

Escopo nos laços

Transcrição

Para vermos o laço de forma mais estruturada e desafiadora, faremos uma somatória com os números de 0 a 10, criando a classe `TestaSomatoria`:

```
public class TestaSomatoria {  
  
    public static void main(String[] args) {  
        int contador = 0;  
        while(contador <= 10) {  
            int total = 0;  
            total = total + contador;  
  
            System.out.println(total);  
            contador++;  
        }  
    }  
}
```

Vamos imprimir as somatórias parciais para ver o que está acontecendo?

Queremos que se mostre 0, seguido de 1, e então 2, 3, por causa de 1 + 2, e então 6, de 1 + 2 + 3. No entanto, obteremos:

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

Ué! Não funcionou! Isto porque toda vez que se entra no `while`, é criada uma nova variável `total` por causa do escopo e, ao voltarmos ao próximo laço, quando ocorre a **iteração**, ele zera de novo, pois a velha `total` já deixou de existir.

Falta acertarmos o escopo declarando e inicializando a variável `total` após a linha que contém `contador`:

```
public class TestaSomatoria {  
  
    public static void main(String[] args) {  
        int contador = 0;  
        int total = 0;  
  
        while(contador <= 10) {
```

```
        total = total + contador;

        System.out.println(total);
        contador++;
    }
}
```

Se salvarmos e rodarmos novamente, desta vez veremos o seguinte no Console:

```
0
1
3
6
10
15
21
28
36
45
55
```

Ou seja, as somatórias parciais, incluindo a última, 55, que é o número desejado. Se quisermos apenas este resultado final, poderemos deixar o código assim:

```
public class TestaSomatoria {

    public static void main(String[] args) {
        int contador = 0;
        int total = 0;

        while(contador <= 10) {
            total = total + contador;
            contador++;
        }
        System.out.println(total);
    }
}
```

Salvando e rodando o código novamente, obtém-se a impressão de 55.

É possível deixar este código mais enxuto, porém focaremos em `total = total + contador;`, que já vimos que pode ser escrito assim: `total += contador;`, o qual traz exatamente o mesmo resultado.