

# Desenvolvimento Aplicado a Desktop

# Classe Random

Gerando números aleatórios em Java com **java.util.Random**

Usando classe **Random**

Usamos a classe Random para gerar números aleatórios em Java. É possível gerar números de vários tipos, incluindo inteiros, double e float.



# Classe Random

```
Rand.java
1 import java.util.Random;
2
3 public class Rand {
4
5     public static void main(String[] args) {
6
7         Random aleatorio = new Random();
8         int valor = aleatorio.nextInt();
9         System.out.println("Número gerado: " + valor);
10
11     }
12
13 }
14
```

```
Console
<terminated> Rand [Java Application] C:\Users\Pedro Albino\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win3
Número gerado: 652586650
```

# Classe Random

Gerando um número inteiro aleatório entre 1 e 30:

```
Rand.java ✕
1 import java.util.Random;
2
3 public class Rand {
4
5     public static void main(String[] args) {
6
7         Random aleatorio = new Random();
8         int valor = aleatorio.nextInt(30)+1;
9         System.out.println("Número gerado: " + valor);
10
11     }
12
13 }
14
```

```
Console ✕
<terminated> Rand [Java Application] C:\Users\Pedro Albino\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full
Número gerado: 12
```

# Classe Random

Gerando um número aleatório do tipo float entre 0 e 100:

```
Rand.java X
1 import java.util.Random;
2
3 public class Rand {
4
5     public static void main(String[] args) {
6
7         Random aleatorio = new Random();
8         float valor = aleatorio.nextFloat()*100;
9         System.out.println("Número gerado: " + valor);
10
11     }
12
13 }
14
```

```
Console X
<terminated> Rand [Java Application] C:\Users\Pedro Albino\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.w
Número gerado: 16.964006
```

# Exceções

Ao executar o código Java, podem ocorrer diferentes erros: erros de codificação cometidos pelo programador, erros devido a uma entrada incorreta ou outros imprevisíveis.

Quando ocorre um erro, o Java normalmente para e gera uma mensagem de erro. O termo técnico para isso é: Java lançará uma exceção (lançará um erro).

# Exceções

A **try** instrução permite definir um bloco de código a ser testado quanto a erros enquanto está sendo executado.

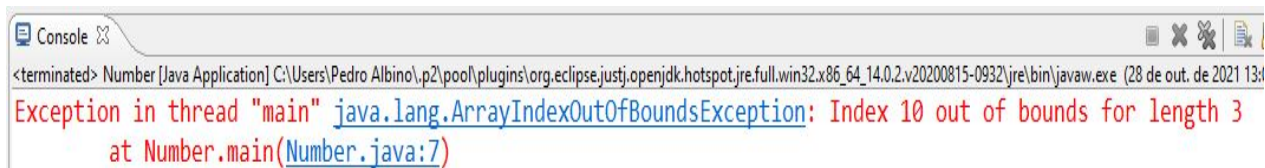
A **catch** instrução permite definir um bloco de código a ser executado, se ocorrer um erro no bloco try.

As palavras **try- catch** chave e vêm em pares:

```
try {  
    // Bloco de código  
}  
catch(Exception e) {  
    // Bloco de código para lidar com erros  
}
```

# Exceções

```
public class Number {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int[] myNumbers = {1, 2, 3};  
        System.out.println(myNumbers[10]);  
    }  
  
}
```



Console

<terminated> Number [Java Application] C:\Users\Pedro Albino\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_14.0.2.v20200815-0932\jre\bin\javaw.exe (28 de out. de 2021 13:00)

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 10 out of bounds for length 3  
at Number.main(Number.java:7)



# Exceções

```
Cath.java
1
2 public class Cath {
3
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         try {
7             int [] meuNumero = {1,2,3};
8             System.out.println(meuNumero[10]);
9         } catch (Exception e) {
10            System.out.println("Tamanho do array não encontrado, reveja novamente");
11        }
12    }
13
14 }
```

```
Console
<terminated> Cath [Java Application] C:\Users\Pedro Albino\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v20200815-0932\jre\bin\javaw.exe (
Tamanho do array não encontrado, reveja novamente|
```

# Exceções

```
Cath.java
1 import java.util.Scanner;
2
3 public class Cath {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner scanner = new Scanner(System.in);
8         try {
9             System.out.println("Digite um valor inteiro");
10            int numero = scanner.nextInt();
11            System.out.println(numero);
12            System.out.println("Digite um valor booleano");
13            boolean texto = scanner.nextBoolean();
14            System.out.println(texto);
15        } catch (Exception e) {
16            System.out.println("Valor errado");
17        }
18    }
19
20 }
21
```

```
Console
<terminated> Cath [Java Application] C:\Users\Pedro Albino\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v...
Digite um valor inteiro
7,8
Valor errado
```

## Jogo Adivinhe os números

Agora é sua vez, inicie um novo projeto e crie você mesmo o programa de contagem de palavras. Siga estas etapas para completar o exercício:

- Gere um número aleatório entre 1-100.
- Crie um loop que peça ao usuário que adivinhe um número.
- Leia a entrada do usuário e compare com o número aleatório.
- Deixe o usuário saber se o número adivinhado foi maior ou menor que o número aleatório.
- Se eles adivinharem o número, feche o loop e diga que ganharam.
- Se eles adivinharem o número, feche o loop e diga que ganharam.
- Se eles gastaram todos os 10 palpites, encerre o loop e diga que perderam.

# Entrada, Saída de Dados e Exceções com Exercícios

1-) Faça um programa para calcular o estoque médio de uma peça, sendo que:  
$$\text{ESTOQUE MÉDIO} = (\text{QUANTIDADE\_MÍNIMA} + \text{QUANTIDADE\_MÁXIMA}) / 2.$$

# Entrada, Saída de Dados e Exceções com Exercícios

1-) Faça um programa para calcular o estoque médio de uma peça, sendo que:  
$$\text{ESTOQUE MÉDIO} = (\text{QUANTIDADE\_MÍNIMA} + \text{QUANTIDADE\_MÁXIMA}) / 2.$$

## Entrada, Saída de Dados e Exceções com Exercícios

2) Efetuar o cálculo da quantidade de litros de combustível gasta em uma viagem, utilizando um automóvel que faz 12 Km por litro. Para obter o cálculo, o usuário deve fornecer o tempo gasto na viagem e a velocidade média.

Desta forma, será possível obter a distância percorrida com a fórmula  **$\text{DISTANCIA} = \text{TEMPO} * \text{VELOCIDADE}$** .

Tendo o valor da distância, basta calcular a quantidade de litros de combustível utilizada na viagem com a fórmula:  **$\text{LITROS\_USADOS} = \text{DISTANCIA} / 12$** . O programa deve apresentar os valores da velocidade média, tempo gasto, a distância percorrida e a quantidade de litros utilizada na viagem. Dica: trabalhe com valores reais.

## Entrada, Saída de Dados e Exceções com Exercícios

3) Ler uma temperatura em graus Celsius e apresentá-la convertida em graus Fahrenheit. A fórmula de conversão de temperatura a ser utilizada é  $F = (9 * C + 160) / 5$ , em que a variável F representa a temperatura em graus Fahrenheit e a variável C representa a temperatura em graus Celsius.

## Entrada, Saída de Dados e Exceções com Exercícios

4) Ler uma temperatura em graus Fahrenheit e apresentá-la convertida em graus Celsius. A fórmula de conversão de temperatura a ser utilizada é  $C = (F - 32) * 5 / 9$ , em que a variável F é a temperatura em graus Fahrenheit e a variável C é a temperatura em graus Celsius.



## Entrada, Saída de Dados e Exceções com Exercícios

5) Faça um algoritmo que leia o nome, o sexo e o estado civil de uma pessoa. Caso sexo seja “F” e estado civil seja “CASADA”, solicitar o tempo de casada (anos).

## Entrada, Saída de Dados e Exceções com Exercícios

6) Faça um algoritmo que leia uma variável do tipo inteiro e some 5 caso seja par ou some 8 caso seja ímpar, imprimir o resultado desta operação.

## Entrada, Saída de Dados e Exceções com Exercícios

7) Percorra todos os números de 1 até 100. Para os números ímpares, exiba no console um “\*”, e para os números pares, dois “\*\*”.