PROGRAMAÇÃO EM JAVA

MÓDULO 3

APROFUNDANDO EM *STRINGS*

Tabela 1 - Métodos da classe String

Verificando o **comprimento** do conteúdo de uma variável do tipo *string*:	Para verificar o comprimento de uma variável do tipo *string* usa-se o método "length()" Exemplo: String Nome = "Luiz"; int tamanho = Nome.**length**(); System.out.println("O tamanho da variável Nome é " + tamanho);

Verificando se os conteúdos de duas variáveis do tipo *string* são **IGUAIS** (maneira 1)
 	
IMPORTANTE:
 	
Letras maiúsculas e minúsculas iguais são consideradas pelo Java como letras diferentes! Por isso "Luiz" é diferente de "luiz".
	Outra forma de verificar se os conteúdos de duas variáveis do tipo *string* são iguais é utilizando o método "equals".
	
	Exemplo:
	
	...
	
	String Nome_Cadastrado= "Luiz";
	
	String Nome_Digitado;
	
	...
	
	<Insere-se aqui o código que permite que o usuário digite um nome e armazena na variável "Nome_Digitado">
	
	...
	
	if (Nome_Cadastrado.**equals**(Nome_Digitado))
	
	System.out.println("Os nomes são iguais! ");
	
	else
	
	System.out.println("Os nomes são diferentes! ");
	
	...

Verificando se os conteúdos de duas variáveis do tipo *string* são **IGUAIS** (maneira 2)
 	
IMPORTANTE:
 	
Com esse método a verificação entre maiúsculas e minúsculas não é realizada. Por isso, neste caso, "Luiz" é igual a "luiz".
	Outra forma de verificar se os conteúdos de duas variáveis do tipo *string* são iguais, porém sem se preocupar se alguma letra de quaisquer das variáveis está em maiúscula ou minúscula é utilizando o método "equalsIgnoreCase".
	
	Exemplo:
	
	...
	String Nome_Cadastrado1 = "Luiz";
	String Nome_Cadastrado2 = "LUIZ";
	if (Nome_Cadastrado1.**equalsIgnoreCase**(Nome_Cadastrado))
	System.out.println("Os nomes são iguais! ");
	else
	System.out.println("Os nomes são diferentes! ");
	...

Concatenando (unindo) *string*	Para unir (concatenar duas ou mais *strings* podemos utilizar o operador de adição "+", como já fizemos em alguns desafios. Porém, outra maneira de fazer é utilizando o método **concat()**. Exemplo: Nome = "Luiz "; String Sobrenome = "Corcini" String Nome_Completo1 = Nome + Sobrenome; String Nome_Completo2 = Nome.**concat**(Sobrenome); System.out.println(Nome_completo1); System.out.println(Nome_completo2); Para o exemplo acima, o conteúdo das variáveis Nome_Completo1 e Nome_Completo2 são iguais. OBS: Note que tem um espaço (propositalmente) no final do conteúdo da variável Nome. Caso você esqueça de deixar este espaço em branco em "Luiz ", o conteúdo das variáveis Nome_Completo1 e Nome_Completo2 será "LuizCorcini" e não "Luiz Corcini"

Identificando um caractere de um *string* em uma determinada posição:	Caso você precise saber o valor de um caractere dentro de uma *string* que esteja em um lugar específico, pode utilizar o método **charAt()**.
	
	Exemplo:
	
	Nome = “Luiz “;
	
	System.out.println(“A primeira letra do nome é : + Nome.charAt(0));
	
	for (int i = 0; i < Nome.length() ; i++)
	
	{
	
	System.out.println(Nome.chatAt(i));
	
	}
	
	Executando este código teremos que a frase “A primeira letra do nome é L”
	
	Depois vai aparecer letra por letra da variável nome, uma em baixo da outra, da seguinte forma:
	
	L
	
	u
	
	i
	
	z

Método *substring*	Para conseguir capturar uma parte da *string*, utilizando o método ***substring***
	
	Exemplo:
	
	String Nome_Curso = “Curso de Java “;
	
	System.out.println(Nome_Curso);
	
	//primeira maneira
	
	String subTexto1 = Nome_Curso.substring(9);
	
	System.out.println(subTexto1);
	
	//segunda maneira
	
	String subTexto2 = Nome_Curso.substring(0,5);
	
	System.out.println(subTexto2);
	
	Na primeira maneira, a resposta apresentada é “Java”, pois o compilador começa a ler a partir da posição 9 da String Nome_Curso;
	
	Na segunda maneira, a resposta apresentada é: Curso”, pois o compilador começa a ler na posição ZERO e tem comprimento de cinco caracteres.

Método *replace*	Esse método altera letras dentro de uma *string* Exemplo: String Nome = “Luiz“; System.out.println(“O nome original é: “ + Nome); String Nome_Alterado; Nome_Alterado = Nome.replace(‘z’, ‘s’); System.out.println(“Primeira alteração foi : “ + Nome_Alterado); Nome_Alterado = Nome_Alterado.replace(‘i’, ‘í’); System.out.println(“Segunda alteração foi : “ + Nome_Alterado); Na primeira alteração, a resposta apresentada é “Luis”; Na segunda maneira, a resposta apresentada é: Luís”.
Método trim()	Muitas vezes um usuário distraído insere muitos espaços em branco antes ou depois de um determinado *string*. Isso acontece, por exemplo, quando pedimos para ele escrever o nome da cidade em que ele nasceu. Ele pode escrever, por exemplo “Curitiba”, sem espaços à direita ou à esquerda. Mas, por um erro qualquer, ele pode escrever “ Curitiba “. Note que, no segundo caso, temos um problema por que o nome da Cidade não possui verdadeiramente os espaços em branco a esquerda ou à direita do nome. Para evitar e corrigir este tipo de erro (muito comum) utiliza-se o método **trim()** da seguinte forma: String Cidade = “ Curitiba”; System.out.println(“Nome Original: “ + Cidade); Cidade = Cidade.trim(); System.out.println(“Nome Corrigido: “ + Cidade);