ISTEC
Programação I
Perguntas tipo

1. Indique a diferença entre os três blocos de instruções apresentados:

```
var str1 = "this is a simple string";
var num1 = 1.45;
var answer = true;

var str1 = String("this is a simple string");
var num1 = Number(1.45);
var bool1 = Boolean(true);

var str2 = new String("this is a simple string");
var num2 = new Number(1.45);
var bool2 = new Boolean(true);
```

2. Apresentar , utilizando o método *alert*, uma tabela dos números de 5 a 15 e os respetivos quadrados e cubos.
3. Apresentar, utilizando o método *write*, os primeiros 20 números da série de Fibonacci. A série de Fibonacci traduz-se numa sequência de números
   1, 1, 2, 3, 5, ...
   onde cada número da sequência, a partir do segundo, é a soma dos dois números anteriores.
4. Apresentar o maior de três números introduzidos pelo utilizador através do método *prompt*. Sugere-se a utilização da função Math.max().
5. Modifica o script pedido no ponto 2, utilizando o método *prompt*, para receber um número *n* que representa a quantidade números a apresentar.
6. Receber uma *string*, através do método *prompt*, e apresentar a mensagem "Nome válido" ou "Nome inválido", dependendo se os nomes foram introduzidos respeitando o seguinte formato:
   `último nome, primeiro nome, inicial do meio`
   (Nenhum dos nomes deve ter mais de 15 carateres).
7. Apresentar as palavras de uma frase indicada pelo utilizador, através do método *prompt*, em ordem alfabética.
8. Modificar o script do ponto 7, de forma a receber um segundo *input* do utilizador, indicando o tipo de ordenação a efetuar: "ascendente" ou "descendente". A ordenação das palavras da frase deve ter em conta a indicação do utilizador.

9. Construir uma função intitulada *sem_Zeros( )* que recebe um *array* de números e que permite remover todos os valores zero existentes nesse *array*. A função deve, ainda, devolver *true,* se o array indicado incluía valores zero, ou *false*, caso se verifique a situação contrária.
10. Construir uma função intitulada *e_nomes( )* que recebe um *array* de nomes e que devolve o número de nomes presentes nesse *array* que acabam em "el" e "é".
11. Construir uma função intitulada *primeira_vogal( )* que recebe uma *string* e que devolve a posição da primeira vogal.
12. Construir uma função intitulada *contador( )* que recebe um *array* de números e que devolve o número de números negativos, zeros e valores positivos existentes no *array*. Utilizar a instrução *switch* na função.
13. Construir uma função intitulada *tst_nome( )* que recebe uma *string* e que devolve *true,* se a *string* fornecida tem o formato:
    ```
    string1, string2 letra
    ```
    onde ambas *strings* tem de estar em minúsculas exceto a primeira letra que deve estar em maiúscula. Se não satisfazer o padrão deve devolver *false*.
14. Construir uma função intitulada *linha_media( )* que recebe um *array* de *array* de números e que devolve um *array* com a média de cada linha da matriz fornecida.
15. Construir uma função intitulada *numero_inverso( )* que recebe um número e que devolve outro número cujos dígitos estão na ordem inversa do número fornecido.
16. Utilizando expressões regulares escreva o código que permita indicar as posições iniciais na *string* onde haja correspondências de palavras que começam por um **t** e terminam com a letra **e**. Considere como exemplo a seguinte frase: "*Now is the time and this is the time and that is the time*".
17. Construa uma função que permita substituir todos os valores presentes num *array* que correspondam a um padrão indicado por um novo valor.
18. Construa uma função que converta um *array* de números decimais para um novo *array* de correspondentes hexadecimais.
19. Construa um script *JavaScript* que permita verificar se todos os elementos de um *array* correspondem a um determinado padrão.
20. Construa um script *JavaScript* que permita verificar se existe algum elemento de um *array* correspondem a um determinado padrão.
21. Construa uma função que permita devolver o somatório de todos os elementos do *array* indicado.
22. Escreva uma função, aplicando a técnica de *closuses* que permita, sem criar uma variável global, multiplicar o primeiro número fornecido com outros números.
23. Escreva uma função anónima que permita efetuar o incremento da sua variável interna, sempre que tal for solicitado. Não utilizar variáveis globais.
24. Construa um iterador (*constroiIterador*) que permita aceder de forma consecutiva

aos objetos um *array* de objetos. Por exemplo, considere o seguinte *array* de objetos:

```
var tarefas = [
    {"tarefa": function() {
                return 'Tarefa 1';
            }
    },
    {"tarefa": function() {
                return 'Tarefa 2';
            }
    },
    {"tarefa": function() {
                return 'Tarefa 3';
            }
    }
];
```

Para melhor compreensão apresentam-se instruções de utilização e teste do iterador:

```
> var sequenciaTarefas = constroiIterador(tarefas);
> sequenciaTarefas.next().value.tarefa();
  'Tarefa 1'
> sequenciaTarefas.next().value.tarefa();
  'Tarefa 2'
```

25. Acrescente ao objeto *String* um novo método que permita devolver o conteúdo da *String* em formato de exclamação. Para melhor compreensão considere o seguinte código:

```
> var str = 'olá';
> str.exlamacao();
  olá!
```

26. Crie o código *JavaScript* através do qual seja possível instanciar novos objetos do tipo "Musica" cujas caraterísticas são:

    a. três propriedades: titulo, artista e categoria;

    b. um método próprio "*ConcatenaTituloArtista*" que devolve uma *string* com a informação do título e do artista;

    c. um método "AdicionaTipo" partilhado por todos os objetos, que permite adicionar ou atualizar o tipo de música.

27. Crie código *JavaScript* que permita instanciar novos objetos "Livro" e "LivroTecnico" considerando que:

    d. Os objetos do tipo **Livro** possuem duas propriedades (titulo e autor) e dois métodos (**getAutor** e **getTitulo**) que devolvem respetivamente o conteúdo das propriedades autor e titulo

    e. Os objetos **LivroTecnico** herdam todo o conteúdo de **Livro**, acrescido da propriedade categoria e do método **getLivro** que descreve o livro sua totalidade (titulo, autor e categoria)

28. Considere a seguinte página html:

```
<!DOCTYPE html>
```

```
<html>
	<head>
		<title>Coleção de imagens</title>
	</head>
	<body>
		<p><img src="firstimage.jpg" alt="image description" /></p>
		<p><img src="secondimage.jpg" alt="image description"
		/></p>
		<p><img src="thirdimage.jpg" alt="image description" /></p>
	</body>
</html>
```

Construa o script que permita acrescentar uma nova imagem (someimg.jpg) ao documento.

29. Considere a seguinte página html

```
<!DOCTYPE html>
<html>
	<head>
		<title>Searching for strings</title>
	<style>
		.found {
			background-color: #ff0;
		}
	</style>
</head>
<body>
	<form id="texto">
		<textarea id="editor" cols="150" rows="10">
		</textarea>
	<p>
		Search pattern: <input id="padrao" type="text" />
	</p>
	</form>
	<button id="procura">Search for pattern</button>
	<div id="resultadopesquisa"></div>
</body>
</html>
```

Elabore um script *JavaScript* que permita, ao clicar no botão "procura", apresentar o texto, na div "resultadopesquisa", destacando as palavras que obedecem ao padrão definido no *input* "padrao" através da propriedade "**.found**" definida em CSS.

30. Considere a seguinte página html:

```
<!DOCTYPE html>
<html>
    <head>
        <title>TODO</title>
        <meta charset="UTF-8">
        <style>
            #redbox{
                position: absolute;
                left: 100px;
                top: 100px;
                width: 200px; height: 200px;
                background-color: red;
            }
        </style>
```

```
    </head>
    <body>
        <div id="redbox"></div>
    </body>
</html>
```

Elabore um script *JavaScript* que permita deslocar o **div "redbox"** de 5 em 5 px de forma automática. O movimento começa e termina sempre que o utilizador clica no contentor div.

31. Considere a seguinte página html:

```
<!DOCTYPE html>
<html>
    <head>
        <title>TODO</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <table id="table1">
        <tr>
            <td>Washington</td><td class="quant">145</td>
        </tr>
        <tr>
            <td>Oregon</td><td class="quant">233</td>
        </tr>
        <tr>
            <td>Missouri</td><td class="quant">833</td>
        </tr>
        </table>
    </body>
</html>
```

Elabore um script *JavaScript* que acrescente à tabela uma nova linha que contenha a soma dos valores das restantes linhas da tabela.

32. Considere o seguinte documento html:

```
<!DOCTYPE html>
<html>
      <head>
            <meta charset="utf-8">
            <title>CSS Computed Style</title>
            <style>
                  .test {
                        background-color: red;
                        width: 100px;
                        height: 100px;
                  }
            </style>
      </head>
      <body>
            <div id="test"><p>Hi</p></div>
      </body>
</html>
```

Construa o script que permita acrescentar um atributo *class* com o valor *test* ao elemento div do documento.

33. Considere o seguinte documento html:

```
<html>
    <head>
        <title>lista</title>
        <style>
            .elem{
                background-color: #eeffee;
            }
        </style>
    </head>
    <body>
        <div>
            <ul id="lista">
                <li>Um</li>
                <li>Dois</li>
                <li>Três</li>
                <li>Quatro</li>
                <li>Cinco</li>
                <li>Seis</li>
                <li>sete</li>
                <li>Oito</li>
                <li>Nove</li>
            </ul>
        </div>
    </body>
</html>
```

Construa o script que permita colocar a lista não ordenada do documento com aparência de "zebra". Ou seja, apenas os elementos de ordem par têm cor de fundo definida no seletor CSS ".elem".

34. Considere o seguinte documento html:

```
<html>
    <head>
        <title>lista</title>
    </head>
    <body>
        <form>
            <input type="text" id="editor" size="150" />
        </form>
        <button id="insere">Insere parágrafo</button>
        <div id="target">
        </div>
</html>
```

Construa ou script que permita, ao clicar no botão, acrescentar o conteúdo da caixa de texto como um novo parágrafo dentro do contentor div.

35. Considere o seguinte documento html:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Adding and Removing Elements</title>
        <style>
            table {
                  border-collapse: collapse;
            }
```

```
            td, th {
                  padding: 5px;
                  border: 1px solid #ccc;
            }
            tr:nth-child(2n+1)
            {
                  background-color: #eeffee;
            }
        </style>
    </head>
    <body>
        <table id="mixed">
            <tr>
                  <th>Value One</th>
                  <th>Value two</th>
                  <th>Value three</th>
            </tr>
        </table>
        <div id="result"></div>
    </body>
</html>
```

Construa um script que permita, adicionar três linhas à tabela com os seguintes dados:
[123.45, "apple", true];
[65, "banana", false];
[1034.99, "cherry", false];
Construa, ainda, código que permita, ao clicar numa linha da tabela, eliminar a linha em causa.

36. Considere o seguinte documento html:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Overlay</title>
        <style>
        .overlay
        {
            background-color: #000;
            opacity: .5;
            filter: alpha(opacity=50);
            position: fixed; top: 0; left: 0;
            width: 100%; height: 100%;
            z-index: 10;
        }
        .overlaymsg
        {
            position: absolute;
            background-color: yellow;
            padding: 10px;
            width: 200px;
            height: 200px;
            font-size: 2em;
            z-index: 11;
            top: 50%;
            left: 50%;
            margin-left: -100px;
            margin-top: -100px;
        }
```

```
        </style>

    </head>
    <body>
        <p>Existing material.</p>
</body>
</html>
```

Construa um script *JavaScript* que permita, no arranque do programa, criar um fundo de sobreposição (*overlay*) e de seguida apresente uma janela (div) com a configuração definida na class *overlaymsg* e com a mensagem "*Não se esqueça de indicar a sua mailing list! (Clique nesta janela para fechar)*". A janela e o fundo de sobreposição devem ser eliminados da árvore DOM quando o utilizador clicar na janela de mensagem.

37. Construir um ficheiro html que contenha cinco *radio buttons*, etiquetados com vermelho, azul, verde, amarelo e laranja, respetivamente e programar os eventos para os respetivos botões de forma a emitir mensagens que apresenta a cor favorita escolhida.
38. Construir, testar e validar um documento HTML que contenha *checkboxs* para: maça (59 cêntimos cada), laranja (49 cêntimos cada) e banana (39 cêntimos cada) e um botão *Submit*. Cada *checkbox* deve possuir o seu próprio manipulador do evento *click*. Estes manipuladores devem acumular o custo da respetiva fruta ao custo total. O evento do botão submit deve apresentar uma caixa de diálogo *alert* com a mensagem "*O custo total é de xxx €*", onde *xxx* corresponde ao custo total da fruta escolhida, incluindo 6 por cento de IVA. Este evento deve evitar o funcionamento por defeito do botão *submit*.
39. Construir, testar e validar um documento HTML semelhante ao do exercício 38. Neste caso, utilize caixas de texto em vez caixas *checkbox*. As caixas de texto devem receber um número que corresponde o número de unidades de cada peça de fruta. O documento deve funcionar exatamente como o do Exercício 38.
40. Construir, testar e validar um documento HTML semelhante ao exercício 39, mas implementando validações do conteúdo das caixas de texto. Essas validações devem assegurar que os valores recebidos são números compreendidos entre 0 e 99.
41. Modificar o exercício 40 de forma a adicionar uma propriedade *max* com o valor máximo e uma propriedade *min* com o valor mínimo permitidos em cada caixa de texto do tipo *number* (ver input type_number). O manipulador de eventos de cada caixa de texto deve usar essa propriedades para validar o conteúdo introduzido em cada caixa.
42. Construir, testar e validar um documento HTML que contenha as seguintes de informação do utilizador: último nome, primeiro nome, inicial do meio, idade (maior do que 17) e peso (entre 80 a 300) . Devem ser criados manipuladores de eventos para os elementos do formulário que recolhem as informações. Os manipuladores devem verificar os dados de entrada segundo os parâmetros definidos. Devem ser emitidas mensagens em janela de *alert* quando forem detectados erros.
43. Considere o seguinte código html:

```
<html>
    <head>
```

```
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
    </head>
    <body>
        <form>
            <input name="" type="checkbox" />
            <input name="" type="radio" />
            <input name="" type="text" />
            <input name="" type="button" />
        </form>
        <form>
            <input name="" type="checkbox" />
            <input name="" type="radio" />
            <input name="" type="text" />
            <input name="" type="button" />
        </form>
            <input name="" type="checkbox" />
            <input name="" type="radio" />
            <input name="" type="text" />
            <input name="" type="button" />
        <script type="text/JavaScript" src="js/jquery-
1.11.3.js"></script>
</body>
</html>
```

Crie código jQuery que permita informar:

a. Quantos caixas de input existem nos formulários do documento;
b. Quantas caixas de input existem no primeiro formulário;
c. Quantas caixas de input existem em todo o documento;

44. Considere o seguinte código html:

```
<html>
    <head>
    </head>
    <body>
          <p>Ut ad videntur facilisis <em>elit</em> cum. Nibh insitam
          erat facit <em>saepius</em> magna. Nam ex liber iriure et
          imperdiet. Et mirum erosiis te habent. </p>
          <p>Claram claritatem eu amet dignissim magna. Dignissim quam
          elit facer eros illum. Et qui ex esse <em>tincidunt</em>
          anteposuerit. Nulla nam odio ii vulputate feugait.</p>
          <p>In quis <em>laoreet</em> te legunt euismod. Claritatem
          <em>consuetudium</em> wisi sit velit facilisi.</p>
        <script src="js/jquery-1.11.3.js"></script>
    </body>
</html>
```

Escreva o código *jQuery* que permita informar quantas palavras em itálico têm os parágrafos do documento.

45. Considere o seguinte código html:

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
    </head>
```

```
        <body>
            <h3>Anchors</h3>
            <a href='#'>Anchor Element</a>
            <a href='#'>Anchor Element</a>
            <a href='#'>Anchor Element</a>
        </body>
    </html>
```

Escreva script, utilizando *jQuery*, que permita:

a. quando clicar no elemento <h3> deve ser inserido na árvore DOM um novo elemento <a> em tudo semelhante aos já existentes no documento.
b. quando clicar em cima de um elemento <a> esse mesmo elemento deve ser removido da árvore DOM.

46. Considere o seguinte código html:

```
<html>
    <head>
    </head>
    <body>
        <ul>
        <li class='remove'>name</li>
        <li>name</li>
        <li class='remove'>name</li>
        <li class='remove'>name</li>
        <li>name</li>
        <li class='remove'>name</li>
        <li>name</li>
        <li class='remove'>name</li>
        </ul>
        <button>Substitui</button>
    </body>
</html>
```

Escreva um script, utilizando *jQuery*, que substitua os items da lista com a class "remove" por um item com o conteúdo textual "removido".

47. Considere o seguinte código html:

```
    <html>
        <head>
        </head>
        <body>
            <a>jquery.com</a>
        </body>
    </html>
```

Escreva o código jQuery que permita atribuir o atributo *href*, com o valor http://www.jquery.com , ao elemento <a> do documento.

48. Considere o seguinte código html:

```
    <html>
    <head>
    </head>
    <body>
    <p></p>
    </body>
    </html>
```

Escreva o código *jQuery* que permita colocar dentro do elemento <p> o seguinte

conteúdo html "`'<strong>Hello World</strong>, I am a <em><p></em> element.'`". Depois disto, deve, também, mostrar, através de da instrução *alert*, o conteúdo html do elemento <body> e o conteúdo textual do elemento <p>.

49. Considere o seguinte código html:

```
<a href="/category">Category</a>
<ul id="nav">
<li><a href="#anchor1">Anchor 1</a></li>
<li><a href="#anchor2">Anchor 2</a></li>
<li><span><a href="#anchor3">Anchor 3</a></span></li>
</ul>
<div id="content">
<h1>Main title</h1>
<h2>Section title</h2>
<p>Some content...</p>
<h2>Section title</h2>
<p>More content...</p>
</div>
<ol>
<li>First item</li>
<li>Second item</li>
<li>Third item</li>
<li>Fourth item</li>
</ol>
<table>
<tr><td>0</td><td>even</td></tr>
<tr><td>1</td><td>odd</td></tr>
<tr><td>2</td><td>even</td></tr>
<tr><td>3</td><td>odd</td></tr>
<tr><td>4</td><td>even</td></tr>
</table>
```

    a. Qual a instrução *jquery* que permite selecionar os elementos que têm o elemento <a> com descendente direto?
    b. Qual a instrução *jquery* que permite selecionar apenas elementos <h2> imediatamente seguidos por elementos <h1>?
    c. Qual a instrução *jquery* que permite selecionar o primeiro elemento da lista ordenada?
    d. Qual o código *jquery* que permite preencher com uma cor de fundo das linhas pares da tabela. Considera para o efeito que existe um seletor CSS com a seguinte configuração: `table tr.even { background: #CCC;}`.

50. Considere o seguinte código html:

```
<html>
    <head>
        <title>TODO</title>
        <meta charset="UTF-8">
        <style type="text/css">
            .even { background-color: #ffffff; }
            .odd { background-color: #cccccc; }
        </style>
    </head>
    <body>
        <h2>Family Members</h2>
        <ul>
            <li>Ralph</li>
            <li>Hope</li>
            <li>Brandon</li>
            <li>Jordan</li>
```

```
            <li>Ralphie</li>
        </ul>
    </body>
</html>
```

Escreva código *jQuery* que permita colocar, alternadamente, os elementos da lista com um cor de fundo diferente.

51. Considere o seguinte pedaço de código html:

```
...
<h2>Eastern Conference</h2>
<ol id="east">
<li>Boston Bruins</li>
<li>Washington Capitals</li>
<li>New Jersey Devils</li>
<li>Pittsburgh Penguins</li>
<li>Philadelphia Flyers</li>
<li>Carolina Hurricanes</li>
<li>New York Rangers</li>
<li>Montreal Canadians</li>
<li>Florida Panthers</li>
<li>Buffalo Sabres</li>
<li>Ottawa Senators</li>
<li>Toronto Maple Leafs</li>
<li>Atlanta Thrashers</li>
<li>Tampa Bay Lightning</li>
<li>New York Islanders</li>
</ol>
<h2>Western Conference</h2>
<ol id="west">
<li>San Jose Sharks</li>
<li>Detroit Red Wings</li>
<li>Vancouver Canucks</li>
<li>Chicago Blackhawks</li>
<li>Calgary Flames</li>
<li>St. Louis Blues</li>
<li>Columbus Blue Jackets</li>
<li>Anaheim Ducks</li>
<li>Minnesota Wild</li>
<li>Nashville Predators</li>
<li>Edmonton Oilers</li>
<li>Dallas Stars</li>
<li>Phoenix Coyotes</li>
<li>Los Angeles Kings</li>
<li>Colorado Avalanche</li>
</ol>
...
```

Escreva código *jQuery* que permita colocar uma linha divisória a partir do oitavo elemento de cada lista não ordenada.

52. Considere o seguinte pedaço de código html:

```
...
<h2>New York Yankees - Batting Line-up</h2>
<ol>
      <li>Jeter</li>
      <li>Damon</li>
      <li>Teixeira</li>
      <li>Posada</li>
      <li>Swisher</li>
```

```
        <li>Cano</li>
        <li>Cabrera</li>
        <li>Molina</li>
        <li>Ransom</li>
   </ol>
   ...
```

Escreva código *jQuery* que permita inverter a ordem dos elementos da lista.

53. Considere os seguintes dados em formato JSON num ficheiro intitulado "nomes.json":

```
{
      "names": [
            {
                  "first": "Azzie",
                  "last": "Zalenski",
                  "street": "9554 Niemann Crest",
                  "city": "Quinteros Divide",
                  "state": "VA",
                  "zip": "48786"
            },
            // and repeat for n names
      ]
}
```

   a. Escreva código *jquery* que permita carregar os dados guardados no ficheiro JSON e que os apresente através de uma tabela com duas colunas, que apresentem na primeira coluna o nome completo e na segunda a morada completa.
   b. Efetue a mesma coisa mas utilizando o objeto XMLHttpRequest (Ajax).

54. Considere o seguinte pedaço de código html:

```
(...)
<style type="text/css">
    .menu {
      background-color: #ccc;
      list-style: none;
      margin: 0;
      padding: 0;
      width: 10em;
    }
   .menu li {
      margin: 0;
       padding: 5px;
    }
   .menu a {
       color: #333;
   }
   .menuHover {
       background-color: #999;
   }
</style>
(...)

<body>
   <ul class="menu">
      <li><a href="download.html">Download</a></li>
      <li><a href="documentation.html">Documentation</a></li>
      <li><a href="tutorials.html">Tutorials</a></li>
   </ul>
(.....)
```

Escreva código *jquery* que permita, ao passar por cima de um item do menu, que este seja destacado com um fundo diferente (*menuHover*). Ao deixar o item, o fundo deve assumir a configuração de cor de fundo normal.

55. Considere o seguinte pedaço de código html que define um formulário com diversos tipos de elementos:

```
(...)
<style>
          div.question {
              padding: 1em;
          }
          div.errorMessage {
              display: none;
          }
          div.showErrorMessage {
              display: block;
              color: #f00;
              font-weight: bold;
              font-style: italic;
          }
          label.error {
              color: #f00;
              font-style: italic;
          }
 </style>
 </head>
    <body>
        <form action="#">
            <!-- TEXT -->
            <div class="question">
                <label for="t">Username</label>
                <input id="t" name="user" type="text" class="required" />
                <div id="errorMessage_user" class="errorMessage">Please
enter your username.</div>
            </div>
            <!-- PASSWORD -->
            <div class="question">
                <label for="p">Password</label>
                <input id="p" name="pass" type="password" class="required"
/>
                <div id="errorMessage_pass" class="errorMessage">Please
enter your password.</div>
            </div>
            <!-- SELECT ONE -->
            <div class="question">
                <label for="so">Favorite Color</label>
                <select id="so" name="color" class="required">
                    <option value="">Select a Color</option>
                    <option value="ff0000">Red</option>
                    <option value="00ff00">Green</option>
                    <option value="0000ff">Blue</option>
                </select>
                <div id="errorMessage_color" class="errorMessage">Please
select your favorite
                color.</div>
            </div>
            <!-- SELECT MULTIPLE -->
            <div class="question">
                <label for="sm">Favorite Foods</label>
                    <select id="sm" size="3" name="foods"
multiple="multiple" class="required">
```

```
                    <option value="pizza">Pizza</option>
                    <option value="burger">Burger</option>
                    <option value="salad">Salad</option>
                </select>
                <div id="errorMessage_foods" class="errorMessage">Please
choose your favorite
                foods.</div>
            </div>
            <!-- RADIO BUTTONS -->
            <div class="question">
                <span>Writing Hand:</span>
                <input id="r1" type="radio" name="hand" class="required"/>
                <label for="r1">Left</label>
                <input id="r2" type="radio" name="hand" class="required"
/>
                <label for="r2">Right</label>
                <div id="errorMessage_hand" class="errorMessage">Please
select what hand you
                write with.</div>
            </div>
            <!-- TEXTAREA -->
            <div class="question">
                <label for="tt">Comments</label>
                <textarea id="tt" name="comments"
class="required"></textarea>
                <div id="errorMessage_comments"
class="errorMessage">Please tell us what you
                think.</div>
            </div>
            <!-- CHECKBOX -->
            <div class="question">
                <input id="c" type="checkbox" name="legal"
class="required" />
                <label for="c">I agree with the terms and
conditions</label>
                <div id="errorMessage_legal" class="errorMessage">
                                    Please check the box!</div>
            </div>
            <input type="submit" value="Continue" />
        </form>
```

Escreva código *jquery* que permita, no momento de submeter o formulário, validar o conteúdos dos diversos componentes, sabendo que: os elementos do formulário do tipo *text*, *textarea* e *password* não podem estar vazios ou só com espaços; os elementos do tipo *select-one* e *select-multiple*, devem ter valor selecionado; e os *radio button* e *checkbox* têm de ter pelo menos uma opção selecionada. O aviso de verificação deve ser efetuado através das *div errorMessage* e das *label* associados a cada elemento do formulário. O CSS apresentado serve marcar e apresentar o erros. Sugere-se a utilização de um método genérico para a validação dos elementos do formulário.