02

Compare os tipos de laços

Embora o `for`, `while` e `do .. while` sejam todos estruturas que permitam executar ::loops::, existem similaridades e diferenças entre essas construções que serão cobradas na prova.

Comparando while e do .. while

No caso do `while` e `do while`, ambos são muito similares, sendo a principal diferença o fato do `do .. while` ter a condição testada somente após executar o código de dentro do ::loop:: pelo menos uma vez.

```
int i = 20;

//imprime 20, já que só faz o teste após a execução do código
do {
    System.out.println(i);
    i++;
} while(i < 10);

int j = 20;

//não imprime nada, já que testa antes de executar o bloco
while(j < 10){
    System.out.println(i);
    i++;
}
```

COPIAR CÓDIGO

Comparando for e enhanced for

Apesar de ser mais complexo, o `for` simples é mais poderoso que o `enhanced for`. Com o `enhanced for`, não podemos:

- Percorrer mais de uma coleção ao mesmo tempo;
- Remover os elementos da coleção;
- Inicializar um array.

Caso desejemos fazer uma iteração de **leitura, por todos os elementos da coleção**, aí sim o `enhanced for` é a melhor opção.

Comparando while e for

Ambas estruturas de laço permitem executar as mesmas operações em nosso código, e são bem similares. Mas, apesar disso, existem situações em que o código ficará mais simples caso optemos por uma delas.

Geralmente optamos por usar `for` quando sabemos a quantidade de vezes que queremos que o laço seja executado. Pode ser percorrer todos os elementos de uma coleção, (onde sabemos a quantidade de vezes que o loop será executado por saber o tamanho da coleção) ou simplesmente executar o laço uma quantidade fixa de vezes.

Usamos o `while` ou `do .. while` quando não sabemos a quantidade de vezes em que o laço será executado, mas sabemos uma condição que, enquanto for verdadeira, fará com que o laço seja repetido.

O exemplo a seguir mostra um código no qual conhecemos a condição de parada, mas não faz sentido nenhum ter uma variável inicializada ou uma condição de incremento, então escolhemos um `while`:

```
while(conta.getSaldo() > 0) {
    conta.saca(1000);
}
```

COPIAR CÓDIGO

Note que, caso queira contar quantas vezes foi sacado, faria sentido usar um `for` :

```
int saques;
for(saques = 0; conta.getSaldo() > 0; saques++) {
    conta.saca(1000);
}
System.out.println("Saquei " + saques + " vezes");
```

COPIAR CÓDIGO