



Mão na massa: utilizando o tipo double

Vamos praticar um pouco do uso da variável do tipo **double**

1) Crie uma classe "**TestaPontoFlutuante**" e estruture seu método main.

2) Declare uma variável **salario** com o valor de 1250.70 , dessa forma:

```
double salario = 1250.70;
```

Se **salario** fosse do tipo **int** receberíamos um erro, já que temos casas decimais.

3) Imprima a variável **salario** e verifique se o valor está correto, Nosso código deve ficar assim:

```
public class TestaPontoFlutuante{

    public static void main(String[] args){

        double salario = 1250.70;

        System.out.println(salario);
    }
}
```

4) Utilize o + para concatenar a variável **salario** com um texto:

```
System.out.println("meu salario é " + salario);
```

5) Lembre-se que não podemos colocar um valor com ponto flutuante dentro de uma variável do tipo **int**, teste:

```
public class TestaPontoFlutuante{

    public static void main(String[] args){

        double salario = 1250.70;

        System.out.println(salario);

        int valor = 12.5;           //não compila.

        int valor2 = 0.0;           //não compila.

        double teste = 125.50;      // compila

        int valor3 = teste;         //não compila.

    }
}
```

O Java é bastante tipado, veja que nem com o valor **0.0** nosso código compilará, então tenha atenção com isso!

6) Teste também um cálculo de divisão:

```
int divisao = 5 / 2;  
System.out.println(divisao);
```

Nossa saída será **2**.

É importante ressaltar que quando fazemos uma divisão não exata, mas estamos dividindo números inteiros, o Java sempre **truncará** o resultado. Nesse exemplo a saída será sempre **2**.

Se a variável **divisao** fosse do tipo **double**, teríamos o mesmo problema, a única diferença seria que nossa saída teria casa decimal: **2.0**

7) Para resolver o problema, coloque em um dos valores da divisão com alguma casa decimal, além de mudar o tipo da variável para double:

```
double divisao = 5.0 / 2;  
System.out.println(divisao);
```

Agora sim obteremos a saída esperada: **2.5**