

Laços encadeados

Transcrição

Já vimos todos os comandos básicos da sintaxe. Vamos praticar o uso dos laços, com o `if`, para sedimentarmos este conhecimento adquirido no curso! Criaremos uma classe para testarmos **laços encadeados**, aninhados uns aos outros:

`TestaLacos`, com um `main` para imprimirmos de `0` a `10` dez vezes em linhas distintas, com a tabuada de cada número.

Usaremos o `int multiplicador`, começando pela tabuada do `1`, indo à do `10`. Dentro deste laço, queremos fazer outro, com valor diverso, como em um `contador`, também começando do `0` e indo a `10`.

```
public class TestaLacos {  
  
    public static void main(String[] args) {  
        for(int multiplicador = 1; multiplicador <= 10; multiplicador++) {  
            for(int contador = 0; contador <= 10; contador++) {  
                System.out.println(multiplicador * contador);  
            }  
        }  
    }  
}
```

Se pedirmos para que seja impresso `multiplicador * contador`, obteremos algo gigantesco, como verificaremos salvando e rodando o código. Serão impressos os resultados contendo as tabuadas, mas queremos algo um pouco mais organizado. Para isso, em vez de utilizarmos o `System.out.println()`, usaremos `System.out.print()`, seguido de `System.out.print(" ");`, que nos trará os números todos alinhados horizontalmente.

Ainda não é isto que queremos! Queremos um "Enter" a cada tabuada, quer dizer, cada tabuada em uma linha. Vamos, então, incluir outro `System.out.println()` após o escopo do segundo `for`, assim:

```
public class TestaLacos {  
  
    public static void main(String[] args) {  
        for(int multiplicador = 1; multiplicador <= 10; multiplicador++) {  
            for(int contador = 0; contador <= 10; contador++) {  
                System.out.print(multiplicador * contador);  
                System.out.print(" ");  
            }  
            System.out.println();  
        }  
    }  
}
```

Salvando e rodando o código acima, obteremos, como gostaríamos:

```
0 1 2 3 4 5 6 7 8 9 10  
0 2 4 6 8 10 12 14 16 18 20  
0 3 6 9 12 15 18 21 24 27 30
```

```
0 4 8 12 16 20 24 28 32 36 40
0 5 10 15 20 25 30 35 40 45 50
0 6 12 18 24 30 36 42 48 54 60
0 7 14 21 28 35 42 49 56 63 70
0 8 16 24 32 40 48 56 64 72 80
0 9 18 27 36 45 54 63 72 81 90
0 10 20 30 40 50 60 70 80 90 100
```