Classe Scanner: Leitura de dados via teclado
Para utilizar a classe Scanner em uma aplicação Java deve-se utilizar o seguinte procedimento:

1. importar o pacote java.util:

```
import java.util.Scanner;
```

2. Instanciar e criar um objeto Scanner:

```
Scanner ler = new Scanner(System.in);
```

3. Lendo um valor inteiro:

```
int n;
System.out.printf("\nInforme um número inteiro: ");
n = ler.nextInt();
```

4. Lendo um valor real:

```
float preco;
System.out.printf("\nInforme o preço da mercadoria: R$ ");
preco = ler.nextFloat();

double salario;
System.out.printf("\nInforme o salário do Funcionário: R$ ");
salario = ler.nextDouble();
```

5. Lendo uma String, usado na leitura de palavras simples que não usam o caractere de espaço (ou barra de espaço):

```
String s;
System.out.printf("\nInforme uma palavra simples: ");
s = ler.next();
```

6. Lendo uma String, usado na leitura de palavras compostas, por exemplo, Porto Alegre:

```
String s;
System.out.printf("\nInforme uma palavra composta: ");
s = ler.nextLine();
```

7. Lendo um caractere usando o método read() do pacote de classes System.in:

```
public static void main(String args[]) throws Exception {
      char c;
      System.out.printf("\nInforme um Caractere: ");
      c = (char)System.in.read();
 }

ou

 public static void main(String[] args)
 {
        Scanner sc = new Scanner(System.in);

        System.out.printf("\nInforme um Caractere: ");
        char c = sc.next().charAt(0);
        System.out.println("c = " + c);
```

```
       }
```

8. Lendo um boolean:

```
import java.util.*;

public class ExemploLeituraBoolean {
    public static void main(String[] args) {
            System.out.print("Você tem 18 anos ou mais? ");
            Scanner sc = new Scanner(System.in);
            boolean maiorIdade = sc.nextBoolean();
            if (maiorIdade == true) {
                System.out.println("Maior de Idade!");
            } else if (maiorIdade == false) {
                System.out.println("Menor de Idade!");
            }
            sc.close();
     }
}
```

9. Importante! Na leitura consecutiva de valores numéricos e String deve-se esvaziar o buffer do teclado antes da leitura do valor String, por exemplo:

```
       Scanner leia = new Scanner(System.in);
      int num;
      String s;
      System.out.printf("\nInforme um Número Inteiro: ");
      num = leia.nextInt();
      leia.nextLine(); // esvazia o buffer do teclado

      System.out.printf("\nInforme uma palavra: ");
      s = leia.nextLine();
```

Verificar conteúdo de Strings

```
 public void validarPreenchimento(String palavra)
 {
        //1 - Verificar se a String é nula
        if (palavra == null) {
           System.out.println("Erro de Preenchimento de String! A String
está NULA!");
        }

        //2- Verificar se a String está vazia
        if (palavra.isEmpty()) {
            System.out.println("Erro de Preenchimento de String! A String
está VAZIA!");
        }

        //3 - Verificar se a String possui algum texto específico SEM
ignorar diferenças entre maiúscula e minúscula
        if (palavra.equals("Teste")) {
```

```
            System.out.println("Strings Iguais!");
        }

        //4 - Verificar se a String possui algum texto específico
IGNORANDO diferenças entre maiúscula e minúscula
        if (palavra.equalsIgnoreCase("Teste")) {
            System.out.println("Strings Iguais!");
        }
}
```

Funcionamento do Math. random( )
A Definição oficial diz: "Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0. " Ou seja, retorna um ponto flutuante, sendo esse um número pseudo-aleatório no intervalo [0, 1), isto é, de 0 (inclusive) até 1, mas não incluindo 1 (exclusivo). Em outras palavras, gera um número aleatório entre 0 até 0.9999999.

Dessa forma, a chamada do método Math.random() geraria a possível sequência de valores aleatórios:

0.5824683890332182
0.15599339785628574
0.19135110741912686
0.4996250046384343
0.0008728340690463687

Para gerar números aleatórios inteiros, devemos multiplicar o método por um inteiro qualquer. Por exemplo:

Math.random() * 10
Em seguida, devemos utilizar o casting (int) para pegar apenas a parte inteira da multiplicação:

((int)(Math.random() * 10))
O código acima irá gerar números aleatórios no seguinte intervalo 0, 1, 2, 3, 4, 5, 6, 7, 8, e 9. INCLUINDO O 0, MAS EXCLUINDO O 10.

Caso quiséssemos gerar um número entre 1 e 10, incluindo 1 e incluindo o 10. Deveríamos utilizar o seguinte código:

(((int)(Math.random() * 10)) + 1)
No código acima, estamos multiplicando Math.random() por 10 e transformando o valor double gerado em int e, somente depois, somamos com o valor 1, para garantir que sejam gerados números incluindo o 1 e incluindo o 10. Exemplificando:

se o método Math.random() sorteasse o valor limite mínimo, 0.0, esse valor seria multiplicado por 10, que seria igual a 0.0, transformando em int ficaria apenas 0 e, após somando com 1 seria igual a 1 (limite mínimo do intervalo que se quer gerar).
já se o método Math.random() sorteasse o valor limite máximo, 0.99, esse valor seria multiplicado por 10, que seria igual a 9.9, transformando em int ficaria apenas 9 e, após somando com 1 seria igual a 10 (limite máximo do intervalo que se quer gerar).

Podemos transformar nosso código em uma função onde recebemos como parâmetro o valor máximo aleatório que será incluído na geração, pois temos a soma de 1 ao final que garante isso.

```
public int getRandom(int max) {
   return ( ((int)(Math.random() * max )) + 1 )
}
```

Abaixo seguem vários exemplos com a geração de aleatórios dos mais variados padrões:

```
Gerar Número entre 0 (inclusive) e 1 (exclusivo)
public int getRandom() {
    return ( (int) Math.random() );
}
Gerar Número entre dois valores,  incluindo o mínimo e excluindo o máximo
public int getRandom(int min, int max) {
    return ( ((int) (Math.random() * (max - min))) + min );
}
Gerar Número entre dois valores,  incluindo o mínimo e incluindo o máximo
public int getRandom(int min, int max) {
    return ( ((int) (Math.random() * (max - min + 1))) + min );
}
-> Diferença entre Math.random() e import java.util.Random
```

As instâncias da classe Random são objetos geradores de números aleatórios sem escopo de intervalo como o Math.random().

```
import java.util.Random;

public class Random1
{
    public static void main(String[] args)
   {
        //instância um objeto da classe Random usando o construtor padrão
        Random gerador = new Random();

        //imprime um número inteiro aleatório
        System.out.println(gerador.nextInt());
        //imprime um número inteiro aleatório entre 0 e 25
        System.out.println(gerador.nextInt(26));
    }
}
```

A sequência de números retornada será sempre diferente a cada execução do programa, pois utilizamos o construtor default (sem parâmetros) da classe "Random". As sequências são diferentes, pois quando se usa o construtor default, o Java escolhe "por conta própria" uma semente diferente a cada execução. Mas também é possível gerar sequências fixas, para isso é preciso fornecer a sua própria semente para o construtor:

```
import java.util.Random;
```

```
public class Random3
{
    public static void main(String[] args)
   {
        //instância um objeto da classe Random especificando a semente
        Random gerador = new Random(19700621);

        //imprime sequência de 10 números inteiros aleatórios entre 0 e
25
        for (int i = 0; i < 10; i++) {
            System.out.println(gerador.nextInt(26));
         }
    }
}
```

A classe "Random" também fornece métodos para a geração de números reais: nextDouble() e nextFloat(). Estes métodos não aceitam parâmetros, sempre gerando números entre 0 e 1.

```
import java.util.Random;

public class RandomReal
{
    public static void main(String[] args)
    {
        Random r = new Random();

        System.out.println(r.nextDouble());
        System.out.println(r.nextFloat());
    }
}
```