Nome: Djeniffer Letícia Kremer

Disciplina: Laboratório de Programação

Professor: Lucas Fogaça



O **java.lang.StrictMath.sqrt()** é um método embutido da classe StrictMath em Java que é usado para o retornar a raiz quadrada de um valor do tipo double passado a ele como argumento.

Sintaxe: public static double sqrt(*double num*)

System.out.println("Raiz de 8 = " + math.sqrt(8));

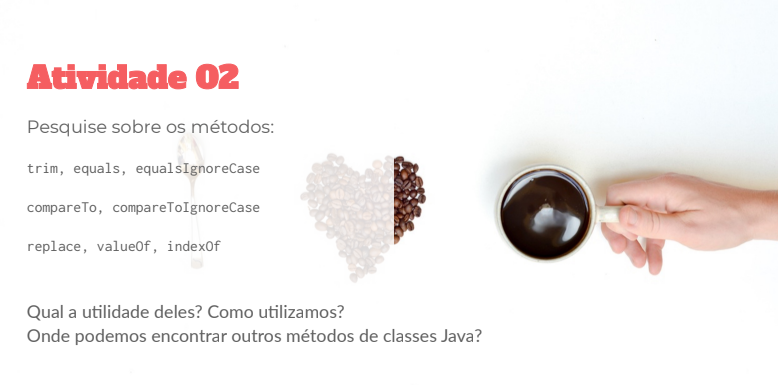
O **java.lang.Math .pow()** é usado para calcular um aumento de número à potência de algum outro número. Esta função aceita dois parâmetros e retorna o valor do primeiro parâmetro elevado ao segundo parâmetro. Existem alguns casos especiais, conforme listado abaixo:

* Se o segundo parâmetro for zero positivo ou negativo, o resultado será 1,0.
* Se o segundo parâmetro for 1,0, o resultado será o mesmo do primeiro parâmetro.
* Se o segundo parâmetro for NaN, o resultado também será NaN.

Sintaxe: Math.pow(base, expoente)

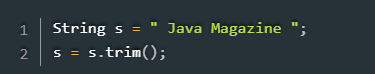
O **java.lang.Math.abs()** retorna o valor absoluto de um determinado argumento.

Para calcular o valor absoluto de qualquer número, temos um método especificado em Java denominado **abs()** presente na **classe java.lang.Math** . O valor absoluto se refere ao valor positivo correspondente ao número passado como nos argumentos. Agora você deve estar se perguntando o que exatamente isso significa, então por isso é referido, não importa qual seja o número positivo ou negativo que foi passado para o cálculo, o cálculo ocorrerá sobre o número positivo correspondente em ambos os casos.

* Se o argumento não for negativo, o argumento será retornado.
* Se o argumento for negativo, a negação do argumento é retornada.
* **Sintaxe:** public static DataType abs(DataType a)

**Trim**

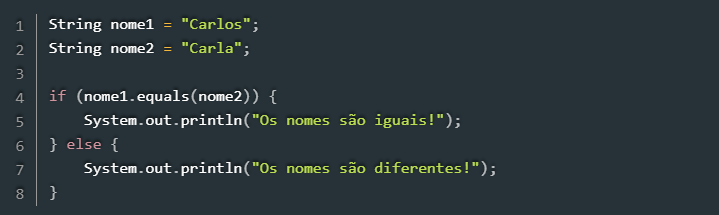
O método trim() retorna uma String sem os espaços em branco no começo e no final da mesma. É útil quando se trabalha com preenchimento de formulários. Antes de salvar no banco de dados as informações preenchidas pelo usuário na tela, pode-se chamar o método trim() em todos os campos de texto preenchidos pelo operador, de modo a evitar que espaços em branco desnecessários sejam gravados, economizando espaço no banco de dados. O código abaixo exemplifica a utilização do método trim():



Após a execução do comando na linha 2, a variável de referência se apontará para a nova String criada sem os espaços que existiam na linha 1 (“Java Magazine”). Todavia, como pode ser observado, o espaço em branco do meio da palavra não foi removido.

**Equals**

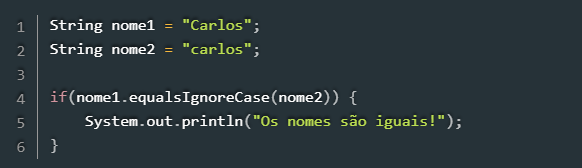
Equals é um método da classe Object utilizado para testar a relação de igualdade entre dois objetos. Esse método está presente em todas as classes, porque todas elas derivam de Object. A forma como essa comparação será feita pode ser determinada por quem está escrevendo a classe e no caso de String, seu autor preparou o método equals para verificar se duas Strings contém exatamente os mesmos caracteres.



Nesse caso a mensagem "Os nomes são diferentes!" será impressa, uma vez que nome1 contém uma sequência de caracteres diferente de nome2.

**equalsIgnoreCase**

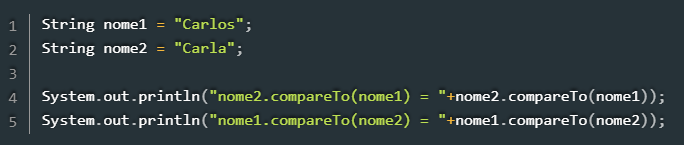
Esse método ignora a distinção entre letras maiúsculas e minúsculas nas duas strings comparadas



Nesse novo exemplo os nomes são iguais, então o método equalsIgnoreCase retornará true

**compareTo**

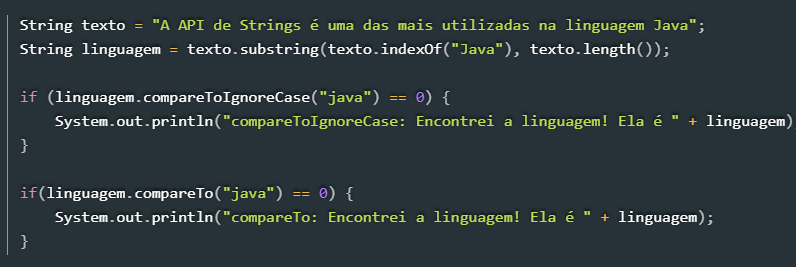
Esse método pode retornar 0 se as strings forem iguais, um número negativo se a string que invoca o compareTo for menor que a string que é passada como um argumento e um número positivo se a string que invoca o compareTo for maior que a string que é passada como argumento.



Neste caso, compareTo vai nos dar um número negativo no primeiro caso, porque Carla é menor que Carlos, é um número positivo no segundo caso porque Carlos é maior que Carla.

**compareToIgnoreCase**

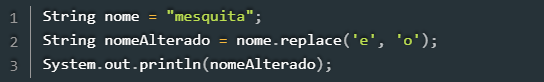
É uma função que compara textos lexigraficamente, ignorando se as letras são maiúsculas ou minúsculas. No exemplo abaixo fazemos duas comparações, uma utilizando compareTo e a outra usando compareToIgnoreCase para analisarmos as diferenças



Após a execução desse código apenas a mensagem "compareToIgnoreCase: Encontrei a linguagem! Ela é Java" será impressa, uma vez que para compareTo os textos "JAVA" e "java" são diferentes.

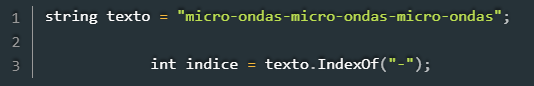
**Replace**

Retorna um novo objeto contendo a string original com um trecho especificado substituído por outra expressão indicada. Esse método deixa a string original inalterada. A versão sobrecarregada do método **replace** permite substituir substrings em vez de caracteres individuais.



**IndexOf**

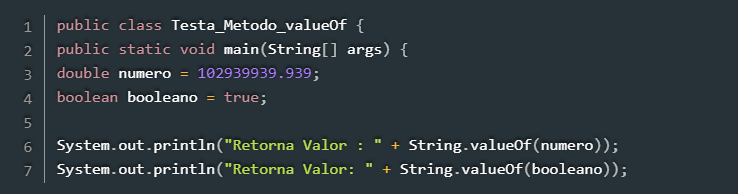
Esse método é simples: serve para sabermos qual índice determinado pedaço da string corresponde. Ele retorna um valor do tipo int.



O valor da variável será cinco. Simples assim.

**valueOf**

valueOf é um método estático da classe String, que não precisa de uma instância para ser invocado. Ele converte um tipo primitivo em um objeto do tipo String.



ONDE PODEMOS ENCONTRAR: Documentação oficial Java.

https://docs.oracle.com/javase/8/docs/api/