



Java Foundations

4-5

A Classe Math

ORACLE
Academy



Copyright © 2020, Oracle e/ou suas empresas afiliadas. Todos os direitos reservados.

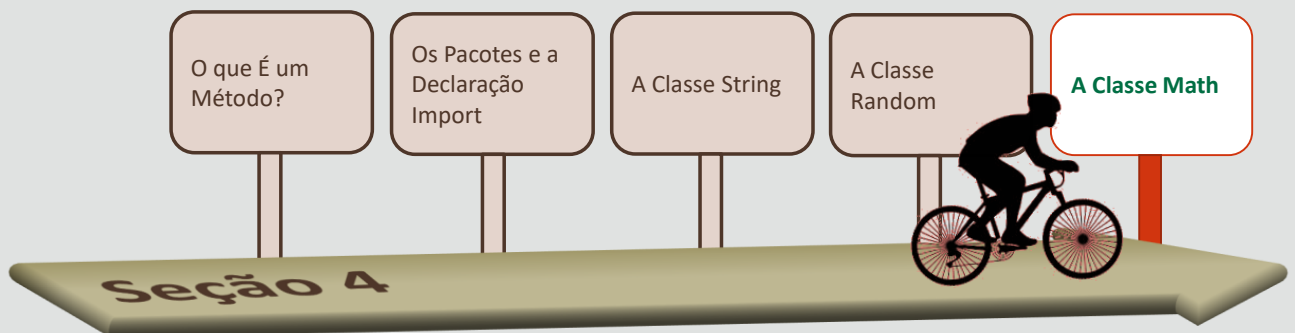
Objetivos

- Esta lição abrange o seguinte objetivo:
 - Entender os métodos da classe Math
 - Usar métodos da classe Math para executar cálculos matemáticos
 - Usar campos da Classe Math



Tópicos

- **Introdução à Classe Math**
- Usando métodos da Classe Math
- Usando campos da Classe Math



ORACLE
Academy

JFo 4-5
A Classe Math

Copyright © 2020, Oracle e/ou suas empresas afiliadas. Todos os direitos reservados.

4

Executando Cálculos Matemáticos

- Ao desenvolver programas, pode ser que você precise fazer cálculos matemáticos mais avançados do que os fornecidos pelos operadores matemáticos básicos do Java
- Por exemplo:
 - Localizando o valor máximo ou mínimo de dois valores
 - Arredondando valores
 - Funções logarítmicas
 - Raiz quadrada
 - Funções trigonométricas
- A classe Java Math contém métodos para executar cálculos matemáticos

A Classe Math

- É uma das muitas classes incluídas nas bibliotecas de classes Java
- Contém métodos que executam várias funções matemáticas
- É parte do pacote `java.lang`

Documentação da Classe Math

- Você pode acessar a documentação aqui:
– <http://docs.oracle.com/javase/8/docs/api/index.html>

Role para ver uma lista de campos e métodos disponíveis nessa classe

Java™ Platform Standard Ed. 7

Overview Package **Class** Use Tree Deprecated Index Help

Prev Class Next Class Frames No Frames

Summary Method Field Class Method Deprecated Field Class Method

java.lang

Class Math

java.lang.Object

java.lang.Math

public final class Math

extends Object

The class `Math` contains methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions.

Unlike some of the numeric methods of class `StrictMath`, all implementations of the equivalent functions of class `Math` are not defined to return the bit-for-bit same results. This relaxation permits better-performing implementations where strict reproducibility is not required.

By default many of the `Math` methods simply call the equivalent method in `StrictMath` for their implementation. Code generators are encouraged to use platform-specific native libraries or microprocessor instructions, where available, to provide higher-performance implementations of `Math` methods. Such higher-performance implementations still must conform to the specification for `Math`.

The quality of implementation specifications concern two properties, accuracy of the returned result and monotonicity of the method. Accuracy of the floating-point `Math` methods is measured in terms of *ulps*, units in the last place. For a given floating-point format, an *ulp* of a specific real number value is the distance between the two floating-point values bracketing that numerical value. When discussing the accuracy of a method as a whole rather than at a specific argument, the number of *ulps* cited is for the worst-case error at any argument. If a method always has an error less than 0.5 *ulps*, the method always returns the floating-point number nearest the exact result; such a method is correctly rounded. A correctly rounded method is generally the best a floating-point approximation can be; however, it is impractical for many floating-point methods to be correctly rounded. Instead, for the `Math` class, a larger error bound of 1 or 2 *ulps* is allowed for certain methods. Informally, with a 1 *ulp* error bound, when the exact result is a representable number, the exact result should be returned as the computed result; otherwise, either of the two floating-point values which bracket the exact result may be returned. For exact results large in magnitude, one of the endpoints of the bracket may be infinite. Besides accuracy at individual arguments, maintaining proper relations between the method at different arguments is also important. Therefore, most methods with more than 0.5 *ulp* errors are required to be *weak-monotonic*: whenever the mathematical function is non-decreasing, so is the floating-point approximation, likewise, whenever the mathematical function is non-increasing, so is the floating-point approximation. Not all approximations that have 1 *ulp* accuracy will automatically meet the monotonicity requirements.

Since:

JDK1.0

Field Summary

Fields	
Modifier and Type	Field and Description
<code>static double</code>	<code>E</code>

ORACLE
Academy

JFo 4-5
A Classe Math

Copyright © 2020, Oracle e/ou suas empresas afiliadas. Todos os direitos reservados.

7

Role o painel esquerdo inferior para baixo e clique no link Math para exibir a documentação no painel principal à direita.

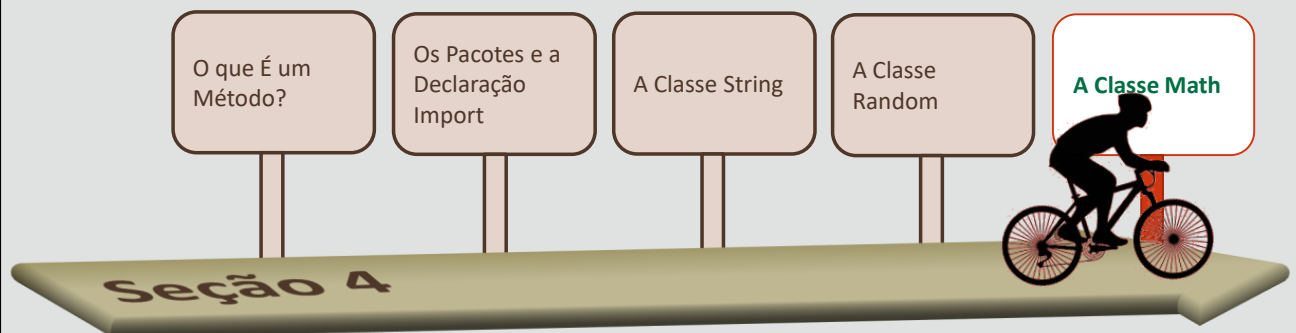


Exercício 1

- Consulte a documentação da classe Math:
 - Standard Edition para Java SE 8:
<http://docs.oracle.com/javase/8/docs/api/>
- Veja se você consegue localizar um valor para PI e um método para calcular a raiz quadrada de um número

Tópicos

- Introdução à Classe Math
- **Usando métodos da Classe Math**
- Usando campos da Classe Math



ORACLE
Academy

JFo 4-5
A Classe Math

Copyright © 2020, Oracle e/ou suas empresas afiliadas. Todos os direitos reservados.

9

Alguns dos Métodos Disponíveis na Classe Math

Nome do Método	Descrição
abs(valor)	valor absoluto
ceil(valor)	arredonda para cima
cos(valor)	cosseno, em radianos
floor(valor)	arredonda para baixo
log(valor)	logaritmo base e
log10(valor)	logaritmo base 10
max(valor1, valor2)	o maior dos dois valores
min(valor1, valor2)	o menor dos dois valores
pow(base, expoente)	base até a potência do expoente
random()	aleatório duplo entre 0 e 1
round(valor)	número inteiro mais próximo
sin(valor)	seno, em radianos
sqrt(valor)	raiz quadrada

O que Há de Diferente em Relação à Classe Math?

- Os métodos da classe Math são métodos estáticos
- Os métodos estáticos podem ser chamados por meio de um nome de classe
- Isso significa que você não precisa criar um objeto da classe Math para chamar os métodos
- Por exemplo, para chamar os métodos da classe Random, você precisa criar um objeto da classe Random como este:

```
Random rndNum = new Random();  
int randomNum = rndNum.nextInt();
```

Como Você Chama os Métodos da Classe Math?

- Você pode chamar métodos da classe Math sem criar uma instância da classe Math, desta forma:

- Sintaxe:

- Math.methodName(parâmetros)

- Exemplo:

- **Math.sqrt**(121.0) ;

Chame métodos prefixando-os com o operador dot de Math

Chamando um Método e Observando Seu Resultado

- Vamos ver um exemplo de como chamar um método e observar seu resultado:

```
public static void main(String[] args) {  
  
    Math.sqrt(121.0);  
} //fim do método main
```

- Observe a saída:
 - Nenhuma saída é exibida
 - Simplesmente chamar esses métodos não produz um resultado visível

Como os Métodos da Classe Math Funcionam?

- Os métodos de Math não imprimem os resultados no console
- Cada método retorna um resultado numérico
- O valor retornado é mais flexível do que a impressão
- Você pode armazená-lo, imprimi-lo ou combiná-lo com uma expressão maior

Armazenando e Imprimindo os Resultados

- Para ver o resultado, você deve imprimi-lo ou armazená-lo em uma variável
- Por exemplo:
 - Imprima o resultado:

```
public static void main(String[] args) { //11.0
    System.out.println("Raiz quadrada: " + Math.sqrt(121.0));
} //fim do método main
```

- Armazene o valor:

```
public static void main(String[] args) {
    double sqroot= Math.sqrt(121.0); //11.0
    System.out.println("Raiz quadrada: " + sqroot);
} //fim do método main
```

Combinando os Resultados

- Você pode combinar os resultados e usá-los em uma expressão maior, como esta:

```
public static void main(String[] args) {  
    double result = Math.min(3, 7) + Math.abs(-50);  
    System.out.println("O resultado é " + result); //53  
}//fim do método main
```




Exercício 2

- No papel, calcule as seguintes instruções Java e anote os resultados:
 - `Math.abs(-1.23)`
 - `Math.pow(3, 2)`
 - `Math.sqrt(121.0) - Math.sqrt(256.0)`
 - `Math.abs(Math.min(-3, -5))`

Exercício 3



- Considere uma variável inteira denominada age
- Use os métodos Math.max e Math.min para responder às seguintes perguntas:
 - Que expressão substituiria idades negativas por 0?
 - Que expressão limitaria a idade máxima a 40?

Resposta:

Que expressão substituiria idades negativas por 0?

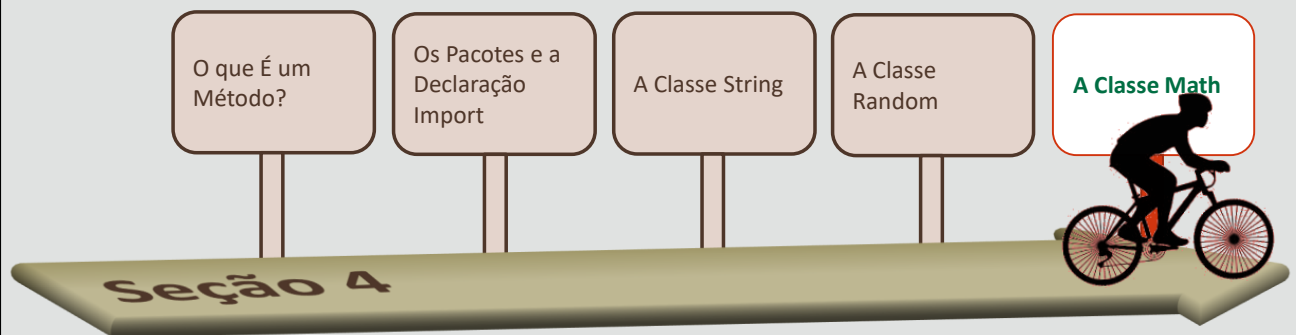
```
Math.max (age, 0)
```

Que expressão limitaria a idade máxima a 40?

```
Math.min (age, 40)
```

Tópicos

- Introdução à Classe Math
- Usando métodos da Classe Math
- **Usando campos da Classe Math**



ORACLE
Academy

JFo 4-5
A Classe Math

Copyright © 2020, Oracle e/ou suas empresas afiliadas. Todos os direitos reservados.

19

Campos na Classe Math

- A classe Math contém dois campos constantes: PI e E

Campo	Descrição
Math.E	2.7182818...
Math.PI	3.1415926...

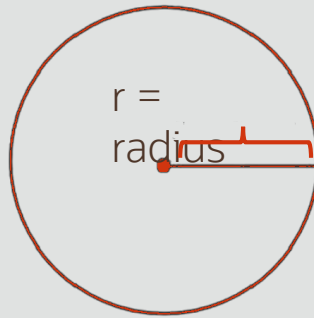


Campo PI

- A classe Math contém uma constante, PI
- Ela contém um valor duplo: 3.14159265358979323846
- Lembre-se de que os métodos da classe Math são métodos estáticos e são acessados usando o nome da classe Math
- Da mesma forma, PI é uma variável estática na classe Math e é acessada usando o nome da classe Math
- Para usar PI em um programa, especifique o nome da classe (Math) e PI, separados pelo operador dot:
 - Math.PI

Calculando a Área de um Círculo

- Suponha que você escreva um programa Java para calcular a área de um círculo
- Esta é a fórmula para calcular a área de um círculo:
 - $\text{Área} = \text{PI} * \text{raio} * \text{raio}$
 - Onde PI é uma constante (aproximadamente 3,1416)



Computando a Área de um Círculo

- Usar o campo Math.PI para calcular a área gera um resultado mais preciso do que usar um valor constante para pi como 3,14

```
public class AreaOfCircle {  
    public static void main(String args[]) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("Insira o raio: ");  
        double radius = sc.nextDouble();  
        double area = Math.PI * radius * radius;  
        System.out.println("A área do círculo é: " + area);  
    } //fim do método main  
} //fim da classe AreaOfCircle
```

Saída:

Insira o raio: 7.5

A área do círculo é: 176.71458676442586

Exercício 4



- O índice de massa corpórea (IMC) é calculado desta forma:

$$BMI = \frac{weight}{height^2} \times 703$$

- Importe e abra o projeto MathEx
- Examine `ComputeBMI.java`
- Escreva um programa que calcule o IMC e arredonde-o



Exercício 4



- Use os métodos da classe Math e exiba a saída como:
 - Informe o peso em libras: 132.5
 - Informe a altura em polegadas: 62.5
 - Seu IMC é 24



Resumo

- Nesta lição, você deverá ter aprendido a:
 - Usar métodos da classe Math para executar cálculos matemáticos
 - Usar campos da Classe Math



