**Comandos de Controle de Fluxo​: Repetição**

* As estruturas de **Controle de Repetição** são utilizadas para repetir um determinado grupo de comandos por diversas vezes.​
* As estruturas de **Repetição** também são conhecidas por **laços** ou **loop**.​
* Se não montadas adequadamente, as estruturas de **Repetição** podem produzir laços infinitos em um programa, ou seja, o programa pode ficar eternamente executando o conjunto de instruções, até que seja retirado da memória de forma forçada.
* As estruturas de **Controle de Repetição** mais comuns são (entre parênteses seu nome em Java):​
  + baseadas em contador: para (for)​
  + baseadas em condição: enquanto (while / do while)
* Quando usar qual?​
  + para (for): quando se sabe o número de vezes que o laço será executado;​
  + enquanto (while): quando não se tem certeza sobre o número de vezes que o laço será executado, podendo inclusive nunca ser executado;​
  + faça-enquanto (do-while): quando não se tem certeza sobre o número de vezes que o laço será executado, mas se sabe que ele tem que ser executado pelo menos uma vez;​

**Comandos de Controle de Fluxo​: Repetição: para faça (for)**

* As estruturas de **repetição com contador** permitem a execução de um grupo de comandos num determinado número conhecido de vezes.
* Exemplos:
* Para 10 carros faça
* lave o carro
* Para 5 bombons faça
* coma o bombom
* para 40 alunos faça
* calcule a média do aluno
* Ler n
* Para n valores faça
* some o valor
* Sintaxe e exemplo do para (**for** em java).
* Em Português Estruturado:​
* para contador de inicio a fim faça
* bloco de comandos
* fim para
* Em Java (formato):​
* for (expressão 1; expressão 2; expressão 3) {​
* [bloco de comandos]​

}

* Em Português Estruturado:​
* potencia = 1
* para contador de 1 a 5 faça
* potencia = potencia \* x
* fim para
* Em Java:​
* int potencia = 1;
* for (int contador = 1; contador <= 5; contador++) {
* potencia \*= x;

}

* Em Java as expressões podem ser:​
  + numéricas (real ou inteira) - ex: for (double x = 10; x > 1; x -= 0.5)​
  + caractere - ex: for (char x = 'a'; x <= 'z'; x++).
* Importante:​
  + **expressão 1**: é a expressão que define o valor de inicialização de uma variável de controle. Esta expressão é uma instrução de atribuição executada uma única vez antes de o laço ser iniciado​
  + **expressão 2**: é a expressão que define a condição de teste da variável de controle. Ela é avaliada como verdadeira ou falsa no início de cada execução do laço. Se a expressão for verdadeira, o bloco de comandos do laço é executado. Se for falsa, o laço é encerrado e o controle passa para a instrução seguinte. Normalmente usa os operadores relacionais: >, <, >=, <=​
  + **expressão 3**: é a expressão que define a forma de atualização da variável de controle a cada vez que o laço for repetido. Ela é executada ao final de cada repetição do laço. A atualização pode ser feita de forma crescente ou decrescente e em qualquer valor

**Introdução a Linguagem de Programação**

Videoaula\_5\_2 (18:12): <https://web.microsoftstream.com/video/a883f9e3-c794-45d4-af8a-5680c26fa7e6>  
Videoaula\_5\_2 (18:12): [link alternativo](https://furb-my.sharepoint.com/:v:/g/personal/dalton_furb_br/EQ4jF9-zHZZDo4auco41Rb0BJAhU3SLF1mVdNKuiQLwsRA?e=RWa2D5)

**Comandos de Controle de Fluxo​: Repetição: enquanto (while / do while)**

* As estruturas de repetição condicionais avaliam uma determinada condição lógica para fazer a execução do laço de repetição. A condição sempre será avaliada como **verdadeira** ou **falsa**.
* A estrutura condicional mais comum é o **enquanto-faça**. A estrutura **enquanto-faça** executa um conjunto de instruções enquanto a expressão lógica for avaliada como verdadeira.​
* A condição pode estar no início ou no fim do bloco de execução.
* Exemplos:
* Enquanto tem carros para lavar faça
* lave o carro
* Enquanto tem bombons para comer faça​
* coma o bombom
* Enquanto não encontrei o livro na biblioteca faça​
* procure em outra estante​
* Faça
* calcule a média do aluno​
* Enquanto tem alunos​

**Comandos de Controle de Fluxo​: Repetição: enquanto (while)**

* Sintaxe e exemplo do comando de controle de repetição **enquanto** com a condição no **início** do bloco.
* Em Português Estruturado:​
* enquanto <expressão-lógica> faça
* <sequência-de-comandos>
* fim enquanto
* Em Java:​
* while (expressão-lógica) {
* <sequência-de-comandos>

}

* Em Português Estruturado:​
* contador = 1
* potencia = x
* enquanto potencia <= 100 faça
* potencia = potencia \* x
* contador = contador + 1
* fim enquanto
* Em Java:​
* int contador = 1;
* int potencia = x;
* while (potencia <= 100) {
* potencia \*= x;
* contador++;

}

**Comandos de Controle de Fluxo​: Repetição: enquanto (do while)**

* Sintaxe e exemplo do comando de controle de repetição **enquanto** com a condição no **fim** do bloco.
* Em Português Estruturado:​
* faça
* <sequência-de-comandos>
* enquanto <expressão-lógica>
* Em Java:​
* do {
* <sequência-de-comandos>

} while (expressão-lógica);

* Em Português Estruturado:​
* contador = 1
* potencia = x
* faça
* potencia = potencia \* x
* contador = contador + 1
* enquanto potencia <= 100
* Em Java:​
* int contador = 1;
* int potencia = x;
* do {
* potencia \*= x;
* contador++;

} while (potencia <= 100);

**Comandos de Controle de Fluxo​: Repetição: enquanto (while / do while)**

* Importante:
  + expressão-lógica: é a expressão avaliada antes de cada repetição do laço. Quando seu resultado for **verdadeiro**, <sequência-de-comandos> é executada​
  + na expressão lógica podem ser usados os operadores lógicos NAO, E ou OU, como uma estrutura de seleção do tipo se, por exemplo​
* Em Java deve-se notar o uso de ponto e vírgula (;) após a expressão lógica **apenas** no do-while.​