDADEIRO	FALSO
FALSO	VERDADEIRO	FALSO	VERDADEIRO	VERDADEIRO	VERDADEIRO
VERDADEIRO	VERDADEIRO	VERDADEIRO	VERDADEIRO	FALSO	FALSO

Operadores Lógicos de curto-circuito

- Java fornece versões especiais de curto-circuito de seus operadores lógicos AND e OR que podem ser usadas para produzir código mais eficiente.
- Em uma operação AND, se o primeiro operando for falso, o resultado será falso não importando o valor do segundo operando. Em uma operação OR, se o primeiro operando for verdadeiro, o resultado da operação será verdadeiro não importando o valor do segundo operando.

Operadores Lógicos de curto-circuito

- Logo, nestes dois casos, não há necessidade de avaliar o segundo operando. Quando não avaliamos o segundo operando, economizamos tempo e um código mais eficiente é produzido.
- O operador AND de curto-circuito é && e o operador OR de curto-circuito é ||. Seus equivalente comuns são & e |. A única diferença entre as versões comum e de curto-circuito é que a versão comum sempre avalia cada operando e a versão curto-circuito só avalia o segundo operando quando necessário.

Operador de atribuição

- O operador de atribuição é o sinal de igual simples, =. Esse operador funciona em Java de modo bem parecido a como funciona em qualquer outra linguagem de computador. Ela tem esta forma geral:

var = expressão;

- Aqui o tipo de var deve ser compatível com o tipo de expressão. O operador de atribuição tem uma propriedade interessante que talvez você não conheça: ele permite a criação de uma cadeia de atribuições.

Operador de atribuição

- Por exemplo considere este fragmento:

```
int x,y,x;

x = y = z = 100; // cofigura x,y,e z com 100
```

- Ele configura as variáveis x, y e z com 100 usando a mesma instrução. Isso funciona porque = é um operador que fornece o valor da expressão do lado direito. Log, o valor de z = 100 é 100, que é então atribuído a y, que por sua vez é atribuído a x. O uso de uma “cadeia de atribuição” é uma maneira fácil de configurar um grupo de variáveis com um valor comum.

Exercício

- Criar um programa para exibir a tabela verdade dos operadores lógicos de Java. Esse projeto faz uso de vários recursos abordados, inclusive uma das sequências de escape Java e o s operadores lógicos.
- Para assegurar que as colunas fiquem alinhadas, usar a sequência de escape ***/t*** para embutir tabulações em cada string de saída.
- O nome do arquivo Java será **LogicalOpTable.java**