

# PROGRAMAÇÃO ORIENTADA A OBJETOS

---

Mais controles de repetição

Prof. Emanuel Barreiros



# O Comando **switch**


- Comando de seleção múltipla
- Executa diferentes ações de acordo com o valor de uma expressão constante de tipo **byte**, **short**, **int** ou **char**

```
int variavel;  
  
switch (variavel) {  
    case 0 :  
        comandos;  
        break; //obrigatorio  
    case 1 :  
        comandos;  
        break; //obrigatorio  
    default :  
        comandos;  
        break; //opcional  
}
```

# Comando **break**

- O comando **break** é usado para causar a imediata saída do comando dentro do qual foi usado
  - Pode ser usado com os comandos de repetição **while**, **for**, **do...while** e **switch**
- A execução continua na primeira linha após o bloco do comando abortado

```
for (int i = 0; i < 100; i++) {  
    if (i == 10) {  
        break;  
    }  
}
```



# Comando **continue**

- O comando **continue**, quando executado dentro um laço **for**, **while** e **do...while**, ignora os demais comandos abaixo dele e inicia uma nova iteração
- Em laços **while** e **do...while**, a condição de continuação do loop é executada imediatamente após a execução do comando **continue**
- Em laços **for**, a expressão de incremento (ou decremento) é executada e a condição de continuação do loop é avaliada

# Comando **continue**

- Exemplo:

```
for (int i = 0; i < 100; i++) {  
    if (i == 10) {  
        continue;  
  
        System.out.println("Nunca será executado...");  
    }  
}
```

# Operadores booleanos

- O AND é representado por &&
- O OR é representado por ||
- Os operadores && e || são preguiçosos (lazy evaluation)
  - Se já se souber o resultado de uma expressão lógica antes de toda sua avaliação, o Java não fará a avaliação de toda a expressão

# Operadores lógicos

- Também existe os operadores & e |
- O AND lógico (&) e o OR lógico (|) testam todas as possibilidades, mesmo que o resultado já seja conhecido
- Não são preguiçosos

# Mais operadores lógicos

- OU-exclusivo ( $\wedge$  em Java, não confunda com o AND da lógica tradicional)
  - Este operador só leva o resultado para **true** se apenas um dos operandos for **true**

| expressao<br>1 | expressao<br>2 | expressao1 $\wedge$ expressao2 |
|----------------|----------------|--------------------------------|
| false          | false          | false                          |
| false          | true           | true                           |
| true           | false          | true                           |
| true           | true           | false                          |



# Exercício

1. Crie uma pequena calculadora usando o comando switch. O usuário deve informar a operação a ser realizada, depois informar os operandos e exibir o resultado
2. Escreva um método que calcule a o n-ésimo valor na sequencia de Fibonacci
3. Crie uma aplicação que gere um gráfico de barras horizontais com sinais de igual. Receba do usuário um inteiro que representa a quantidade de itens que serão informados, depois obtenha os números e os exiba da seguinte forma:
  1. 3 ===
  2. 7 =====
  3. 5 =====
  4. 2 ==