



# Módulo 10





A decorative graphic on the left side of the slide. It features a large central hexagon with a blue-to-teal gradient, containing the number '1'. Surrounding this central hexagon are several smaller hexagons of varying shades of blue and teal. Some of these smaller hexagons contain white icons: a lightbulb, a thumbs-up, a smartphone, a magnifying glass, and a gear. There is also a network-like icon with a central node and radiating lines, and a speech bubble icon.

1

# Controle de fluxos



# Controle de fluxos

O controle de fluxo é efetuado por condicionais que modificam o percurso do programa. Em java existem alguns controles:

- if
- else
- else if
- for
- while
- do while
- switch
- break
- continue





# if, else e else if

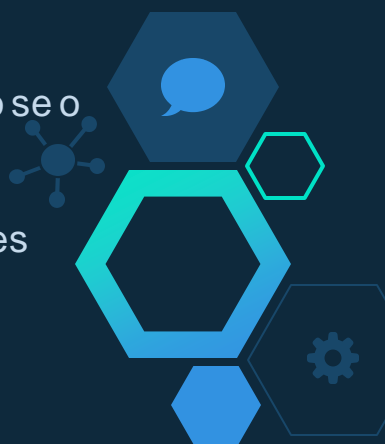
São tomadas de decisões que definem o que é verdadeiro e o que é falso.

IF significa que a instrução é verdadeira e else que é false.

Se (if) for tal coisa, faça isso, caso contrário (else), faça aquela outra coisa.

O if pode ser declarado sozinho mas o else só pode ser definido se o if for definido primeiro.

Em conjunto com os controles de fluxos utilizamos os operadores lógicos.





# if, else e else if

Exemplo 1:

```
int result = 0;  
  
if (result > 1) {  
  
} else {  
  
}
```





# if, else e else if

Exemplo 2:

```
int result = 0;  
  
if (result > 1 && result < 5) {  
  
} else if (result >= 5 && result < 8) {  
  
} else {  
  
}
```





# if, else e else if

Exemplo 3:

```
int num = 10;  
if (num >= 5) {  
    if (num >= 8) {  
  
    } else if (num >= 6) {  
  
    } else {  
  
    }  
} else {  
    if (num >= 1 && <= 3) {  
  
    } else {  
  
    }  
}
```





A decorative pattern of hexagons in various shades of blue and cyan. Some hexagons contain icons: a lightbulb, a thumbs up, a network of nodes, a smartphone, a magnifying glass, a gear, and a speech bubble. The number '2' is centered in a large cyan hexagon.

2

Loop com for, while, do  
while, break and  
continue



# for

O loop for é mais fixo, permite executar o conjunto de sentenças por um número determinado de vezes. O princípio do loop for é ser um contador. Exemplo:

```
for(int i = 0; i < 10; i++) {  
    System.out.println("Linha " + i);  
}
```





# break e continue

São dois comandos de controle que são usados juntos com os controles de fluxos for e while.



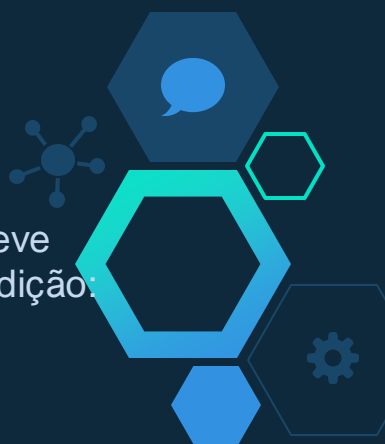


# break

O comando break faz com que um loop pare em uma determinada execução.

```
public static void main (String args []) {  
    for (int contador=1; contador<=1000; contador++){  
        System.out.println("Esta é a repetição nr: "+contador);  
        if (contador==10)  
            break;  
    }  
}
```

No exemplo acima, temos uma repetição que se inicia em 1 e deve terminar em mil (1.000), mas dentro desta estrutura há uma condição: se a variável for igual a 10 saia da estrutura de repetição.






# continue

O comando continue é diferente do comando break, ele não para a execução e sim continua.

```
public static void main (String args []){  
    for (int contador=1; contador<=100; contador++){  
        if (contador%5!=0)  
            continue;  
        System.out.println("Contador: "+contador);  
    }  
}
```

O código acima conta de 1 a 100, mas só imprime os números múltiplos por 5, Ignorando o código que imprime os valores.





# while

É um fluxo de controle que executa uma comparação, caso esta comparação seja verdadeira o fluxo entra no loop. Esta comparação é feita em primeiro lugar, caso o resultado seja verdadeiro ele executa o código dentro do loop.

```
int count = 0;
while(count < 2) {
    System.out.println("Repetição nr: " + contador);
}
```





# while

O loop while é um loop infinito necessitando que uma variável seja modificada a cada loop executado.

```
int count = 0;
while(count < 2) {
    System.out.println("Repetição nr: " + contador);
    count++;
}
```





# Diferenças for e while


A principal diferença entre o controle de fluxo for e o while é que o for você sabe quando a sua execução vai parar, pois você atribui uma variável de controle. Já no while ele continuará executando o código até que uma condição (Você não sabe quando isso irá ocorrer) seja atingida.

for = para

while = enquanto

para 1 até 10 imprima hello

enquanto 10 for menor que 20 imprima hello







# do while

O loop do while é igual ao loop while, só que ao invés de verificar a condição em primeiro lugar ele executa o código dentro do loop e depois verifica a condição.  
Ou seja ele sempre vai executar o que está dentro do loop uma vez.

```
int count = 0;
```

Loop infinito:

```
do {  
    System.out.println("Repetição nr: " + contador);  
} while (count < 2);
```





# do while

Loop com contador:

```
do {  
    System.out.println("Repetição nr: " + contador);  
    count++;  
} while (count < 2);
```



A decorative graphic on the left side of the slide. It features a large cyan hexagon with the number '3' inside. Surrounding this central hexagon are several smaller hexagons of varying shades of blue and cyan. Some of these smaller hexagons contain white icons: a lightbulb, a thumbs-up, a smartphone, a magnifying glass, and a gear. There is also a network-like icon with a central node and several smaller nodes connected by lines.

3

Switch, Case e Default



# Switch, Case e Default

O switch verifica uma variável e age de acordo com os seus casos.  
Exemplo.

```
SWITCH (variável) {  
    CASE valor :  
        Código a ser executado caso o valor de case seja o  
        mesmo da variável de switch.  
        break;  
    CASE valor:  
        Outro código;  
        break;  
}
```





# Referências

