

transforme ■ se



JAVA

AULA 2 - ESTRUTURAS CONDICIONAIS E REPETIÇÃO

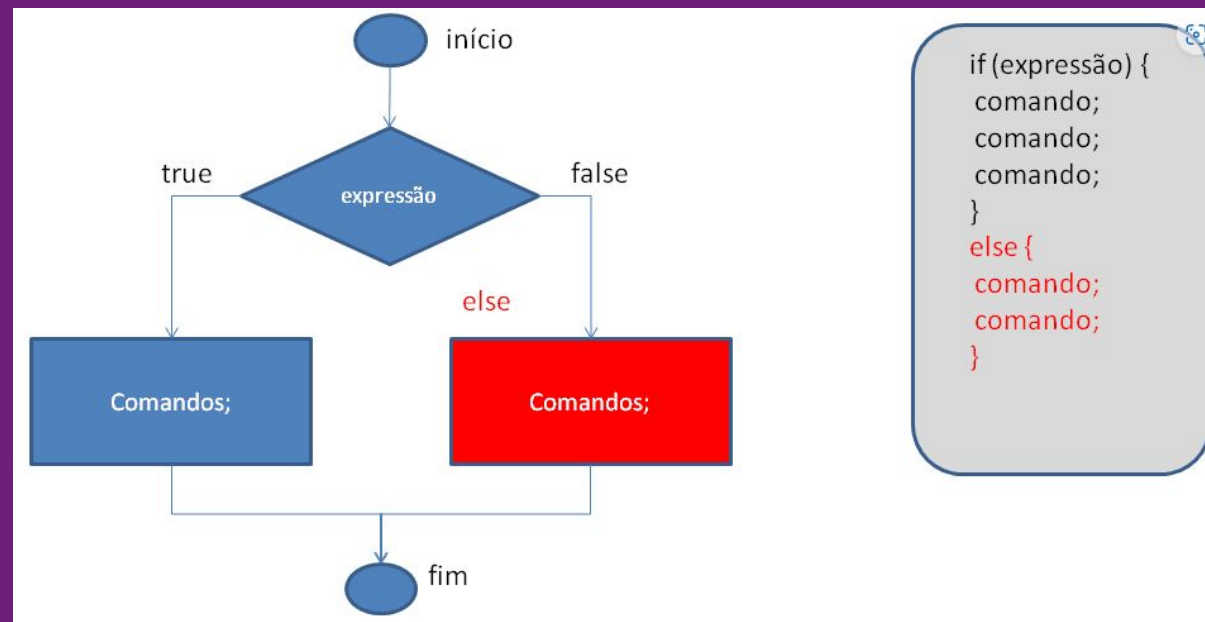


ESTRUTURAS CONDICIONAIS




Estruturas condicionais

As estruturas condicionais existem em todas as linguagens de programação e possibilitam que a execução de um programa seja desviada de acordo com certas condições. Os comandos condicionais (ou ainda instruções condicionais) usados em Java são if-else e switch-case. Essas duas estruturas de desvio existentes na linguagem possibilitam executar diferentes trechos de um programa com base em certas condições.



A estrutura if-else

O if, em conjunto com o else, forma uma estrutura que permite a seleção entre dois caminhos distintos para execução, dependendo do resultado (verdadeiro ou falso) de uma expressão lógica (condição). Nesse tipo de estrutura, se a condição for verdadeira, são executadas as instruções que estiverem posicionadas entre as instruções if/else. Sendo a condição falsa, são executadas as instruções que estiverem após a instrução else. A sintaxe para a utilização do conjunto if else é demonstrada em seguida. Observe que a condição sempre deve aparecer entre parênteses, item obrigatório na linguagem Java.

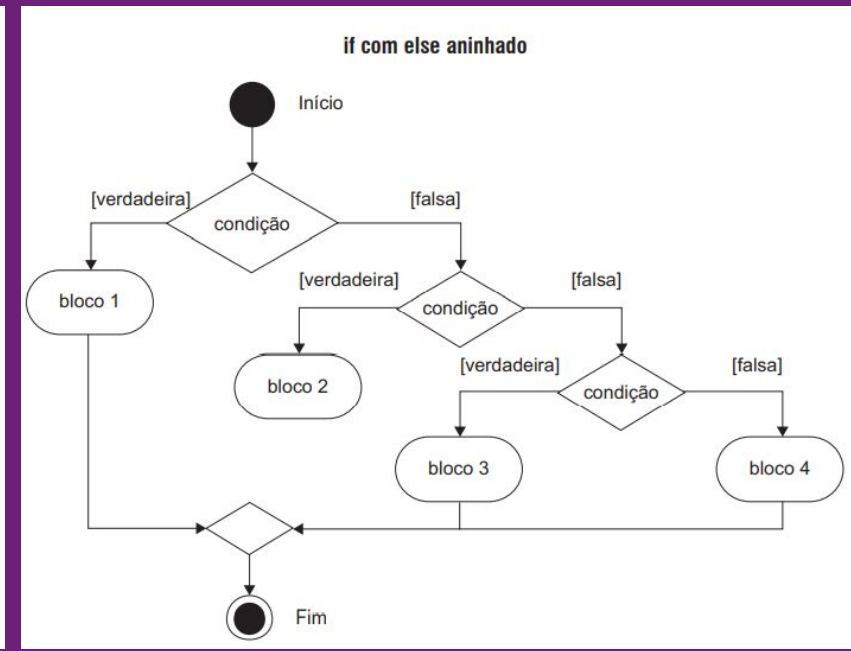
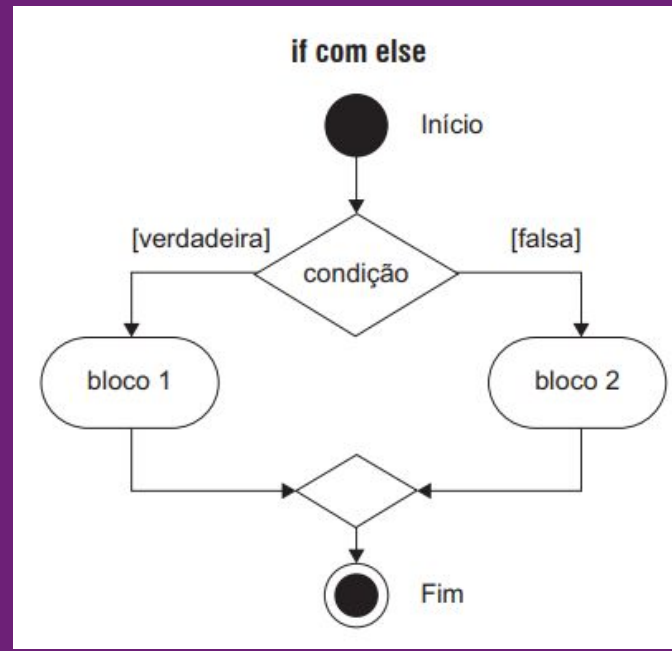
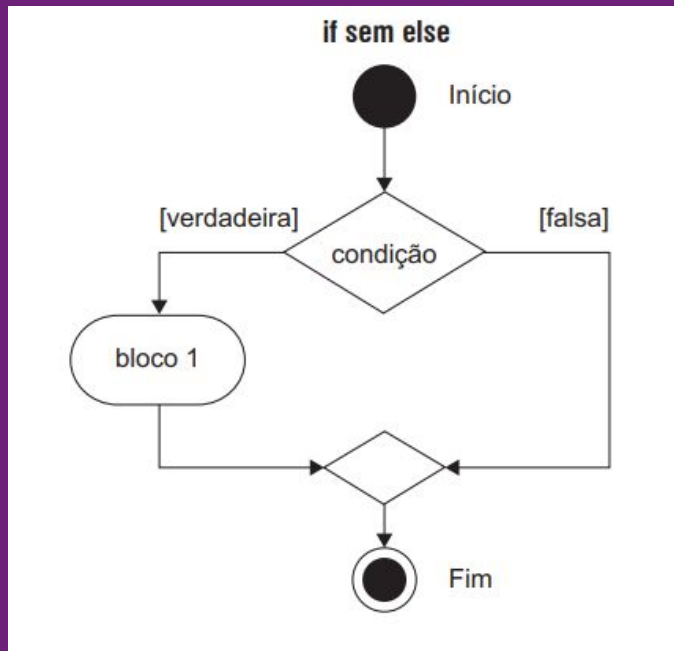


```
1  public class Condicionais {  
2  
3      if (<Condição>) {  
4          <Instruções para condição verdadeira>  
5      } else {  
6          <Instruções para condição falsa>  
7      }  
8  
9  }
```

A estrutura if-else

Assim como a maioria das instruções em Java, o conjunto if-else deve ser utilizado com minúsculas e, caso haja apenas uma instrução a ser executada, tanto no if como no else, o uso das chaves é desnecessário. Lembre-se de que as chaves são utilizadas quando um bloco de instruções precisa ser executado, isto é, mais do que uma instrução.

A estrutura if-else apresentada não é a única válida, pois existem outras maneiras diferentes de se criar essa estrutura: if sem o else, if com o else e if com o else aninhado



A estrutura if-else

Estrutura 1: if sem else

O Exemplo mostra um uso prático do if sem a presença do else. Trata-se de uma classe em que o usuário seleciona uma opção (Masculino ou Feminino) e a partir disso é usada a instrução if para executar instruções diferentes.

```
1 public class Condicionais {
2
3     public static void main(String[] args) {
4         Object[] opcoes = {"Masculino", "Feminino"};
5         String resposta = (String) JOptionPane.showInputDialog(
6             null,
7             "Selecione o sexo:\n ",
8             "Pesquisa",
9             JOptionPane.PLAIN_MESSAGE,
10            null,
11            opcoes,
12            "Masculino");
13
14         if(resposta == null) {
15             JOptionPane.showMessageDialog(null, "Você pressionou cancel");
16         }
17         if(resposta == "Masculino") {
18             JOptionPane.showMessageDialog(null, "Você selecionou o sexo Masculino");
19         }
20         if(resposta == "Feminino") {
21             JOptionPane.showMessageDialog(null, "Você selecionou o sexo Feminino");
22         }
23
24         System.exit(0);
25     }
26
27 }
```

A estrutura if-else

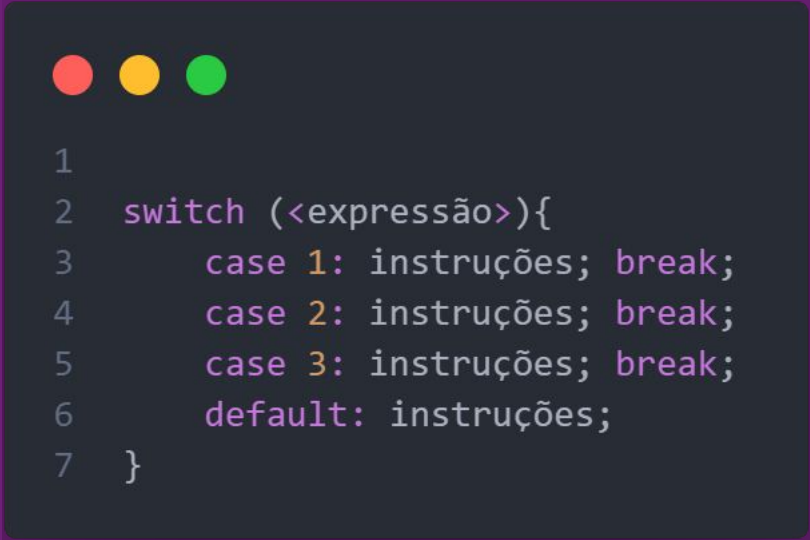
Exercício proposto!

Monte uma classe que vai receber o número de um mês e verifique se é válido e mostre escrito na tela o nome do mês, por exemplo: Usuário digitou 5 o programa mostra: Maio. Caso o usuário digite um mês inválido, mostre a mensagem: “Mês inválido”

A estrutura switch-case

A estrutura switch-case se refere a uma outra modalidade de desvio da execução do programa de acordo com certas condições, semelhante ao uso da instrução if. Ao trabalhar com uma grande quantidade de desvios condicionais contendo instruções if, pode-se comprometer a inteligibilidade do programa, dificultando sua interpretação. A estrutura switch-case possibilita uma forma mais adequada e eficiente de atender a esse tipo de situação, constituindo-se uma estrutura de controle com múltipla escolha.

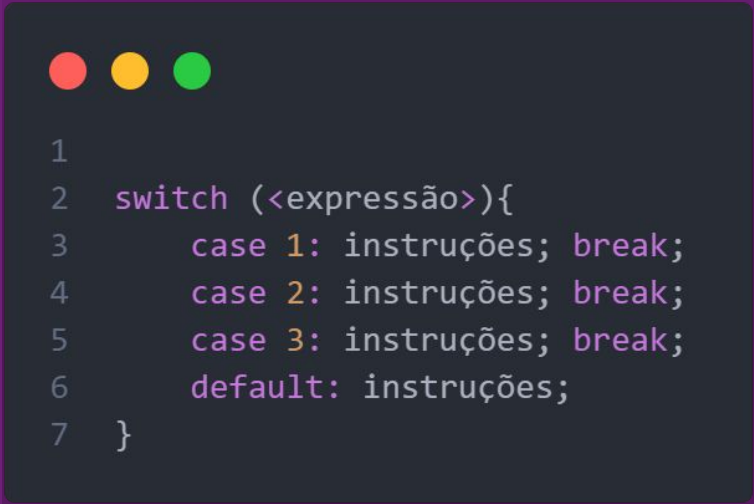
A estrutura switch-case equivale a um conjunto de instruções if encadeadas, fornecendo maior inteligibilidade. Sua sintaxe é a seguinte:



```
1  
2  switch (<expressão>){  
3      case 1: instruções; break;  
4      case 2: instruções; break;  
5      case 3: instruções; break;  
6      default: instruções;  
7  }
```

A estrutura switch-case

Na primeira linha do switch é avaliado o resultado da expressão, que é comparado nas diretivas case, executando o bloco de instruções quando a expressão coincidir com o valor colocado ao lado direito do case. Em outras palavras, supondo que o valor da expressão seja igual a 2, serão executadas as instruções localizadas entre case 2: e break. A cada case o programa compara o valor da expressão com o valor colocado no case. Caso os valores sejam iguais, todas as instruções são executadas até que se encontre uma instrução break, que encerra o switch e faz a execução do programa desviar para o ponto após a chave de encerramento do switch. O programa percorre todas as diretivas case até que uma delas seja igual à expressão inserida no switch. Caso nenhuma diretiva case possua o valor correspondente da expressão, serão executadas as instruções localizadas na diretiva default que é opcional.



```
1
2  switch (<expressão>){
3      case 1: instruções; break;
4      case 2: instruções; break;
5      case 3: instruções; break;
6      default: instruções;
7  }
```

Laços de repetição



Laços de repetição

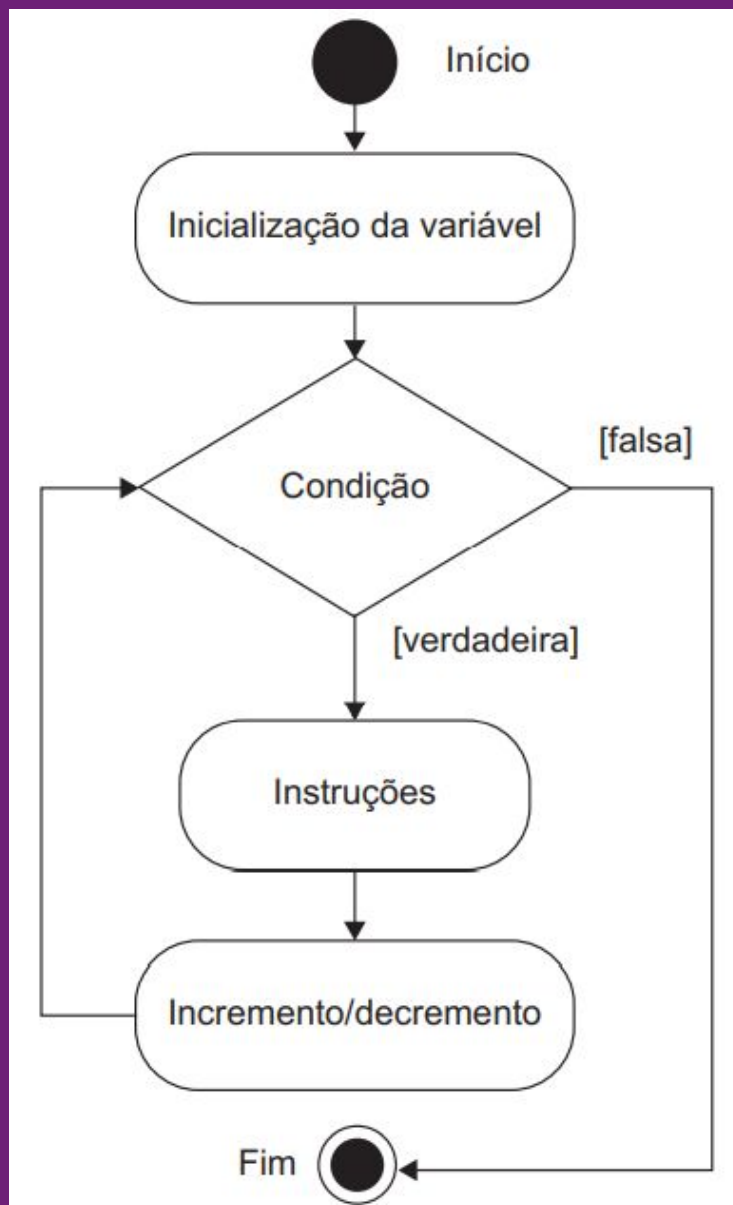
Os laços de repetição (looping) formam uma importante estrutura nas linguagens de programação por possibilitarem a repetição da execução de um bloco de instruções em um programa. Eles determinam que um certo bloco seja executado repetidamente até que uma condição específica ocorra. A repetição é uma das estruturas mais usadas em programação, possibilitando a criação de contadores, temporizadores, rotinas para classificação, obtenção e recuperação de dados. A criação de laços de repetição em Java é feita a partir das estruturas `for`, `while` e `do-while`.

Uso do laço for

A instrução for é um tipo de contador finito, isto é, ela realiza a contagem de um valor inicial conhecido até um valor final também conhecido. Uma possível representação gráfica da estrutura de funcionamento de um laço de repetição pode ser visualizada na figura na próxima tela. No início da execução da estrutura é inicializada uma variável. Após isso, o valor dessa variável é verificado na condição (losango), e enquanto essa condição for verdadeira o bloco de instruções será executado dentro da estrutura.

Somente quando a condição se tornar falsa é que a execução será desviada para o final da estrutura do laço. O incremento ou decremento do valor da variável é essencial para que o laço tenha uma saída (encerre), caso contrário a execução nunca sairia do laço.

Uso do laço for



Uso do laço for

A estrutura de repetição com for contém uma variável de controle do tipo contador, que pode ser crescente ou decrescente e possui a seguinte sintaxe:

- **for** (inicialização; condição; incremento ou decremento)

Em que:

- **Inicialização**: é o valor inicial da variável de controle do laço.
- **Condição**: contém uma expressão booleana que será usada para controlar a continuidade do laço. Deve conter o valor final que a variável de controle pode assumir dentro do laço.
- **Incremento** ou decremento: é o passo com que a variável de controle será acrescida ou decrescida. Esse incremento pode ser realizado por meio de uma variável inteira ou com ponto flutuante, permitindo pequenos incrementos decimais.



```
1  for (int cont=0; cont<10; cont++){  
2      <conjunto de instruções>  
3  }
```

Uso do laço for


Essa sintaxe pode ser interpretada como: inicialize a variável `cont` com zero e repita o conjunto de instruções enquanto o valor de `cont` for menor que 10. Cada vez que o conjunto de instruções é executado, o valor do `cont` é incrementado (`cont++`). Observe que a variável `cont` pode ser declarada e inicializada na própria estrutura do `for`. Quando isso ocorre, a variável passa a ter escopo local ao `for`, isto é, seu valor é reconhecido apenas dentro da estrutura em que foi declarada (entre as chaves de abertura e encerramento). Após o encerramento do laço, a variável perde sua referência e não pode mais ser usada, salvo se declarada novamente. Se você desejar usar o valor de uma variável após o encerramento do laço, basta declará-la antes de iniciar o laço, isto é, antes da linha que define o `for`.



```
1  for (int cont=0; cont<10; cont++){  
2      <conjunto de instruções>  
3  }
```


Uso do laço while

O while é outro laço condicional, isto é, um conjunto de instruções que são repetidas enquanto o resultado de uma expressão lógica (uma condição) é avaliado como verdadeiro. Veja a seguir sua sintaxe:

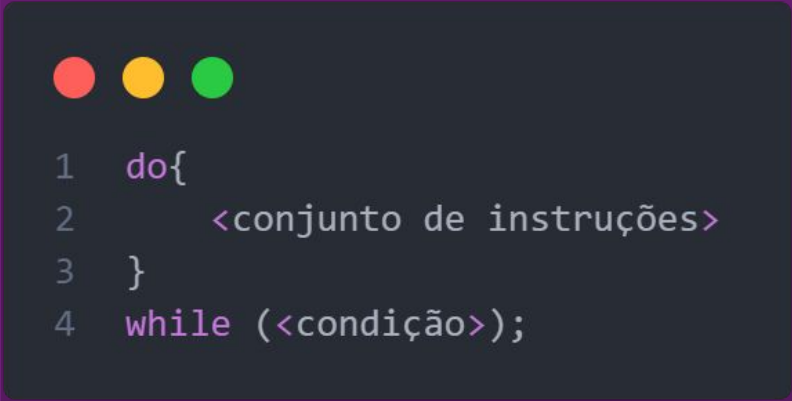


```
1  while (<condição>){  
2      <conjunto de instruções>  
3  }
```

A instrução while avalia o resultado da expressão (condição) antes de executar as instruções do bloco { }, assim é possível que as instruções nunca sejam executadas, caso a condição seja inicialmente falsa. Um problema típico, relacionado à avaliação da condição while, é o laço infinito. Caso a condição nunca se torne falsa, o laço será repetido infinitamente.

Uso do do...while

Há outro tipo de laço condicional, o chamado do-while, que é bem parecido com o while, porém o conjunto de instruções é executado antes da avaliação da expressão lógica. Isso faz com que essas instruções sejam executadas pelo menos uma vez. Veja como a sintaxe do do-while é bem parecida com a do while:



```
1  do{  
2      <conjunto de instruções>  
3  }  
4  while (<condição>);
```

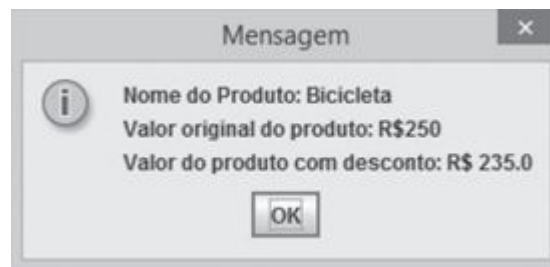
EXERCÍCIOS



Exercícios

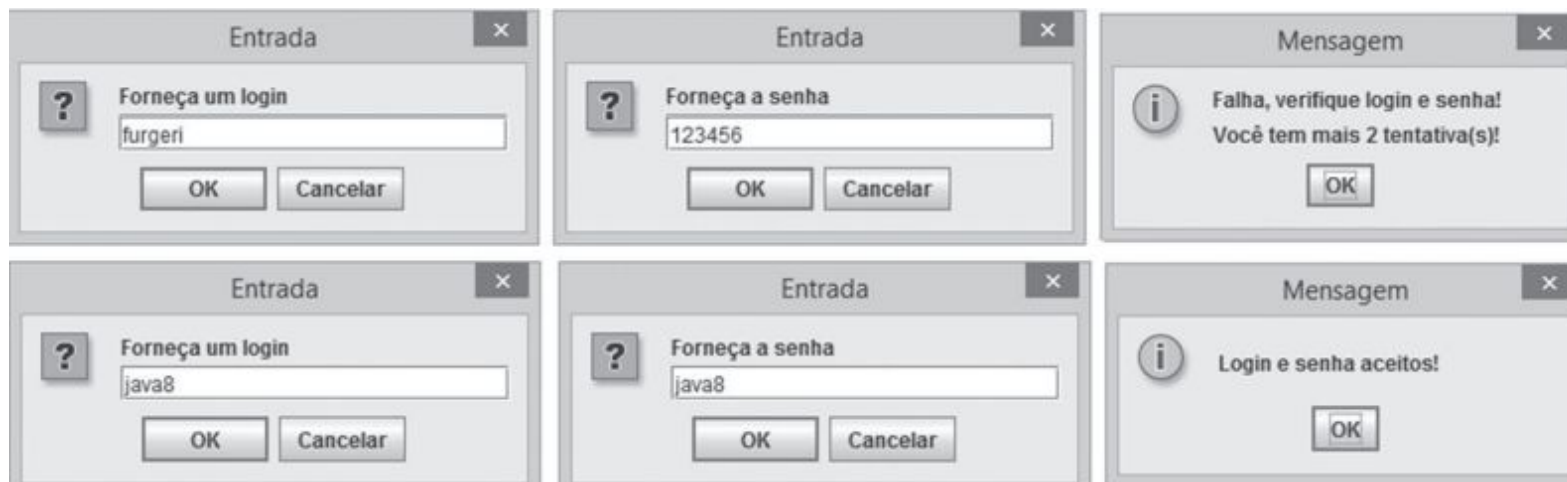
1. Usando JOptionPane, elabore uma classe que receba o nome de um produto e o seu valor. O desconto deve ser calculado de acordo com o valor fornecido conforme abaixo. Utilizando a estrutura if-else, apresente em tela o nome do produto, valor original do produto e o novo valor depois de ser realizado o desconto. Caso o valor digitado seja menor que zero, deve ser emitida uma mensagem de aviso.

Valor (R\$)	Desconto (%)
>=50 e <200	5
>=200 e <500	6
>=500 e <1000	7
>=1000	8



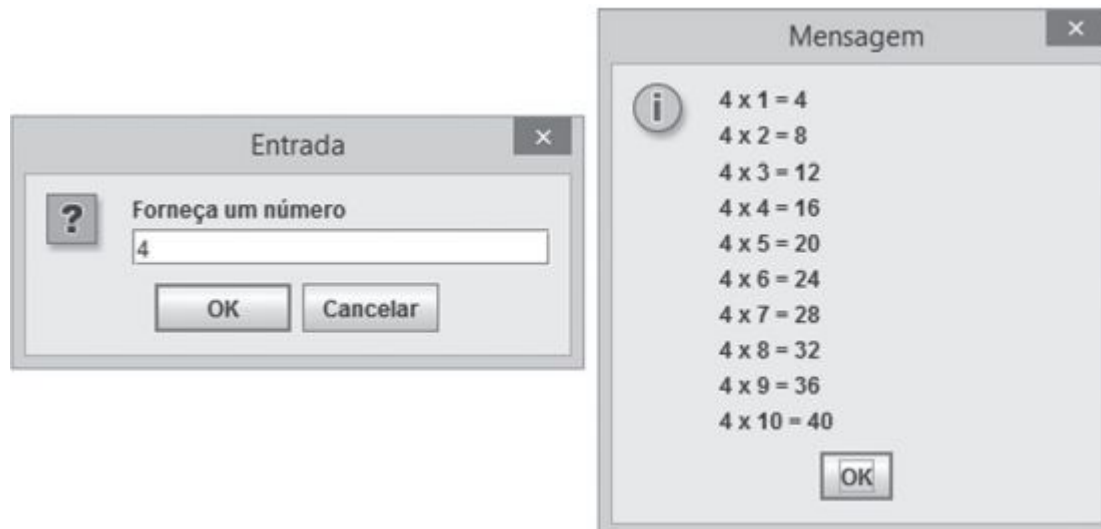
Exercícios

2. Faça uma classe que solicite login e senha, simulando o acesso a um sistema. Considere que os conteúdos de login e senha originais são iguais a “java8”. O usuário deverá fornecer login e senha e você irá compará-los com os conteúdos originais. O usuário tem três chances para digitar corretamente os dados de login e senha. Para cada tentativa incorreta deve aparecer uma mensagem alertando o erro e apresentando a quantidade de tentativas que ainda restam. Veja um exemplo de execução abaixo, em que o usuário já digitou o login e senha incorretos por duas vezes, restando apenas uma última chance.



Exercícios

3. Faça uma classe que apresente em tela a tabuada de qualquer número. O usuário fornece o número desejado e a classe apresenta a relação de todos os cálculos de 1 a 10.



transforme ■ se

O conhecimento é o poder
de transformar o seu futuro.