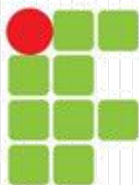


Algoritmos (11006009)

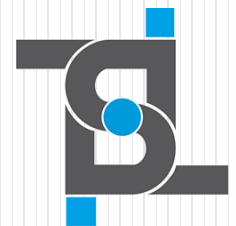
Aula 4 – Operadores, Expressões e Funções

Bruno B. Boniati

bruno.boniati@iffarroupilha.edu.br



**INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
FARROUPILHA
Câmpus Frederico Westphalen



**Sistemas para
Internet**

O que podemos fazer com variáveis?

- Armazenar dados de entrada para posterior processamento;
- Armazenar o resultado do processamento;


- Para representar as operações que podem ser realizadas com variáveis é preciso conhecer o conceito de **operador**:

- Símbolos que permitem a realização de operações sobre variáveis que podem modificar seu valor.

- Ex.

```
var salario = 1006.88 * 1.5
```

```
var salario = 1006.88 * 1.5
```



No exemplo acima a variável “salario” está recebendo o resultado da multiplicação do valor 1006.88 pelo valor 1.5

Observe que o sinal de = (igual) foi utilizado como operador de atribuição (atribuindo o resultado da expressão à esquerda para a variável à direita)

Para construir a expressão foi utilizado o operador * (asterisco) o qual representa a operação de multiplicação.

Categorias dos operadores

- **Aritméticos:** computam expressões matemáticas
- **Relacionais:** realizam comparações de valores de mesmo tipo
- **Lógicos:** permitem combinar resultados de comparações relacionais
- **Concatenação:** atua sobre dados do tipo *string* (juntando seu conteúdo)

Operadores aritméticos

- Computam expressões matemáticas sobre valores numéricos.
- Ex.
 - **var** a = 10; **var** b = 3;

Operador	Significado	Exemplo	Resultado
+	Adição	var c = a + b	c = 13
-	Subtração	var c = a - b	c = 7
*	Multiplicação	var c = a * b	c = 30
/	Divisão	var c = a / b	c = 3.3333333333333333
%	Módulo (resto da divisão)	var c = a % b	c = 1

Atribuição por ...

- Permitem simplificar a operação de atribuição usando a variável à esquerda como destino.
- Ex.
 - **var** a = 10; **var** b = 3;

Operador	Significado	Exemplo	Resultado
+=	Atribuição por Adição	a += b	a = 13
-=	Atribuição por Subtração	a -= b	a = 7
*=	Atribuição por Multiplicação	a *= b	a = 30
/=	Atribuição por Divisão	a /= b	a = 3.333333333333333
%=	Atribuição por Módulo	a %= b	a = 1

Incremento & Decremento

- Simplificam a operação de incrementar ou decrementar o valor de uma variável numérica
- Ex.

```
var a = 10
```

```
var b = 3
```

```
a++ //o valor armazenado em a é 11  
    //é o mesmo que fazer a = a + 1
```

```
b-- //o valor armazenado em b é 2  
    //é o mesmo que fazer b = b - 1
```

Incremento & Decremento

Pré & Pós-fixado


- Qual a diferença do código a seguir
- Ex.

```
var a = 10
```

```
a++  Incremento pós-fixado
```

```
alert(a)
```

```
var a = 10
```

```
++a  Incremento pré-fixado
```

```
alert(a)
```

Percebeu alguma diferença?

Incremento & Decremento

Pré & Pós-fixado

- Experimente agora ...

```
var b = 10
```

```
a = b++
```

```
alert(a)
```

```
alert(b)
```

Isso é o mesmo que fazer:

```
a = b
```

```
b++
```

```
var b = 10
```

```
a = ++b
```

```
alert(a)
```

```
alert(b)
```

Isso é o mesmo que fazer:

```
++b
```

```
a = b
```

Agora deu diferença, né?

Operadores relacionais

- Comparam valores e produzem resultados lógicos (verdadeiro ou falso).
- Ex.
 - **var** a = 10; **var** b = 3;

Operador	Significado	Exemplo	Resultado
==	Igualdade	var c = a == b	c = false
!=	Diferença	var c = a != b	c = true
>	Maior que	var c = a > b	c = true
<	Menor que	var c = a < b	c = false
>=	Maior ou Igual	var c = a >= b	c = true
<=	Menor ou Igual	var c = a <= b	c = false

Operadores lógicos

- Combinam resultados ou expressões lógicas produzindo novos valores lógicos
- Ex.

- **var** a = 10 > 5; **var** b = 3 != (4-1);

Operador	Significado	Exemplo	Resultado
!	Negação	var c = !a	c = false
&&	“E” lógico (conjunção)	var c = a && b	c = false
	“Ou” lógico (disjunção)	var c = a b	c = true

Tabelas verdade dos operadores lógicos

- Negação

A	!A
true	false
false	true

- “E”
(conjunção)

A	B	A && B
true	true	true
true	false	false
false	true	false
false	false	false

- “OU”
(disjunção)

A	B	A B
true	true	true
true	false	true
false	true	true
false	false	false

Expressões

- Parte da tarefa de programar consiste em produzir expressões, ou seja, instruções formadas por variáveis, constantes, operadores, etc.
- Vamos a um exemplo:
 - Como poderíamos representar a seguinte fórmula em um programa de computador?

- $$A = \frac{\pi r^2}{b+12} \times 3 + 5 \div b$$

Linearização de Expressões

- Atividade que consiste na decomposição das partes de uma expressão matemática em uma única linha, utilizando-se das regras de precedência dos operadores ou parêntesis para resolver eventuais ambiguidades..

- A expressão...

- $A = \frac{\pi r^2}{b+12} \times 3 + 5 \div b$

- Linearizada fica assim ...

- $A = 3 * ((3.14*r*r)/(b+12)) + 5/b$

Funções

- Agora imagine que você precisa representar a seguinte equação utilizando uma linguagem de programação:

- $$A = \frac{\pi r^2}{b+12} \times \sqrt{3 + 5 \div b}$$

- O problema todo é como *extrair* $\sqrt{\text{raiz quadrada}}$ da expressão $3 + 5 / b$, certo? Para isso precisamos utilizar uma função. Ex.:

- $$A = (3.14 * r * r) / (b + 12) * \text{Math.sqrt}(3 + 5 / b)$$

Funções (cont.)

- Funções são como “comandos extra” disponibilizados pela linguagem de programação ou por bibliotecas (coleções) externas e que ampliam o vocabulário da linguagem de programação.
- Existem inúmeras funções prontas para muitas tarefas que podem, em um primeiro momento, parecer impossíveis de serem feitas com os comandos conhecidos.
- A sintaxe das funções é a seguinte ...

```
var nome_variavel = funcao( <argumentos> )
```


Funções (exemplos)

- Veja alguns exemplos:
 - `Math.sqrt(<numero>)`: Extrai a raiz quadrada do número
 - `Math.round(<numero>)`: Arredonda o número
 - `Math.floor(<numero>)`: Retira a parte fracionária do número
 - `Math.random()`: Sorteia um valor entre 0 e 1
- Exemplos de objetos do tipo String
 - `var str = "Instituto Federal Farroupilha"`
 - `str.charAt(<numero>)`: retorna o caractere na posição indicada pelo n°
 - `str.indexOf(<string>)`: retorna a posição da substring
 - `str.substr(<numero1>, <numero2>)`: retorna um pedaço da string, da posição indicada pelo número 1 até a posição indicada pelo número 2.

(algumas funções possuem argumentos, outras não)

Outros operadores

- **+ (Concatenação de strings)**
 - O operador “+” (utilizado para adição em expressões matemáticas) se for utilizado com variáveis ou valores string fará a concatenação (junção) de ambos.
- **=== (igualdade restrita) e !== (desigualdade restrita)**
 - Este operador é utilizado para testar se uma variável tem o mesmo conteúdo e é do mesmo tipo (está presente em linguagens interpretadas como JavaScript e PHP). Veja este exemplo:

```
var a = "10"  
var b = 10  
  
alert(a == b)  
alert(a === b)
```

Outros operadores (cont.)

- **? : (operador ternário)**
 - Utilizado para decidir sobre duas instruções a partir de uma condição;
 - Sintaxe:
 - `<teste lógico> ? <instrução_true> : <instrução_false>`

`var situacao = (media >= 70) ? "Aprovado" : "Exame"`
 - O `<teste lógico>` será avaliado e caso retorne true (verdadeiro) então a `<instrução_true>` será executada, caso contrário (se o teste for false) então a `<instrução_false>` será executada.

Prioridade dos Operadores

- As linguagens de programação definem prioridades para seus operadores (para definir quais deles serão avaliados primeiro.
- Os parêntesis alteram a prioridade (os parêntesis mais internos serão sempre os primeiros a serem executados)

Tabela das Prioridade dos Operadores

Prioridade	Operador	Descrição
1º	() [] . funções	Parêntesis (), indexação de matrizes [], acesso a campos de objetos (.) e FUNÇÕES
2º	++ -- !	Incremento, decremento e negação
3º	* / %	Multiplicação, divisão, divisão de módulo
4º	+ -	Adição (ou concatenação) e subtração
5º	< <= > >=	Menor, menor ou igual, maior, maior ou igual
6º	== != === !==	Igualdade, desigualdade, igualdade estrita e desigualdade estrita
7º	&&	“E” lógico
8º	 	“OU” lógico
10º	? :	Operador ternário (condicional)

Para praticar

- Avalie as expressões a seguir e diga o resultado que será produzido:

a) $2 + 8 * 3$

b) $2 + 3 * 5 > 3 * 5 \ \&\& \ 3 * 4 < 4 + 9$

c) $2*(2+5)/(4-5)+\text{Math.sqrt}(81)$

Desafio – Funções utilitárias

- O desafio da semana consiste em pesquisar e demonstrar o uso de três funções do JavaScript.
- Para tanto desenvolva uma página com a descrição da função e um exemplo de sua utilização.
- Sugestões:
 - http://www.w3schools.com/js/js_string_methods.asp
 - http://www.w3schools.com/js/js_number_methods.asp
 - http://www.w3schools.com/js/js_date_methods.asp