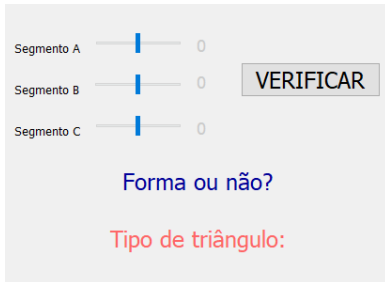


# EXERCÍCIOS DE JAVA 10

## Exercícios Formas de Triângulo Swing

### Rascunho:



OBS: foi utilizado 3 controles deslizantes (sliders).

OBS 2: o “Forma ou não” e o “Tipo de triângulo” estão dentro de um **painel**.

### Variáveis:

Labels:	Sliders:	Painel:	Labels no painel:	Botão:
lblA	sliA	panResposta	lblStatus	btnVerificar
lblB	sliB		lblTipo	
lblC	sliC			

### Método construtor (esconder painel):

Comando para esconder:

```
public classes() {  
    initComponents();  
    panResposta.setVisible(false);  
}
```

### Código-fonte:

Comando para aparecer o painel (depois de ativar o evento do botão):

```
private void btnVerificarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    panResposta.setVisible(true);  
}
```

Comando para os sliders (selecionar os 3):

- Mudar o **maximum** para: 20
- **Minimun**: 0
- **Value**: 0
- Para cada slider: botão direito – Eventos – Change – stateChanged

```
private void sliAStateChanged(javax.swing.event.ChangeEvent evt) {  
    // TODO add your handling code here:  
    lblA.setText(Integer.toString(sliA.getValue()));  
}
```

OBS: programar isso para cada slider!

## Comando para verificar se esses três segmentos formariam um triângulo:

Lembrando: para formar um triângulo, cada segmento tem que ser menor que a soma dos outros dois!

Ou seja:  $a < b + c$  &&  $b < a + c$  &&  $c < a + b$

```
private void btnVerificarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    int a = sliA.getValue();  
    int b = sliB.getValue();  
    int c = sliC.getValue();  
    if (a<b+c && b<a+c && c<a+b) {  
        lblStatus.setText("Forma um Triângulo");  
    } else {  
        lblStatus.setText("Não forma um triângulo");  
    }  
}
```

## Comando para o tipo de triângulo:

Lembrando:  $!=$  significa não igual (ou seja, é diferente)

Exemplo  $a != b$  (a é diferente de b)

```
lblStatus.setText("Forma um Triângulo");  
if (a==b && b==c) {  
    lblTipo.setText("Equilátero");  
} else if (a!=b && b!=c && a!=c) {  
    lblTipo.setText("Escaleno");  
} else {  
    lblTipo.setText("Isósceles");  
}
```

The diagram shows four arrows pointing from specific lines of code to explanatory boxes:

- From `lblStatus.setText("Forma um Triângulo");` to a box: "O comando está dentro do lblStatus para formar um triângulo!"
- From `lblTipo.setText("Equilátero");` to a box: "Fórmula para o equilátero (todos os lados iguais)"
- From `lblTipo.setText("Escaleno");` to a box: "Fórmula para o escaleno (todos os lados diferentes)"
- From `lblTipo.setText("Isósceles");` to a box: "Como o isósceles está no último else, então não precisa fórmula!"

## Comando completo:

```
private void btnVerificarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    int a = sliA.getValue();  
    int b = sliB.getValue();  
    int c = sliC.getValue();  
    if (a<b+c && b<a+c && c<a+b) {  
        lblStatus.setText("Forma um Triângulo");  
        if (a==b && b==c) {  
            lblTipo.setText("Equilátero");  
        } else if (a!=b && b!=c && a!=c) {  
            lblTipo.setText("Escaleno");  
        } else {  
            lblTipo.setText("Isósceles");  
        }  
    } else {  
        lblStatus.setText("Não forma um triângulo");  
        lblTipo.setText("----");  
    }  
    panResposta.setVisible(true);  
}
```

Dica: sempre cuidar com a indentação do código (saber onde está cada if, cada else)