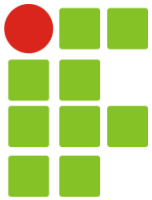


INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE
CAMPUS JOÃO CÂMARA

PROGRAMAÇÃO ESTRUTURADA E ORIENTADA A OBJETOS - STRINGS

Nickerson Fonseca Ferreira
nickerson.ferreira@ifrn.edu.br

Introdução

- **String** é uma classe Java que é responsável por armazenar uma sequência de caracteres tratada como uma única unidade.
- Pode incluir letras, dígitos, caracteres especiais (+, -, *, /, etc.).
- Uma String são escritas como uma sequência de caracteres entre aspas duplas.
  - "Nickerson Fonseca Ferreira" (nome)
  - "Rua ABC, nº 123, João Pessoa" (endereço)
  - "(83) 99999-9899" (telefone)

Introdução

- String **não** é um tipo primitivo, porém o caractere é!
  - char letra = ‘a’;
- Para criarmos uma nova String seguiremos a mesma regra para declaração de variáveis.
- String nome = “Nickerson Ferreira”;

ou

- String cor = new String(“Azul”);
- Lembram dos Vetores ?? A String vai ser utilizada de forma semelhante. Possui índices e uma série de métodos para manipular seu conteúdo.

Manipulando Strings

- Podemos unir o conteúdo de duas ou mais Strings. Para isso, utilizamos o operador +.

```
String nome = "Nickerson";
String sobrenome = "Ferreira";
String nomeComp = nome + "  " + sobrenome;
```

- Ou podemos utilizar o método concat, que realiza a mesma função.

```
String nomeComp = "";
nomeComp = nomeComp.concat(nome);
nomeComp = nomeComp.concat("  ");
nomeComp = nomeComp.concat(sobrenome);
```

Strings e seus métodos

- Por ser uma classe Java, possui uma série de métodos que podemos utilizar.
  - charAt(int indice), retorna um char;
  - length(), retorna um inteiro;
  - equals(String s2), retorna um boolean;
  - equalsIgnoreCase(String s2), retorna um boolean;
  - startsWith(String inicio), retorna boolean;
  - endsWith(String fim), retorna boolean;
  - indexOf(String texto), retorna um inteiro;
  - lastIndexOf(String texto), retorna um inteiro;

Strings e seus métodos

- substring(int indice, int qtd), retorna uma String;
- replace(String orig, String subst), retorna uma String;
- trim(), retorna uma String;
- toUpperCase(), retorna uma String;
- toLowerCase(), retorna uma String;
- split(String divisor), retorna um String[ ];

Utilizando os métodos

- charAt(int indice): retorna o "char" que está presente no índice informado.

```
String nome = "Nickerson";
System.out.println("O char é: " + nome.charAt(3));
```

```
run:
O char é: k
CONSTRUÍDO COM SUCESSO (tempo total: 1 segundo)
```

- length(): retorna um inteiro indicando a quantidade de caracteres da String.

```
run:
O tamanho é: 9
```

```
String nome = "Nickerson";
System.out.println("O tamanho é: " + nome.length());
```

Utilizando os métodos

- equals(String texto): retorna um boolean (valor lógico) com o resultado da comparação.

```
String nome = "Nickerson";
String sobrenome = "Ferreira";

System.out.println("O nome é igual: " + nome.equals("José"));

if (sobrenome.equals("Ferreira")) {
    System.out.println("Fala primo!");
}
```

```
run:
O nome é igual: false
Fala primo!
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- equalsIgnoreCase(String texto): retorna um boolean (valor lógico) com o resultado da comparação. Ignora maiúsculas e minúsculas.

```
String nome = "Nickerson";
String sobrenome = "Ferreira";

System.out.println("O nome é igual: " + nome.equals("nickerson"));
System.out.println("O nome é igual: " + nome.equalsIgnoreCase("nickerson"));
```

```
run:
O nome é igual: false
O nome é igual: true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- startsWith(String texto): retorna um boolean (valor lógico) – true se a String iniciar com o conteúdo de **texto**.

```
String nome = "Nickerson";
String sobrenome = "Ferreira";

System.out.println("O nome inicia com Nick: " + nome.startsWith("Nick"));
System.out.println("O nome inicia com Jos: " + nome.startsWith("Jos"));
```

```
run:
O nome inicia com Nick: true
O nome inicia com Jos: false
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- endsWith(String texto): retorna um boolean (valor lógico) – true se a String finalizar com o conteúdo de **texto**.

```
String nome = "Nickerson";
String sobrenome = "Ferreira";

System.out.println("O nome finaliza com son: " + nome.endsWith("son"));
System.out.println("O nome finaliza com Jos: " + nome.endsWith("Jos"));
```

```
run:
O nome finaliza com son: true
O nome finaliza com Jos: false
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- indexOf(String texto): retorna um inteiro com o número do índice que o conteúdo de **texto** esta na String.

```
String nomeComp = "Nickerson Fonseca Ferreira";

System.out.println("O índice que Ferreira está: " + nomeComp.indexOf("Ferreira"));
System.out.println("O índice que Fonseca está: " + nomeComp.indexOf("Fonseca"));
```

```
run:
O índice que Ferreira está: 18
O índice que Fonseca está: 10
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- lastIndexOf(String texto): retorna um inteiro com o número do último índice que o conteúdo de **texto** esta na String.

```
String nomeComp = "Nickerson Fonseca Ferreira, professor do IFRN. "
        + "Nickerson, nasceu em João Pessoa...";

System.out.println("O último índice que Nickerson está: "
        + nomeComp.lastIndexOf("Nickerson"));
```

```
run:
O último índice que Nickerson está: 47
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- substring(int inicio, int qtd): retorna uma String (parte da original) com o conteúdo a partir do índice de inicio até a quantidade de caracteres.

```
String nomeComp = "NickersonFonsecaFerreira, professor do IFRN. "
        + "Nickerson, nasceu em João Pessoa...";

System.out.println("O trecho selecionado foi: "
        + nomeComp.substring(0, 9));

System.out.println("O trecho selecionado foi: "
        + nomeComp.substring(45));
```

```
run:
O trecho selecionado foi: Nickerson
O trecho selecionado foi: Nickerson, nasceu em João Pessoa...
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- replace(String texto, String subst): retorna uma String substituindo o conteúdo presente em **texto** pelo conteúdo de **subst**.

```
String nomeComp = "#Nickerson#Fonseca#Ferreira#";

System.out.println("O trecho substituído foi: "
        + nomeComp.replace("#", "_"));
```

```
run:
O trecho substituído foi: _Nickerson_Fonseca_Ferreira_
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- trim(): retorna uma String removendo todos os espaços em branco antes e depois da String original.

```
String nomeComp = "      Nickerson Fonseca Ferreira          ";

System.out.println("Tamanho da String original: "
        + nomeComp.length());

System.out.println("A String removendo os espaços: "
        + nomeComp.trim());

System.out.println("Tamanho da String sem os espaços: "
        + nomeComp.trim().length());
```

```
run:
Tamanho da String original: 42
A String removendo os espaços: Nickerson Fonseca Ferreira
Tamanho da String sem os espaços: 26
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

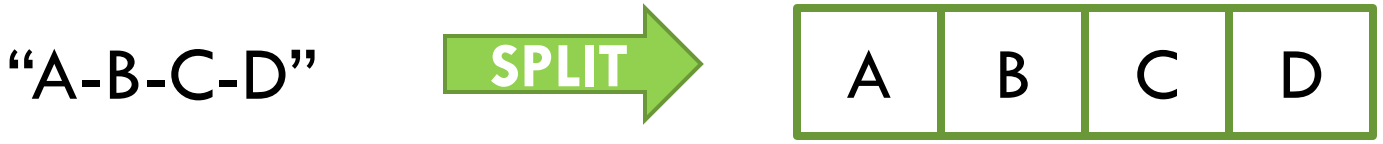
- toUpperCase(): retorna uma String alterando todos os caracteres para **MAIÚSCULOS**.
- toLowerCase(): retorna uma String alterando todos os caracteres para **MINÚSCULOS**.

```
String nomeComp = "Nickerson Fonseca Ferreira";

System.out.println("Tudo MAIÚSCULO: "
        + nomeComp.toUpperCase());

System.out.println("Tudo minúsculo: "
        + nomeComp.toLowerCase());
```

```
run:
Tudo MAIÚSCULO: NICKERSON FONSECA FERREIRA
Tudo minúsculo: nickerson fonseca ferreira
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Utilizando os métodos

- split(String marcador): retorna um Vetor de Strings (String[]) separando cada String entre os marcadores encontrados na String original.



Utilizando os métodos


```
String nomeComp = "Nickerson#Fonseca#Ferreira#IFRN#Professor";
System.out.println("Tamanho do Vetor: "
        + nomeComp.split("#").length);

String[] partes = nomeComp.split("#");
String vet = "Vetor: {";

for (String s : partes) {
    vet += s + ", ";
}

System.out.println(vet.substring(0, vet.length()-2) + "}");
```

```
run:
Tamanho do Vetor: 5
Vetor: {Nickerson, Fonseca, Ferreira, IFRN, Professor}
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Exercícios

- Faça um programa que, a partir de uma string digitada pelo usuário, imprima:

a) O número de caracteres da string.

b) A string com todas suas letras em maiúsculo.

c) O número de vogais da string.

d) Se a string digitada começa com “UNI” (ignorando maiúsculas/minúsculas).

e) Se a string digitada termina com “RIO” (ignorando maiúsculas/minúsculas).

f) O número de dígitos (0 a 9) da string.

g) Se a string é um palíndromo ou não.