



Estrutura e Banco de Dados

Conteúdo:

- JavaScript
 - Estruturas de decisão
 - Estruturas de repetição
 - Vetores
 - Formulários
 - Criação
 - Validação
- Claudiney Sanches Júnior

Estruturas de decisão

- São blocos de códigos que serão executados somente se uma dada condição for satisfeita.
- Para cada condição temos diversas ações diferentes.
- No nosso dia-a-dia estamos sempre tomando decisões, por exemplo:
 - se estiver chovendo levo o guarda-chuva.
 - se for final de semana e estiver sol irei para a praia.
- Temos disponível em JavaScript as seguintes estruturas de decisões:
 - if
 - if ... else
 - if ... else if ... else...
 - switch

Estrutura de decisão if

Decisão Simples (if)

Sintaxe:

```
if (condição)
{
    //instruções caso a condição seja verdadeira
}
```

Exemplo:

```
<script type="text/javascript">
var data_hora = new Date();
var hora = data_hora.getHours();
```

```
if (hora < 10)
{
    alert("Bom dia!!!");
}
```

```
</script>
```

para condição true (verdadeiro)

Na condição da estrutura, utilizamos os operadores de comparação e os operadores lógicos. Se a condição for **verdadeira**, o bloco é executado.

Estrutura de decisão if ... else ...

Decisão Composta (if ... else ...)

Sintaxe:

```
if (condição)
{
    //instruções caso a condição seja verdadeira
}
else
{
    //instruções caso a condição seja falsa
}
```

Exemplo:

```
<script type="text/javascript">
var data_hora = new Date();
var hora = data_hora.getHours();
```

```
if (hora < 10)
{
    alert("Bom dia!!!");
}
```

else

```
{
    alert("Olá, tenha um bom dia!!!");
}
```

```
</script>
```

para condição true (verdadeiro)

para condição false (falso)

Se a condição for **verdadeira**, o bloco do **if** é **executado**, senão, se a condição for **falsa**, o bloco do **else** é que será **executado**.

Estrutura de decisão if (encadeados)

Decisão Aninhada (if ... else ... if ... else ...)

Sintaxe:

```
if (condição 1)
{
    //instruções 1
}
else if (condição 2)
{
    //instruções 2
}
else
{
    //instruções 3
}
```

A decisão aninhada seria um conjunto de if's, para cada if devemos ter uma condição e poderemos caso necessário colocar o else (senão).

Também podemos ter um bloco if dentro de um outro bloco if, ou dentro de um bloco else.

Estrutura de decisão if (encadeados)

```
<script type="text/javascript">
  var data_hora = new Date()
  var hora = data_hora.getHours()
  var previsao_tempo = "chuvoso"
  if (hora < 10)
  {
    alert("Bom dia...");
    if (previsao_tempo == "chuvoso")
    {
      alert("Está chovendo, pego o guarda-chuva")
    }
  }
  else if (hora > 12 && hora < 18)
  {
    alert("Boa tarde...");
  }
  else
  {
    alert("Boa noite...");
  }
</script>
```

Perceba que podemos ter um bloco if dentro de outro também, o mesmo pode ocorrer com o bloco else, ou seja, podemos também ter um bloco if dentro do else. Se preferir, utilize chaves para cada sub bloco de if.

Operador ternário if

- Semelhante ao bloco if porém para casos bem simples, aceita somente uma instrução para cada caso
- Sintaxe:
(condição) ? //instrução se true : //instrução se false
- Exemplo
var x = 10;
var y = 20;
(x > y) ? alert("Sim") : alert("Não");
document.write((x < y) ? "Sim" : "Não");

Estrutura de decisão switch ... case

- Utilizado quando temos várias condições simples

- Sintaxe:

```
switch (valor) {  
    case valor1:  
        //instruções 1  
        break  
    case valor2:  
        //instruções 2  
        break  
    case valor3:  
        //instruções 3  
        break  
    default:  
        //instruções padrão  
}
```

Para cada caso devemos colocar o comando break, este comando irá finalizar o caso e evitar que o caso posterior seja executado.

O default é opcional, ele é executado quando nenhum caso anterior é acionado.

Os valores podem ser String, inteiros, caracteres ou reais.

Não podemos fazer comparações no case, como $x > y$, por exemplo.

Estrutura de decisão switch ... case

- Exemplo

```
<script type="text/javascript">
  var data_hora = new Date();
  dia_semana = data_hora.getDay();
  switch (dia_semana)
  {
    case 0:
      alert("Domingo de descanso merecido.");
      break;
    case 5:
      alert("Obaaa, sexta-feira.");
      break;
    case 6:
      alert("Maravilha, sabadão!!")
      break;
    default:
      alert("Semana longaaaa.");
  }
</script>
```

Estruturas de repetição

- São utilizadas quando necessitamos repetir um bloco de instruções.
- Podemos executar um laço de repetição com um número específico de vezes ou enquanto uma condição for verdadeira.
- Essas estruturas também são conhecidas como iteração ou loop.
- Em JavaScript temos:
 - for
 - while
 - do while

Estrutura de repetição for

- Utilizada quando sabemos o número de repetições que serão feitas.

- Sintaxe:

```
for (valor inicial; condição; incremento/decremento)
{
    //instruções que serão repetidas
}
```

- Exemplo:

```
<script type="text/javascript">
var cont;
for (cont = 0; cont < 10 ; cont++)
{
    document.write("Número: " + cont + "<b> , </b>");
}
</script>
```

Iremos executar o laço 10 vezes, a cada passagem, será impresso o valor da variável cont.

Estrutura de repetição while

- Executa um bloco de instruções enquanto uma certa condição for verdadeira

- Sintaxe:

```
while (condição)
{
    //instruções
    //alteração do valor da condição
}
```

- Exemplo:

```
<script type="text/javascript">
var cont = 1;
while (cont < 10)
{
    document.write("Número: " + cont + "<br />");
    cont = cont + 1; //ou cont++;
}
</script>
```

A variável deve ser inicializada antes do bloco.

Iremos executar o laço 10 vezes, a cada passagem, será impresso o valor da variável cont.

Perceba que dentro do laço é inserido o incremento da variável, com isso em algum momento a condição se tornará falsa.

Estrutura de repetição do...while

- Executa um bloco de instruções enquanto uma certa condição for verdadeira, porém na primeira passagem as instruções são executadas visto que o teste da condição é feito somente no final.

- Sintaxe:

```
do
{
    //instruções
    //alteração do valor da condição
} while (condição);
```

A variável deve ser inicializada antes do bloco.

- Exemplo:

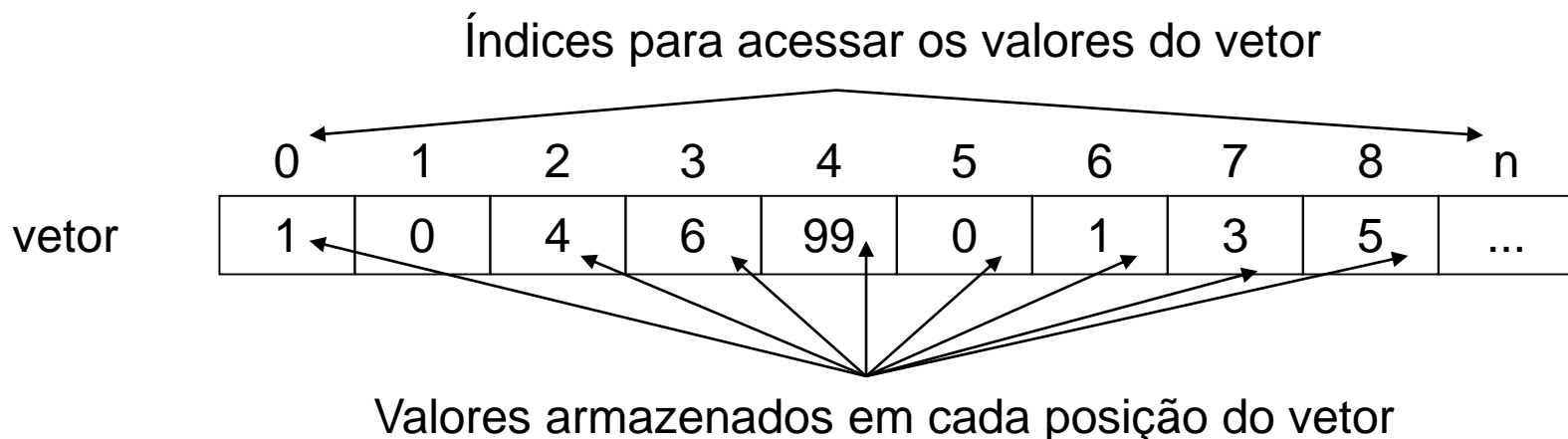
```
<script type="text/javascript">
var cont=1;
do
{
    document.write("Número: " + cont + "<br />");
    cont = cont + 1; //ou cont++;
} while (cont < 10);
</script>
```

Iremos executar o laço 10 vezes, a cada passagem, será impresso o valor da variável cont.

Perceba que dentro do laço é inserido o incremento da variável, com isso em algum momento a condição se tornará falsa.

Vetores

- Vetores são vistos como coleção de valores, onde através de um único nome de variável, temos várias posições para guardar informações.
- Em uma representação gráfica podemos visualizar um vetor da seguinte forma:



OBS: o primeiro índice de um vetor começa sempre em 0 (zero)

Vetores em JavaScript

- Para criar um vetor em JavaScript utilizamos o objeto Array, este objeto além de criar um vetor em memória também nos fornece diversos métodos para manipular o vetor criado, esses métodos serão listados posteriormente.
- Podemos criar um vetor de três forma:
 - Sem tamanho fixo (obviamente vazio, sem valores)
 - Com tamanho fixo (cada posição será vazia também)
 - Com valores (indicamos os dados que irão ser armazenados em cada posição)

Criando vetores de JS

```
<script language="javascript">  
//vetor sem tamanho definido  
var vetor1 = new Array();  
//vetor com tamanho definido  
var vetor2=new Array(3);  
//vetor com posições e valores definidos  
var vetor3=new Array("maça","banana","morango");  
</script>
```

- OBS: em todos os casos acima, podemos inserir mais posições sempre que necessário

Acessando um vetor

- Para visualizarmos ou atribuirmos um valor para um vetor devemos acessar cada posição através do índice, este índice deve ficar entre colchetes “[...]”.
- Exemplo:

```
<script type="text/javascript">  
    var frutas = new Array();  
    frutas[0]="maça";  
    frutas[1]="banana";  
    frutas[2]="morango";  
    alert(frutas[2]);  
</script>
```

Acessando um vetor

- Em quase todos os casos em que trabalhamos com vetores, utilizamos uma estrutura de repetição para acessar suas posições, a mais comum seria a estrutura for, porém também podemos usar o do...while ou o while.
- Exemplo

Sem o uso de repetições (for, while...)

```
<script type="text/javascript">
    var frutas = new Array();
    frutas[0]="maça";
    frutas[1]="banana";
    frutas[2]="morango";
    document.write(frutas[0]);
    document.write(frutas[1]);
    document.write(frutas[2]);
</script>
```

Com uso de repetições (for)

```
<script type="text/javascript">
    var i;
    var frutas = new Array();
    frutas[0]="maça";
    frutas[1]="banana";
    frutas[2]="morango";
    for (i=0;i<=2;i++)
    {
        document.write(frutas[i]);
    }
</script>
```

Propriedade de um vetor (objeto Array)

- length

seta ou retorna o número de elementos de um vetor

Ex:

```
var vetor1 = new Array();  
vetor1[0] = "maça";  
vetor1[1] = "banana";  
document.write("Tamanho: " + vetor1.length);
```

Métodos de um vetor (objeto Array)

- `concat()`
Retorna a junção (cópia) de dois ou mais vetores
Ex:
`//retorna a junção do vetor1 com o vetor2`
`vetor1.concat(vetor2);`
`//retorna a junção do vetor1 com o vetor2 e vetor3`
`vetor1.concat(vetor2,vetor3);`
- `reverse()`
inverte a ordem dos elementos no vetor
Ex:
`vetor1.reverse();`
- `sort()`
ordena os elementos do vetor
Ex:
`vetor1.sort()`
- `toString()`
converte e retorna um vetor em String
Ex:
`vetor1.toString();`

Formulário

- Uma das várias possibilidades da linguagem JS é a validação de formulários, porém antes de iniciarmos esse estudo, vamos ver alguns pontos importantes na criação dos formulários.
- Usamos formulário para enviar dados para o servidor.
- Vamos iniciar pela tag form, ela possui os seguintes atributos (principais):
 - name
define o nome para o formulário
 - id/classe
define um estilo individual/classe para o formulário
 - Method
define o método de envio das informações, pode ser:
 - GET (envia os dados junto com o URL)
 - POST (envia os dados separado do URL, ou seja, em uma string separa)
 - action
define para qual URL os dados serão enviados

Exemplo de formulário

```
<form id="form1" name="form1" method="post"
action="cadastro.php">
```

```
<p>Login:
```

```
<input type="text" name="login" id="login" /><br />
```

```
Senha:
```

```
<input type="text" name="senha" id="senha" /><br />
```

```
<input type="submit" name="button" id="button" value="Enviar" />
```

```
</p>
```

```
</form>
```

Elementos de um formulário

- Quando criamos um formulário, podemos inserir diversos elementos, como por exemplo:
 - Caixa de texto
`<input type="text" name="nome" id="nome" />`
 - Campo de texto (várias linhas)
`<textarea name="mensagem" id="mensagem"></textarea>`
 - Botões
`<input type="submit" value="Enviar" />`
 - Caixa de seleção/lista
`<select name="cidade" id="cidade">`
 `<option value=""></option>`
 `<option value="SP">SP</option>`
 `<option value="RJ">RJ</option>`
`</select>`
 - Botões de opções
M `<input type="radio" name="sexo" id="sexo_m" />`
F `<input type="radio" name="sexo" id="sexo_f" />`
 - Listas de checkbox
Op1 `<input type="checkbox" name="lista" id="op1" />`
Op2 `<input type="checkbox" name="lista" id="op2" />`
Op3 `<input type="checkbox" name="lista" id="op3" />`

Comentário sobre os elementos do form

- No exemplo anterior foi demonstrado diversos elementos de um formulário, o que existe de comum a todos?
- Perceba que todos os elementos possuem um “name” e um “id”, mas por que esses dois atributos juntos?
- O id, como visto em CSS, serve para aplicar um estilo a tag HTML e principalmente serve para identificar esse tag, com isso podemos manipulá-la posteriormente em JavaScript, lembre-se, o id deve ser único em um página, não pode se repetir.
- Já o atributo name, serve para armazenar o valor do elemento, é esse name que enviamos para o servidor com os dados do formulário, podemos associá-los as variáveis, cada campo possui a sua variável para armazenar o valor inserido pelo usuário, e ao clicar no botão “submit”, enviamos essas variáveis para o servidor.

Exemplo form+script php

- Arquivo HTML

```
<form id="form1" name="form1" method="get" action="cadastro.php">  
  <p>Login:  
  <input type="text" name="login" id="login1" /><br />  
  Senha:  
  <input type="text" name="senha" id="senha1" /><br />  
  <input type="submit" name="button" id="button" value="Enviar" />  
  </p>  
</form>
```

- Arquivo cadastro.php

```
<?php  
  echo "Login".$_GET["login"]; //imprime o login informado  
  echo "Senha".$_GET["senha"]; //imprime a senha informada  
?>
```

Acessando os elementos do form em JS

- Antes de aprendermos a validar um formulário, devemos aprender como acessar seus elementos.
- Podemos acessar os elementos de duas formas:
 - Através do seu id (cada elemento possui o seu, mesmo as listas de checkbox e radio possuem id únicos)
 - Através do seu name (cada elemento possui o seu, e as listas de checkbox e radio possuem um único name)

Acessando os elementos

- Para acessar os elementos pelo ID, utilizamos o comando:

Sintaxe:

```
document.getElementById("id_do_elemento").propriedades
```

- Para acessar os elementos pelo NAME, utilizamos o comando:

Sintaxe:

```
document.getElementsByName("name_do_elemento")[indice].propriedade
```

Acessando os elementos (cont.)

- O **getElementById** retorna o elemento propriamente dito
- O **getElementsByName** retorna uma coleção (vetor), onde cada posição corresponde a um elemento, geralmente utilizamos esse comando para acessar listas do tipo radio e checkbox.

Algumas propriedades dos elementos

- **value**
retorna o valor inserido ou selecionado no elemento, aplicável praticamente em todos os elementos do formulário
- **checked**
retorna ou seta um elemento do tipo checkbox ou radio, aceita e retorna valores booleanos, true ou false
- **options[]**
acessa os elementos de uma lista do tipo select
- **disabled**
habilita ou desabilita um elemento, aceita ou retorna os valores true ou false
- **focus()**
função que move o foco para o elemento que está associado

Validando um formulário

- Antes de enviarmos um formulário para o servidor, devemos validar o mesmo para termos certeza de que o usuário entrou com os valores exigidos.
- Após o usuário preencher os dados, ele clica no botão enviar que dispara o evento onsubmit, é neste evento que devemos associar nossa função para validar o formulário.
- Toda nossa lógica de validação deverá ficar nessa função, se encontrar algo errado avisamos o usuário e retornamos false para que a ação do envio seja cancelada, caso esteja tudo certo, retornamos true para que o formulário seja enviado.

Exemplo 1

```
<html>
<head>
<title>Exemplo 1</title>
<script language="javascript">
function validaCampos(){
    if (document.getElementById("login").value == "" ||
        document.getElementById("senha").value == "")
    {
        alert("Preencha todos os campos");
        return false;
    }
    return true;
}
</script>
</head>
```

Exemplo 1

```
<body>
<form name="form1" method="post" action="" onsubmit="return validaCampos()">
Login:
<input name="login" type="text" id="login" size="15">
<br>
Senha:
<input name="senha" type="password" id="senha" size="10">
<br>
<input type="submit" name="button" id="button" value="OK">
</form>
</body>
</html>
```


Exemplo 2 (formulário)

```
<form name="form1" method="post" action="" onsubmit="return validaCampos()">
```

```
<p>Nome:
```

```
<input name="nome" type="text" id="nome" size="60">
```

```
<br>
```

```
<br>
```

```
Sexo:
```

```
<input type="radio" name="sexo" id="sexo1" value="radio"> M
```

```
<input type="radio" name="sexo" id="sexo2" value="radio2">F
```

```
<br>
```

```
<br>
```

```
Cidade:
```

```
<select name="cidade" id="cidade">
```

```
<option value=""></option>
```

```
<option value="SP">S&atilde;o Paulo</option>
```

```
<option value="RJ">Rio de Janeiro</option>
```

```
</select>
```

```
<br>
```

```
<br>
```

Exemplo 2 (formulário)

Filmes favoritos:

<input name="filmes" type="checkbox" id="filmes1" value="Terror">

Terror

<input name="filmes" type="checkbox" id="filmes2" value="Ação">

Ação

<input name="filmes" type="checkbox" id="filmes3" value="Drama">

Drama

<input name="filmes" type="checkbox" id="filmes4" value="Comédia">

Comédia

Mini-currículo:

<textarea name="mini_curriculo" id="mini_curriculo" cols="50" rows="5"></textarea>

<input type="button" name="button2" id="button2" value="Mostrar Filmes selecionados
(getElementById)" onclick="mostraFilmes_ID()" >

<input type="button" name="button3" id="button3" value="Mostrar filmes selecionados
(getElementsByName)" onclick="mostraFilmes_NAME()">

</p>

<p>

<input type="submit" name="button" id="button" value="Gravar">

</p>

</form>

Exemplo 2 (função de validação)

- Para validar o formulário, criamos a seguinte função

```
<script language="javascript">
```

```
function validaCampos(){
```

```
    //instruções
```

```
}
```

```
</script>
```

- Esta função é chamada no evento onsubmit do formulário

Exemplo 2 (códigos da função de validação)

```
//campo de texto nome
//podemos criar uma variável para guardar o valor de um campo
var nome = document.getElementById("nome").value;
if (nome == "" || nome.length < 5)
{
    alert("Preencha corretamente seu nome");
    document.getElementById("nome").focus();
    return false;
}
//campo sexo com getElementById
//neste exemplo iremos trabalhar no if diretamente sem criar variável
//mas também poderíamos fazer igual ao anterior
if (document.getElementById("sexo1").checked == false &&
document.getElementById("sexo2").checked == false)
{
    alert("Escolha seu sexo");
    return false;
}
```

Exemplo 2 (códigos da função de validação)

```
//campo cidade
if (document.getElementById("cidade").value == ""){
    alert("Escolha a cidade onde mora.");
    document.getElementById("cidade").focus();
    return false;
}
//lista de filmes, o usuário pode marcar vários
//iremos validar se ele marcou pelo menos 1
var i;
for (i=1;i<=4;i++){
    if (document.getElementById("filmes"+i).checked == true){
        break;
    }
    if (i==4){
        alert("Escolha pelo menos um filme");
        return false;
    }
}
```

Exemplo 2 (códigos da função de validação)

```
//campo mini-curriculo
if (document.getElementById("mini_curriculo").value == "")
{
    alert("Insira um resumo do seu currículo");
    document.getElementById("mini_curriculo").focus();
    return false;
}
return true;
```

Exemplo 2 (funções para mostrar filmes)

```
function mostraFilmes_ID(){
    var conteudo="";
    var i;
    for (i=1;i<=4;i++){
        if (document.getElementById("filmes"+i).checked == true){
            conteudo = conteudo + document.getElementById("filmes"+i).value + "\n";
        }
    }
    alert(conteudo);
}
function mostraFilmes_NAME(){
    var conteudo="";
    var i;
    for (i=0;i<=3;i++){
        if (document.getElementsByName("filmes")[i].checked == true){
            conteudo = conteudo + document.getElementsByName("filmes")[i].value + "\n";
        }
    }
    alert(conteudo);
}
```

Exercício 1

- Crie um arquivo HTML que contenha um formulário conforme exemplo abaixo:

Faixa salarial	Alíquota
Até R\$ 1.434	0% (isento)
De R\$ 1.434 a R\$ 2.150	7,5%
De R\$ 2.150 a R\$ 2.866	15%
De R\$ 2.866 a R\$ 3.582	22,5%
Acima de R\$ 3.582	27,5%

Calculo do Imposto de Renda

Registro:

Nome:

Salário:

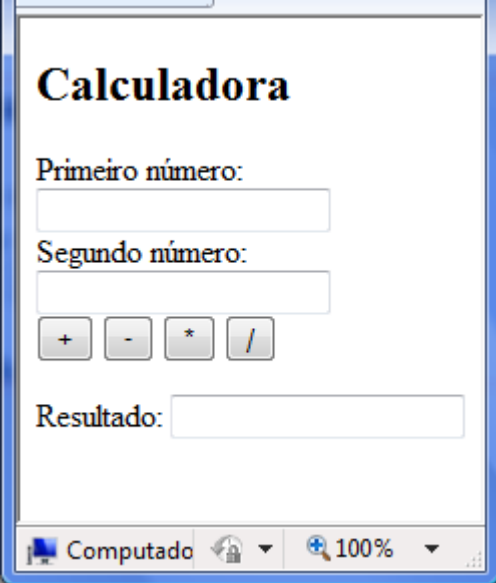
Imposto de Renda:

Computador | Modo Protegido: Desativa 100%

- Neste mesmo arquivo crie uma função Javascript que calcule o imposto de renda a partir do salário informado, respeitando a faixa salarial conforme apresentado na tabela. Depois coloque o resultado do calculo na caixa de texto correspondente ao IR.

Exercício 2

- Crie um arquivo HTML que contenha um formulário conforme exemplo abaixo:
- Neste mesmo arquivo crie uma função Javascript que calcule a operação acionada pelo clique no botão correspondente. Depois coloque o resultado do calculo na caixa de texto correspondente ao Resultado.



The screenshot shows a web browser window with a single tab titled 'Computado'. The browser's address bar is empty. The page content is a form titled 'Calculadora' in bold black text. Below the title, there are two input fields: 'Primeiro número:' and 'Segundo número:'. Between these fields are four buttons: '+', '-', '*', and '/'. Below these buttons is a 'Resultado:' label followed by a larger input field. The browser's status bar at the bottom shows 'Computado', a lock icon, and a zoom level of '100%'.

Exercício 3

- Crie um arquivo HTML que contenha um formulário conforme exemplo abaixo:

Planilha de Notas

RGM	Nome	Exercícios	Prova Par.	Prova Reg.	Susbtitutiva	Média	Exame	Média Final
12345-6	Amanda	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
23456-7	Ailton	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
34567-8	Beatriz	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
45678-9	Carlos	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
56789-0	Edson	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
67890-1	Flávia	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
78901-2	Fernanda	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
89012-3	Marcos	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
90123-4	Neli	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	

- Neste mesmo arquivo crie uma função Javascript que calcule o fechamento das Médias para todos os alunos baseado nas notas informadas na caixas de texto. Crie também uma função que calcule somente os fechamentos da médias finais usando os valores das Médias e dos Exames. Depois coloque o resultado do calculo de cada aluno na coluna correspondente.

Exercício 4

- Crie um arquivo HTML que contenha um formulário conforme exemplo abaixo:

Agenda de Contatos

Nome:

Endereço: , número , compl.

Bairro: - Cidade: - Estado:

e-mail:

Telefones:

Residencial:

Celular:

Cadastrados

Nome	Endereço	e-mail	Tel. Residencial	Tel. Celular
Clara	R. das Camélias, 2, ap. 2, Vila Nova, São Paulo, SP		(11) 4567-8765	
Jéssica	R. das Flores, 3, Vila Velha, São Paulo, SP	jessica@email.com.br	(11) 4567-8765	9567-8765
Roberto	R. dos Cantos, 4, Vila Primavera, São Paulo, SP		4567-8765	
Caio	R. Sem Nome, 9, ap. 2 - bl. 1, Vila Outono, São Paulo, SP	caio@email.com.br	(11) 4567-8765	

- Neste mesmo arquivo crie uma função Javascript que valide os dados obrigatórios.
- Crie uma outra função que acrescente os dados na lista de cadastrados somente se eles passarem pela validação.