# PROGRAMAÇÃO ORIENTADA A OBJETOS

Conversão de tipos, operadores de atribuição compostos e operadores de incremento e decremento

Prof. Emanoel Barreiros

# Conversão de tipos

- A conversão pode ser explícita ou implícita
- O que você acha que acontece nesse trecho de código?

```
double operando1 = 14.0;
double operando2 = 3.0;
int resultado = operando1 / operando2;
```

# Conversão implícita

```
double operando1 = 14.0;
double operando2 = 3.0;
int resultado = operando1 / operando2;
```

- Nesse caso ocorre a conversão implícita. Você vê algum comando para conversão? Não ☺
- O valor 14.0 é dividido por 3.0, obtendo o resultado 4.66666… mas apenas a parte inteira é considerada, desprezando-se a parte decimal.
- O valor resultante na variável resultado é 4 (inteiro)
- O que ocorreu foi a conversão implícita de um valor double para um valor int

# Conversão implícita: CUIDADO!

- Quando ocorre a conversão implícita você pode perder informação
- A conversão de double para int é um exemplo de perda de informação
- Em geral, quando se converte algo de um tipo “maior” para um “menor”, pode-se perder dados.

# Conversão explícita

- A conversão explícita pode nos ajudar a resolver alguns problemas
- Explícita significa que você está dizendo ao Java que deseja realizar a conversão naquele ponto
- Cria um valor temporário para ser usado na expressão
- Também chamado de cast

# Conversão explícita

- O que acontece abaixo?

```
int operando1 = 14;
int operando2 = 3;
double resultado = operando1 / operando2;
```

- O valor 14 é dividido por 3, resultando em 4
- Apenas a parte inteira é considerada, pois os operandos são inteiros
- Podemos resolver isso com conversão explícita

# Conversão explícita

- Resolvendo a bronca

```
int operando1 = 14;
int operando2 = 3;
double resultado = (double) operando1 / operando2;
```

- O cast cria um valor double temporário que será atribuído à variável resultado

# Operadores de atribuição compostos

- Operadores compostos de atribuição abreviam expressões

- Expressões da forma
  - variavel = variavel  operador  expressao
- Se tornam
  - variavel  operador= expressao

- Ex:
  - c = c + 3;
  - c += 3

# Operadores de atribuição compostos

| Operador | Exemplo | Significado | Atribui |
|---|---|---|---|
| Use int c = 3, d = 5, e = 4, f = 6, g = 12 | | | |
| += | c += 7 | c = c + 7 | 10 a *c* |
| -= | d -= 4 | d = d - 4 | 1 a *d* |
| *= | e *= 5 | e = e * 5 | 20 a *e* |
| /= | f /= 3 | f = f / 3 | 2 a *f* |
| %= | g %= 9 | g = g % 9 | 3 a *g* |

# Operadores de incremento e decremento

| Operador | Nome | Exemplo | Significado |
|---|---|---|---|
| ++ | Incremento prefixo | ++a | Incrementa de 1, depois usa o valor na expressão onde foi definido |
| ++ | Incremento posfixo | a++ | Usa o valor atual de *a* na expressão, depois incrementa de 1 |
| -- | Decremento prefixo | --b | Decrementa de 1, depois usa o valor na expressão onde foi definido |
| -- | Decremento posfixo | b-- | Usa o valor atual de *b* na expressão, depois decrementa de 1 |

# Exemplo

```
public class Incremento {
    public static void main( String[] args ) {
        int c;
        c = 5;
        System.out.println( c );
        System.out.println( c++ );
        System.out.println( c );

        System.out.println();

        c=5;
        System.out.println( c );
        System.out.println( ++c );
        System.out.println( c );
    }
}
```