Estratégia Concursos

Aula 00 - Prof. Felipe Mathias e Raphael Lacerda

TJ-RR (Analista Judiciário - Cibersegurança) Desenvolvimento de Software - 2024 (Pós-Edital)

Autor:

Felipe Mathias, Paolla Ramos

05 de Julho de 2024

Índice

APRESENTAÇÃO DA AULA

Olá, alunos! Bem-vindos a mais uma aula do curso de Tecnologia de Informação para concursos públicos, no Estratégia Concursos.

Me chamo Felipe Mathias e serei seu professor na aula de hoje. Sou um catarinense de 30 anos, programador *front end* (ex-programador, se preferirem haha) e atuo como professor de cursos de Tecnologia da Informação voltados a concursos há mais de um ano. Assim como você, também vivo a vida de concurseiro, aguardando minha nomeação como Auditor Fiscal da Secretaria de Fazenda de Minas Gerais (SEF-MG), onde figuro no cadastro de reserva. Atualmente, continuo, em paralelo, estudando para concursos aguardando o meu grande sonho – o cargo de Auditor Fiscal da SEF-SC, com especialidade em TI.

Minha aventura no mundo do ensino surgiu de uma vontade interna de atuar como professor – sempre amei explicar as coisas, além de ter certa facilidade em expressar conceitos mais complexos para pessoas que talvez não tenham tanta experiência na área.

Meu objetivo aqui é digerir assuntos, desde os mais simples aos mais complexos, para que qualquer aluno consiga os entender, seja um programador, operador de infraestrutura, ou simplesmente um leigo que resolveu adentrar no mundo dos concursos e se deparou com TI no seu edital.

Gostaria de pedir que **sempre** vejam as questões comentadas durante a aula. Elas trazem conteúdo essencial para o aprendizado, muitas vezes abordando alguns pontos que não foram abordados no conteúdo e são essenciais para a resolução de questões.

Caso tenha alguma dúvida, não tenha receio de entrar em contato comigo nas minhas redes sociais (especialmente no meu Instagram, que deixarei abaixo), ou no fórum de dúvidas que os responderei assim que possível.

Ah, posto bastante coisa interessante de TI direcionada para concursos lá, dá uma olhadinha que algumas coisas podem te interessar. Volta e meio acerto alguma questão de prova por lá ;)

@fe.fiscal

t.me/fefiscal

JSON

Conceitos Gerais

JSON, ou JavaScript Object Notation, é um formato de arquivo em texto destinado a carregar dados através de comunicações entre sistemas. Apesar de contar com JavaScript no nome, o JSON é um formato de arquivo que independe da linguagem implementada, sendo usado e aceito pela maior parte das linguagens de programação.

O nome "JavaScript Object Notation" indica a forma de estruturação do arquivo. Sua sintaxe é completamente baseada na sintaxe dos objetos de JavaScript. Portanto, temos pares de atributo e valor, separados por vírgulas e envoltos em chaves {}. Então, a sintaxe geral de um arquivo JSON se dá da seguinte forma:

```
{
  atributo1:valor1,
  atributo2:valor2,
  ...
  atributo_n: valor_n
}
```

QUESTÃO DE PROVA

(CEBRASPE/CNPq/2024) No que se refere a serviços de integração, julgue o item que se segue.

A sequência a seguir é uma lista que representa um objeto no formato JSON.

{atributo1:valor1 , atributo2:valor 2}

Comentários:

Arquivos JSON são conjuntos de dados em atributo:valor, envoltos de chaves {}. (Gabarito: Certo)

Alguns atributos dos arquivos JSON o levam a ser o formato preponderante na troca de dados na internet nos dias atuais, principalmente se comparados com um outro formato bem adotado - o XML. Veja:

- Legibilidade: maior legibilidade para humanos
- Baixa verbosidade: a ausência de tags para cada elemento o torna um arquivo menos verboso e, consequentemente, mais leve
- Velocidade de processamento: o JSON tem uma velocidade de processamento maior, se comparado aos seus pares, como o XML
- Compatibilidade: a maior parte das APIs modernas suporta o formato JSON
- Integração com JavaScript: por ser nativo do JavaScript, vários métodos da linguagem fazem uma integração completa com o formato

Abaixo, vou deixar um exemplo de arquivo JSON - e você perceberá como é tranquila a leitura.

```
{
  "nome": "João Silva",
  "idade": 30,
  "sexo": "Masculino",
  "email": "joao.silva@example.com",
  "telefone": "+55 11 98765-4321",
  "endereco": {
    "rua": "Rua das Flores",
    "numero": 123,
    "bairro": "Centro",
    "cidade": "São Paulo",
    "estado": "SP",
    "cep": "01234-567"
  },
  "interesses": ["Viajar", "Tecnologia", "Esportes"],
  "educacao": [
    {
      "instituicao": "Universidade XYZ",
      "curso": "Engenharia de Software",
      "ano_conclusao": 2015
    },
    {
      "instituicao": "Escola ABC",
      "curso": "Ensino Médio",
      "ano_conclusao": 2010
    }
  ]
}
```

Sintaxe

A sintaxe dos arquivos JSON são bem simples:

- Dado são armazenado em pares de chave:valor
- Cada dado deve ser separado por vírgula
- Pares de colchetes indicam *arrays*
- Pares de chaves indicam um objeto

Ao contrário dos objetos em JavaScript, nos arquivos JSON precisamos ter as chaves, que definem os valores, como *strings* - portanto, envoltas em aspas duplas `“ “`. Então, enquanto uma linha de código de objeto em JavaScript pode ser escrita como `nome: “Felipe Mathias”`, em JSON ela deve ser escrita como `“nome”: “Felipe Mathias”`.

Já para os valores, temos algumas definições diferentes. Podemos ter os seguintes tipos de dados, que seguem a mesma padronização de formato que o JavaScript:

- Texto (*string*). Ex.: {“nome”: “Estratégia }
- Número (inteiro ou de ponto flutuante). Ex.: { “nota”: 7.39 }
- Objeto. Ex.: { {“nome”: “Desenvolvimento de Sistemas”, “Matéria”: “TI” } }
- Array. Ex.: { “professores”: [“Felipe”, “Paolla”] }
- Boolean. Ex.: { “aprovado” : true }
- Null. Ex.: { “pendencias”: null }

QUESTÃO DE PROVA

Em se tratando da persistência de dados, os bancos de dados orientados a documentos se mostram adequados para representar e armazenar dados que possuem características comuns, mas que também possuem características distintas entre si. Um dos formatos muito utilizados para representação de dados em um banco de dados orientados a documentos é o formato Javascript Object Notation (JSON).

(CESGRANRIO/IPEA/2024) Nesse contexto, considere o exemplo em JSON apresentado a seguir, que representa dados de um livro da área de banco de dados:

{

```
  "isbn": "9788543025001",
  "autor": [
    {
      "ultimonome": "Elmasri",
      "primeironome": "Ramez"
    },
    {
      "ultimonome": "Navathe",
      "primeironome": "Shamkant"
    }
  ],
  "titulo": "Sistemas de Bancos de Dados",
  "categoria": ["BD","SGBDR","SQL"]
}
```

No exemplo apresentado, observa-se que

a) o livro possui dois vetores, ou arrays.
b) o livro tem cinco pares de nome e valor.
c) o campo "categoria" tem três pares de nome e valor.
d) o campo de nome "9788543025001" tem o valor "isbn".
e) o campo de nome "autor" é formado por um vetor, ou array, de quatro elementos.

Comentários:

Vamos analisar cada alternativa.

a) Certo. Temos duas arrays, caracterizadas pelos colchetes [] - uma para “autor”, e outra para “titulo”.
b) Errado. Temos 4 pares - isbn, autor, titulo e categoria.
c) Errado. O campo “categoria” é uma array, contendo 3 valores distintos.
d) Errado. O campo de nome “isbn” que tem o valor numérico, e não o contrário.
e) Errado. Ele é formado por uma array de 2 elementos, e cada elemento desses é um objeto composto de dois pares de chave-valor.

Portanto, correta a letra A. (Gabarito: Letra A)

Parse

O Parse do JSON, ou, numa tradução literal, sua "tradução", implica em convertermos o formato de texto para um formato legível e utilizável pelas linguagens. Especificamente para o JavaScript, como o JSON é um formato de arquivo nativo, temos uma função completamente integrada que permite tanto transformar JSON em objetos, quanto o caminho oposto.

Para isso, usamos a função **`JSON.parse()`**, em JavaScript. Por exemplo, imagine o seguinte conjunto de dados em JSON:

```
{
  "nome": "Felipe Mathias"
  "profissão": "Professor"
  "idade": 30
}
```

Se quisermos manipular esses dados em JavaScript, selecionando somente uma chave, por exemplo, podemos fazer da seguinte forma:

jQuery

```
const jsonObj = JSON.parse('{"nome": "Felipe Mathias", "profissão":
"Professor", "idade": 30}');

console.log("Nome: ", jsonObj["nome"]);
```

```
Nome: Felipe Mathias
```

Agora pense essa fórmula aplicada a uma requisição HTTP, retornando um conjunto de dados e permitindo que você navegue através desse arquivo JSON com métodos nativos do JavaScript, criando um código extremamente adaptável e de baixo acoplamento.

INDO MAIS FUNDO!

Também é possível que façamos o caminho oposto. A partir de um objeto qualquer, criarmos uma *string* no formato de um arquivo JSON com a função JSON.stringify(). Vamos fazer o mesmo que fizemos anteriormente, mas agora no caminho oposto - começaremos com um objeto qualquer e o converteremos em um arquivo JSON.

jQUery

```
const jsonObj = {
    nome: "Felipe Mathias",
    profissao: "Professor",
    idade: 30
}

let jsonString = JSON.stringify(jsonObj)

console.log("O tipo do dado é :", typeof jsonString)
console.log(jsonString)
```

```
O tipo do dado é : string
{"nome":"Felipe Mathias","profissao":"Professor","idade":30}
```

Por fim, podemos fazer uso da função eval, do JavaScript, para converter um texto JSON em um objeto no JavaScript, principalmente em navegadores que não suportam o JSON.parse. A função `eval()` em JavaScript é usada para avaliar ou executar uma string como código JavaScript. Nesse caso, criamos uma string com base no JSON, e a transformamos em um objeto com a função `eval()`. O código ficará da seguinte forma:

jQUery

```
// JSON em formato de string
const jsonString = '{"nome": "Felipe Mathias", "profissão":
"Professor", "idade": 30}';

// Usando eval para converter JSON em um objeto JavaScript
const jsonObject = eval('(' + jsonString + ')');

console.log(jsonObject.nome);
```

```
Felipe Mathias
```

(CEBRASPE/TCE SC/2016) Julgue o item que se segue a respeito dos padrões XSLT e JSON.

Em navegadores que não possuem apoio para a função JavaScript JSON.parse, pode-se utilizar a função eval para converter um texto JSON em um objeto JavaScript, por meio da sintaxe apresentada a seguir.
var obj = eval ("(" + text + ")");

Comentários:

Perfeito, aluno. O eval() é a uma das formas de transformarmos um arquivo JSON, que é interpretado como uma string, em um objeto dentro do JavaScript, permitindo a aplicação de métodos de manipulação. (Gabarito: Certo)

(Inédita/Prof. Felipe Mathias) Assinale a alternativa que aponta o comando a ser utilizado para converter um arquivo JSON em um objeto que pode ser manipulado pelo JavaScript:

a) JSON.purge
b) JSON.pause
c) JSON.convert
d) JSON.parse
e) JSON.object

Comnetários:

Ao recebermos um arquivo JSON, que estará em formato de texto (string), é necessário transformá-lo em objeto para corretamente manipular suas propriedades, acessar suas chaves, entre outros. Para isso, usa-se o JSON.parse. (Gabarito: Letra D)

Dados tabulares

Muitas vezes utilizamos consultas em SQL para criarmos os arquivos JSON. Precisamos tratar os dados retornados, que são tabulares, em um arquivo JSON para ser utilizado pelos códigos. O JSON não traz nenhuma forma de especificação de estrutura para dados tabulares e, portanto, usualmente tratamos esses dados com um conjunto de arrays, sendo:

- 1 Array para cada atributo (coluna)
- 1 Array composto de outras arrays, representando cada linha isoladamente

Pode ter parecido confuso, então vou lhe trazer um exemplo.

Imagine a seguinte tabela:

ID	Nome	Idade	Pais
1	João	30	Brasil
2	Maria	25	EUA
3	Carlos	40	Espanha

Representando esses dados em JSON, teríamos a seguinte estrutura:

```
{
  "dados": {
    "colunas": ["ID", "Nome", "Idade", "País"],
    "linhas": [
      [1, "João", 30, "Brasil"],
      [2, "Maria", 25, "EUA"],
      [3, "Carlos", 40, "Espanha"]
    ]
  }
}
```

Comentários

Comentários na grande maioria das linguagens de programação são pequenas linhas de instrução, destinadas ao desenvolvedor que irá ler o código, e não ao compilador - e, por esse motivo, elas não são interpretadas. Porém, o JSON não é uma linguagem de programação, e sim uma simples linguagem em texto para transporte de dados.

Por esse motivo, os comentários em JSON não são aceitos. Se inserido algum comentário, usualmente por uma simbologia de destaque (como uma cerquilha `#`), o compilador irá os interpretar como os dados, eles não serão ignorados.

A alternativa mais recomendada é a criação de um par de chave-valor para servir de comentário. Então, a cada novo elemento, podemos criar uma variável por exemplo `“_comentario”: “linha de comentario”`.

QUESTÃO DE PROVA

(CEBRASPE/PREF. FORTALEZA/2023) Acerca de HTTPRequest e JSON, julgue o próximo item.

O processo de adição de comentários em arquivos JSON é igual ao das linguagens de programação, pois, assim como estes, tipicamente ignora comentários quando são executados.

Comentários:

O JSON não suporta comentários “tradicionais”. Uma forma de sobrepor isso é criando pares de chave-valor que são facilmente identificados como comentários - mas, apesar disso, essa linha ainda será interpretada pelo compilador. Dessa forma, a alternativa está incorreta. (Gabarito: Errado)

JSON Schema

Os arquivos JSON não possuem um formato pré-definido, por isso são categorizados nos dados de tipo semiestruturado. Porém, para determinados sistemas, pode ser necessário que um padrão seja seguido, garantindo o correto funcionamento do arquivo.

Para fazer uma validação do esquema, utilizamos o JSON Schema. O arquivo JSON recebido (ou a ser enviado) é passado por um validador, que utiliza o esquema definido para verificar se o arquivo está de acordo com as diretrizes. Esse esquema tem diversos padrões - adotaremos o padrão do json-schema.org para essa aula.

O arquivo Schema parece-se com isso:

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "Book",
  "type": "object",
  "properties": {
    "title": {
      "type": "string",
      "description": "The title of the book"
    },
    "author": {
      "type": "string",
      "description": "The author of the book"
    },
    "published_year": {
      "type": "integer",
      "minimum": 1900,
      "maximum": 2024,
      "description": "The year the book was published"
    },
    "isbn": {
      "type": "string",
      "pattern": "^\\d{3}-\\d{10}$",
      "description": "The ISBN of the book in the format xxx-xxxxxxxxxx"
    }
  },
  "required": ["title", "author", "published_year"],
  "additionalProperties": false
}
```

- $schema: Especifica a versão do JSON Schema utilizada.
- title: Define o título do esquema.
- type: Indica que o objeto JSON esperado deve ser do tipo "object".
- properties: Define as propriedades esperadas no objeto JSON, como "title", "author", "published_year" e "isbn".

- Para cada propriedade, são especificados o tipo (type), uma descrição (description) e, quando aplicável, restrições adicionais, como o ano mínimo e máximo de publicação para "published_year" e um padrão de regex para "isbn".
- required: Lista as propriedades que são obrigatórias no objeto JSON.
- additionalProperties: Define se propriedades adicionais não especificadas no esquema são permitidas (true) ou não (false). Neste caso, propriedades adicionais não são permitidas.

QUESTÃO DE PROVA

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as tecnologias Web, julgue o item abaixo.

É possível criar uma estrutura; ao mais forte aos arquivos JSON, exigindo sua validação, a partir do uso do Esquema JSON, ou JSON Schema.

Comentários:

Certinho! O JSON Schema é responsável por definir alguns padrões para o arquivo JSON, como tipo de dado, tamanho, tipo de informação e campos obrigatórios. (Gabarito: Certo)

RESUMO

O QUE É O JSON?

JSON, ou JavaScript Object Notation, é um arquivo de texto destinado a carregar informações e dados entre sistemas distintos. Ele é um arquivo independente de implementação, aceitado em diversas linguagens, apesar de ser nativo do JavaScript.

COMO É A NOTAÇÃO DO JSON?

No JSON, temos uma notação baseada nos objetos do JavaScript, com algumas alterações. Por ser uma string, envolvemos todos os dados num par de chaves {}. Os dados, por sua vez, são representados em pares de "chaves":"valor. Então, por exemplo, podemos ter a notação {"aula":"JSON"}.

QUESTÕES COMENTADAS

01. (CESGRANRIO/IPEA/2024) Um programador de páginas web escreveu o arquivo XML abaixo no formato JSON, para que usuários internos do Ipea possam gerenciar o acesso aos dados públicos das páginas web.

```
<menu id=''file'' value=''File''>
  <popup>
    <menuitem value=''Localizar'' onclick=''Search()'' />
    <menuitem value=''Alterar'' onclick=''Alter()'' />
    <menuitem value=''Incluir'' onclick=''Include()'' />
    <menuitem value=''Apagar'' onclick=''Delete()'' />
   < /popup>
</menu>
```

Após analisar o arquivo acima, o gerente da área corrigiu a sua sintaxe, obtendo o seguinte arquivo, no formato JSON:

a)
```
{''menu'': {
  ''id'': ''file'',
  ''value'': ''File'',
  ''popup'': {
 [
    {'' menuitem '': ''Localizar'', ''onclick'': ''Search()''},
    {'' menuitem '': ''Alterar'', ''onclick'': ''Alter()''},
    {'' menuitem '': ''Incluir'', ''onclick'': ''Include()''}
    {'' menuitem '': ''Apagar'', ''onclick'': ''Delete()''}
  ]
}}
```

b)
```
{''menu'': {BEGIN
  {''id'': ''file'',''value'': ''File''}
  ''popup'': {
   ''menuitem'': [
     {''Localizar'', ''onclick'': ''Search()''},
     {''Alterar'', ''onclick'': ''Alter()''},
     {''Incluir'', ''onclick'': ''Include()''}
     {''Apagar'', ''onclick'': ''Delete()''}
]
}}END}
```

c)
```
{BEGIN ''menu''
  ''id'': ''file'',
```

```
  ''value'': ''File'',
  ''popup'': {
    ''menuitem'': [
      {''value'': ''Localizar'', ''Search()''},
      {''value'': ''Alterar'', ''Alter()''},
      {''value'': ''Incluir'', ''Include()''}
      {''value'': ''Apagar'', ''Delete()''}
  ]
}
END}
```

d)

```
{''menu'': {
  ''id'': ''file'',
  ''value'': ''File'',
  ''popup'': {
    ''menuitem'': [
      {''value'': ''Localizar'', ''onclick'': ''Search()''},
      {''value'': ''Alterar'', ''onclick'': ''Alter()''},
      {''value'': ''Incluir'', ''onclick'': ''Include()''}
      {''value'': ''Apagar'', ''onclick'': ''Delete()''}
    ]
  }
}}
```

e)

```
{''menu'': {
  ''id'': ''file'',
  ''value'': ''File'',
  ''popup'': {
    ''menuitem'': [
      {''item'' ''Localizar'', ''Search()''},
      {''item'' ''Alterar'', ''Alter()''},
      {''item'' ''Incluir'', ''Include()''}
      {''item'' ''Apagar'', ''Delete()''}
    ]
  }
}}
```

Comentários:

Questão complexa. Vamos fazer a conversão por partes, para entendermos o que acontecerá.

```
<menu id=''file'' value=''File''>
```

Esse trecho aponta que temos uma *tag* de nome menu, com id file e valor File. Podemos traduzir a sintaxe para um objeto, chamado de “menu”.

{ “menu”: {
“id”:”file”,
“value”:”file”,

Vamos à segunda parte.

```
<popup>
   <menuitem value=''Localizar'' onclick=''Search()'' />
   <menuitem value=''Alterar'' onclick=''Alter()'' />
   <menuitem value=''Incluir'' onclick=''Include()'' />
   <menuitem value=''Apagar'' onclick=''Delete()'' />
< /popup>
```

Temos um *popup* com diversos itens de menu, cada um com seu efeito *onclick* e seu valor. Vamos representar o popup como um objeto, e cada menuitem como um item de uma array de objetos:

```
{“popup”: {
  “menuitem”: [
    {“value” : “Localizar”, “onclick” : “Search()”}
    {“value” : “Alterar”, “onclick” : “Alter()”}
    {“value” : “Incluir”, “onclick” : “Include()”}
    {“value” : “Apagar”, “onclick” : “Delete()”}
```

Juntando as duas sintaxes, chegamos à seguinte opção:

```
{''menu'': {
  ''id'': ''file'',
  ''value'': ''File'',
  ''popup'': {
   ''menuitem'': [
    {''value'': ''Localizar'', ''onclick'': ''Search()''},
    {''value'': ''Alterar'', ''onclick'': ''Alter()''},
    {''value'': ''Incluir'', ''onclick'': ''Include()''},
    {''value'': ''Apagar'', ''onclick'': ''Delete()''} ]}}}
```

Gabarito: Letra D

---

02. (CEBRASPE/ITAIPU/2024) Assinale a opção que apresenta a sintaxe correta de representação de um array de nomes e sobrenomes de duas pessoas na linguagem JSON.

a) { { "nome": "JOAO", "sobrenome": "SILVA" } , { "nome": "MARIA", "sobrenome": "SOARES" } }
b) [{ "nome": "JOAO", "sobrenome": "SILVA" } { "nome": "MARIA", "sobrenome": "SOARES" }]
c) { "nome": "JOAO", "sobrenome": "SILVA", "nome": "MARIA", "sobrenome": "SOARES" }
d) [{ "nome": "JOAO", "sobrenome": "SILVA" } , { "nome": "MARIA", "sobrenome": "SOARES" }]

e) { ("nome": "JOAO", "sobrenome": "SILVA") , ("nome": "MARIA", "sobrenome": "SOARES") }

Comentários:

A representação de arrays em JSON é feita a partir de um par de colchetes []. Podemos representar os nomes, portanto, como:

{ “nomes”:
[{“nome”: “nome1”, “sobrenome” : “sobrenome1”},
{“nome”: “nome 2”, “sobrenome” : “sobrenome2”}]
}

Portanto, a alternativa correta é a letra D.

Gabarito: Letra D

---

03. (CEBRASPE/SEPLAN RR/2023) Com relação a programação e desenvolvimento de sistemas, julgue o item a seguir.

A seguinte notação em JSON representa corretamente a propriedade de “Nome” para as empresas A e B.

{“Empresas”:[{“Nome”:”A”},{“Nome”:”B”}]}

Comentários:

Quando vamos elencar atributos compostos e de “mesmo nível”, como é o caso das empresas, podemos fazer isso através de uma array, representada pelos colchetes []. Para cada empresa criaremos um objeto dentro dessa array, que receberá suas características. E é exatamente isso que a questão fez - portanto, correta.

Gabarito: Correto

---

04. (FGV/TCE ES/2023) Pedro está desenvolvendo um programa que manipula dados do tipo Arrays (Vetores) no formato JSON.

Para representar dados desse tipo, Pedro deve usar a sintaxe:

a) {"venda":true}
b) {"nome":"John"}
c) {"idade":null}
d) {"cadastro":["Joao", "Ana", "Pedro"]}
e) {"pessoa":{"nome":"Joao", "idade":30, "estado":"ES"}}

Comentários:

As arrays em JSON são representadas a partir de um par de colchetes [ ]. A alternativa que apresenta a sintaxe é a letra D - que cria uma array para a chave "cadastro".

Gabarito: Letra D

05. (IADES/SEPLAD DF/2023) Qual é o caractere que delimita um array em uma estrutura JSON?

a) Parênteses
b) Colchetes
c) Chaves
d) Asteriscos
e) Cifrões

Comentários:

Você já deve ter percebido que a cobrança vem pesada nas *arrays* em JSON, né? Lembre-se, as arrays são representadas por pares de colchetes [].

Gabarito: Letra B

06. (VUNESP/TCM SP/2023) A alternativa que contém uma representação inválida de um objeto JSON, segundo o padrão ECMA-404, é:

a) {"op": "test", "path": "/a/b/c", "value": "foo"}
b) {"op": "replace", "path": "/a/b/c", "value": 42}
c) {"op": "add", "path": "/a/b/c", "value": ["foo", "bar"]}
d) {"op": "move", "from": "/a/b/c", "path": "/a/b/d"}
e) {"op": "copy", "path", "/a/b/c", "value": "foo"}

Comentários:
As sintaxes, com exceção da letra E, estão todas corretas. Na letra D, o correto seria “path” : “/a/b/c”, e não o uso de vírgula, como foi feito.

Gabarito: Letra E

07. (FUMARC/ALMG/2023) Analise as afirmativas a seguir referentes ao formato JSON:

I – JSON é um acrônimo de JavaScript Object Notation.
II – JSON é um formato para intercâmbio de dados.
III – Um objeto JSON começa com uma “{“ e termina com uma “}”.

Estão CORRETAS as afirmativas:

a) I e II, apenas.
b) I e III, apenas.
c) II e III, apenas.
d) I, II e III.

Comentários:

Vamos analisar os itens.

I. Certo. Esse é o significado de JSON.
II. Certo. O JSON é destinado a intercâmbio de dados entre sistemas.
III. Certo. Identificamos arquivos JSON pela presença de pares de chaves {}.

Portanto, todos os itens são corretos.

Gabarito: Letra D

---

08. (FCC/MPE PB/2023) Considere as propriedades abaixo, seus valores e tipos de dados.

- id, tipo numérico, contendo o valor 432.
- nome, cadeia de caracteres, contendo o valor Paulo.
- cargo, cadeia de caracteres, contendo o valor Promotor.

A forma adequada de criar um objeto JSON contendo estes valores é

a) ["id":432, "nome":"Paulo", "cargo":"Promotor"]
b) {number["id":432], String["nome":"Paulo", "cargo":"Promotor"]}
c) {id[number]:432, nome[string]:"Paulo", cargo[string]:"Promotor"}
d) [int id=432, String nome="Paulo", String cargo="Promotor"]
e) {"id":432, "nome":"Paulo", "cargo":"Promotor"}

Comentários:

Vamos converter cada linha.

- id, tipo numérico, contendo o valor 432.
{“id” : 432}

- nome, cadeia de caracteres, contendo o valor Paulo.
{“nome” : “Paulo”}

- cargo, cadeia de caracteres, contendo o valor Promotor.
{“cargo” : “Promotor”}

A alternativa que, corretamente, traz todas as conversões é a letra E.

Gabarito: Letra D

---

09. (FCC/MPE PB/2023) Considere o fragmento JavaScript abaixo.

<script>
const json_adv = '___I___';
const obj_adv = JSON.parse(json_adv);
</script>

Na lacuna I, a representação correta para um objeto JSON contendo a propriedade nome com o valor Paulo e um array processos contendo os processos 0001842672017 e 0045613912014 é

a) {"nome":"Paulo", "processos[0]":"0001842672017", "processos[1]":"0045613912014"]}
b) {nome="Paulo", processos=["0001842672017", "0045613912014"]}
c) {"nome":"Paulo", "processos":["0001842672017", "0045613912014"]}
d) nome="Paulo", processos{"0001842672017", "0045613912014"}
e) ["nome":"Paulo", "processos":{"0001842672017", "0045613912014"}]

Comentários:

Podemos representar o nome como {"nome":"Paulo"}, e a *string* como "processos": ["00018...", "00456..."]. A alternativa que corretamente faz essas conversões é a letra C.

Gabarito: Letra D

---

10. (FUNDATEC/BRDE/2023) Sobre o JSON (JavaScript Object Notation), assinale a alternativa que apresenta um tipo de dado não suportado.

a) date
b) string
c) number
d) boolean
e) array

Comentários:

Dos tipos apresentados, aquele não suportado pelo JSON está na letra A - o formato DATE. Temos um conjunto de tipos de dado usados no JavaScript que não são suportados pelo JSON:

- Date
- Função
- Undefined

Portanto, nosso gabarito é a letra A.

Gabarito: Letra A

---

11. (FEPESE/PREF. B. CAMBORIÚ/2023) São tipos de dados JSON válidos:

1. Null
2. Data
3. Array
4. Undefined
5. Booleano

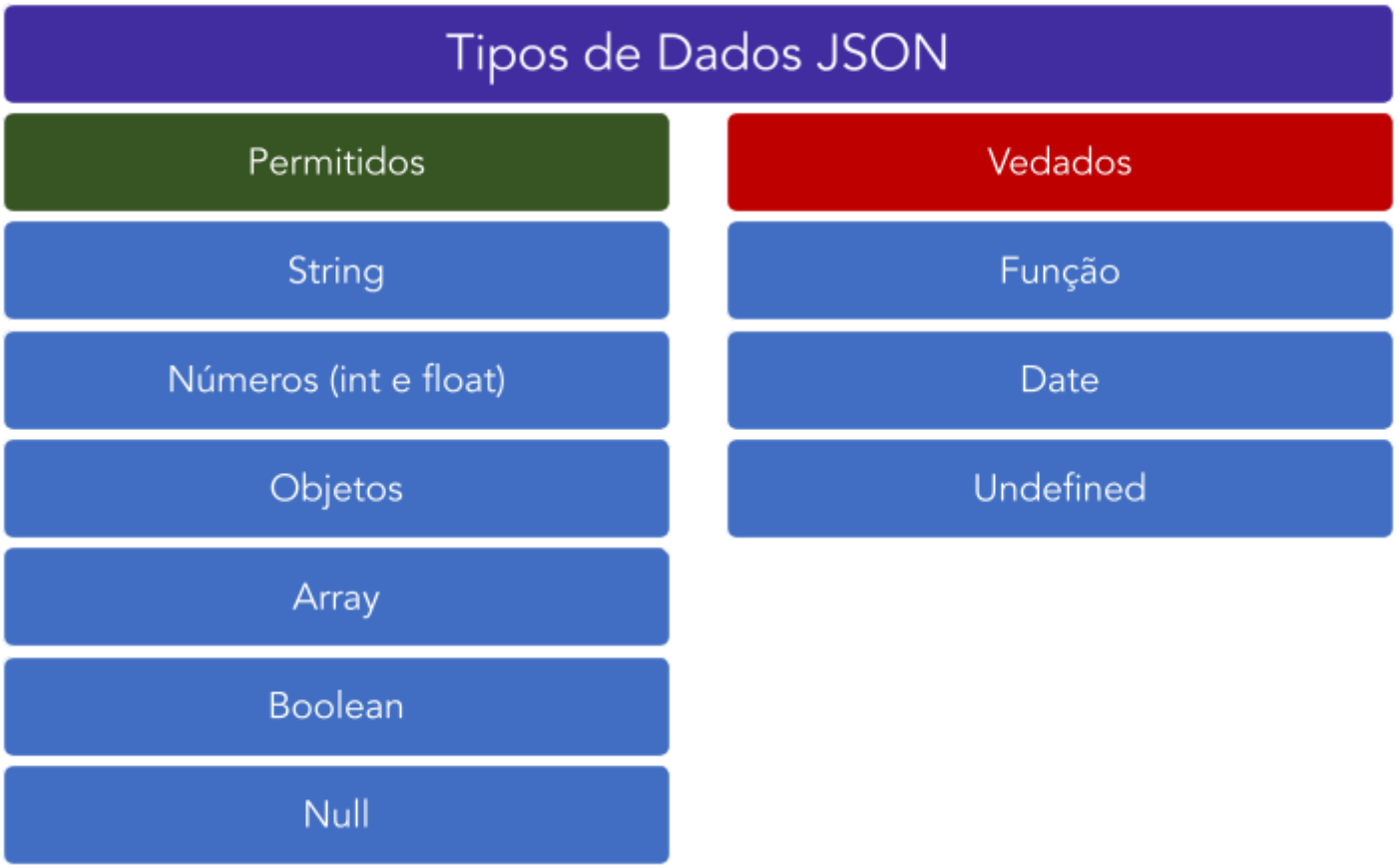
Assinale a alternativa que indica todas as afirmativas corretas.

a) São corretas apenas as afirmativas 2 e 4.
b) São corretas apenas as afirmativas 3 e 5.

c) São corretas apenas as afirmativas 1, 2 e 3.
d) São corretas apenas as afirmativas 1, 3 e 5.
e) São corretas apenas as afirmativas 3, 4 e 5.

Comentários:

Vamos listar os tipos de dados permitidos e vedados?



Analisando a lista, os tipos de dados permitidos são o 1, 3 e 5, enquanto 2 e 4 são vedados.

Gabarito: Letra D

---

12. (FUNDATEC/PROCERGS/2023) O formato JSON (JavaScript Object Notation) surgiu em 2000 como uma alternativa ao formato XML (eXtensible Markup Language). Apresenta-se como um formato de intercâmbio de dados mais leve e com maior facilidade de interpretação para uma boa comunicação de aplicações web. Sendo assim, são características do formato JSON:

I. Suporta uma variedade de tipos de dados, tais como números, strings, booleanos, arrays e objetos.

II. É um formato dependente, sendo utilizado em aplicações web apenas com a linguagem de programação Javascript.
III. É um formato mais leve que o XML, o que significa que pode ser transmitido com maior rapidez.
IV. Tem uma estrutura definida com tags que delimitam os elementos de dados e atributos que fornecem informações adicionais sobre esses elementos.

Quais estão INCORRETAS?

a) Apenas I.
b) Apenas III.
c) Apenas II e IV.
d) Apenas I, II, III.
e) I, II, III e IV.

Comentários:

Vamos às alternativas.

I. Certo. Temos diversos tipos diferentes de dados suportados.
II. Errado. O JSON é independente de implementação.
III. Certo. O JSON, por ser menos verboso, é mais leve e trafega de forma mais rápida que o XML.
IV. Errado. A estrutura do JSON não é definida.

Portanto, temos II e IV como incorretas.

Gabarito: Letra C

---

13. (VUNESP/TJ RS/2023) Uma API RESTful de um sistema de uma loja de livros, quando acessada pela requisição

GET /api/v1/livros

retorna a seguinte resposta no formato JSON:

```
{
  "meta : {
  },
  "data : [{
    "id : 20,
    "title : Mensagem ,
    "author : Fernando Pessoa
  }, {
    "id : 21,
    "title : Alguma Poesia ,
```

```
    "author : Carlos Drummond de Andrade
  }]
}
```

Com base nessas informações, é correto afirmar que

a) meta é um array vazio.
b) o conteúdo retornado está sintaticamente incorreto de acordo com o padrão ECMA-404, pois faltam aspas nos valores de id.
c) o array data possui 2 elementos.
d) o array data possui 6 elementos.
e) o código de status HTTP retornado como resposta é 302 – Found.

Comentários:

Vamos analisar cada alternativa.

a) Errado. A chave "meta" é um objeto, não uma array.
b) Errado. Não há necessidade de aspas para especificar valores.
c) Certo. O array "data" é composto de 2 objetos.
d) Errado. Cuidado para não confundir - o array tem 2 elementos, 2 objetos.
e) Errado. Um pouco fora do escopo da questão, mas 302 é um código de erro, para página temporariamente mudada.

Gabarito: Letra C

---

14. (CEBRASPE/MPE RO/2023) Em ambiente web, o padrão que é utilizado para a troca de informações entre sistemas e que apresenta estrutura composta por elementos do tipo chaves, dois pontos, colchetes e aspas é o

a) SOAP.
b) XML.
c) JSON.
d) XSLT.
e) WSDL.

Comentários:

O formato de arquivo com uma estrutura composta por um par de chaves é o JSON.`

Gabarito: Letra C

---

15. (VUNESP/UFABC/2023) Na notação JSON, de acordo com a especificação ECMA 404, a sequência \u000d, quando usada dentro de uma string, representa o caractere

a) backspace.
b) barra normal.

c) barra invertida.
d) retorno de carro (carriage return).
e) alimentação de linha (line feed).

Comentários:

Questão bem avançada. O caractere \u000d indica o caractere *retorno de carro (carriage return)*. Esse caractere indica um fim de linha de texto para os sistemas operacionais.

Gabarito: Letra D

---

16. (FUVEST/USP/2023) Observe a notação JSON apresentada.

```
[
  {"year": 2024},
  {"location": "unknown"}
]
```

Que tipo de dados está representado pelo componente mais externo da notação?

a) Object.
b) Array.
c) Number.
d) String.
e) Dictionary.

Comentários:

O componente mais externo é um par de colchetes - que indica um Array.

Gabarito: Letra B

---

17. (CEBRASPE/TC DF/2023) A respeito de interoperabilidade de sistemas, DevOps e configuração de software, julgue o item que se segue.

Na notação JSON, cada objeto é representado por uma dupla de nome e valor de propriedade, agrupados entre chaves.

Comentários:

Perfeito! Objetos em JSON são representados por pares de chaves - por exemplo: {"valores": {32, 34, 45, 92} }.

Gabarito: Correto

---

18. (FGV/TJ SE/2023) JSON é um formato de texto autodescritivo utilizado para armazenar e transportar dados escritos em pares chave-valor. Para escrever um par chave-valor JSON, deve-se usar o formato:

a) nome;Maria
b) (nome:"Maria")
c) {nome:Maria}
d) [nome:'Maria']
e) "nome":"Maria"

Comentários:

Para escrever um par de chave-valor em JSON, utilizamos "chave":"valor". Como a questão quer o par nome e Maria, escreveremos "nome":"Maria".

Gabarito: Letra E

---

19. (FGV/TJ SE/2023) A analista Deise recebeu a tarefa de mapear uma tabela de banco de dados para um arquivo no formato JavaScript Object Notation (JSON). Após analisar a estrutura da tabela, Deise concluiu que a melhor estratégia de mapeamento consiste em associar cada valor da tabela a um dos três nomes literais previstos como valores válidos na notação JSON. Ao executar essa estratégia, Deise deve associar cada valor da tabela a um dos nomes literais:

a) true, false, null;
b) True, False, Null;
c) True, False, Undefined;
d) infinite, nan, undefined;
e) Infinite, NaN, Undefined.

Comentários:

O JSON aceita três valores literais: os valores booleanos (true e false) e null. Esses valores devem ser escritos em caixa baixa (minúsculas). A alternativa que traz corretamente esses três valores é a letra A.

Gabarito: Letra A

---

20. (FGV/TJ SE/2023) A técnica Joana é responsável pela manutenção do arquivo processos.json, que possui o seguinte conteúdo no formato JavaScript Object Notation (JSON):

```
{ "processosPorMunicipio": [
    {"nome": "Pedrinhas", "qtd": 33144},
    {"nome": "Telha", "qtd": 48958}]}
```

Joana deve adicionar ao array processosPorMunicipio um novo objeto, relacionado ao Município de Riachuelo, com os atributos nome e qtd. Porém, Joana ainda não dispõe da quantidade de processos de Riachuelo. Para adicionar o novo objeto sem omitir atributos, Joana deve utilizar o valor previsto na notação JSON para ausência de informação.

Logo, Joana deve definir o valor do atributo qtd do novo objeto como:

a) 0;
b) null;
c) NaN;
d) zero;
e) undefined.

Comentários:

Como Joana não tem uma quantidade definida, a técnica deve adicionar um valor literal indicando o conjunto vazio - esse valor é representado pelo atributo null.

Gabarito: Letra B

---

21. (CEBRASPE/DP DF/2022) Julgue o item a seguir, acerca de CSS3, JMS, JSON e JUnit.

Datas devem ser escritas como strings para que sejam utilizadas em JSON.

Comentários:

Perfeito! Como não temos o formato DATE no JSON, é recomendado que as datas sejam escritas como conjuntos de caracteres (strings).

Gabarito: Correto

---

22. (CEBRASPE/DP DF/2022) Julgue o item a seguir, relativo a WSDL, JSON, XML e XSLT.

A estrutura a seguir descreve corretamente, na notação JSON, um veículo da marca ABC, modelo G5 e ano 2019. [marca:"ABC";modelo:"G5";ano:"2019"]

Comentários:

A array construída possui dois erros - primeiro ponto que deveria ser um objeto, e não uma array; o segundo é a ausência das aspas demarcando as chaves.

Gabarito: Errado

---

23. (FGV/PC AM/2022) Sobre as regras sintáticas do JSON (JavaScript Object Notation), avalie as seguintes afirmativas:

I. Dados são codificados como pares name/value.
II. Dados são separados por barras verticais "|".
III. Chaves { } delimitam objetos.
IV. Colchetes [ ] delimitam arrays.

Está correto somente o que é afirmado em

a) I e II.
b) II e III.
c) III e IV.

d) I, II e III.
e) I, III e IV.

Comentários:

Vamos analisar os itens.

I. Certo. Os dados no JSON sao pares de chaves e valores.
II. Errado. Os dados são separados por vírgulas.
III. Certo. As chaves, além de delimitar o arquivo JSON, delimita também objetos.
IV. Certo. Os colchetes [] são responsáveis por delimitar as arrays.

Temos, portanto, I, III e IV corretas.

Gabarito: Letra E

---

24. (FGV/SEFAZ AM/2022) Sobre a gramática da linguagem de intercâmbio de dados JSON (JavaScript Object Notation), assinale a afirmativa correta.

a) Um objeto é um conjunto ordenado de pares nome-valor.
b) Um valor numérico pode ser declarado em base decimal, octal ou hexadecimal.
c) A ausência intencional de qualquer valor de objeto é representada com a palavra reservada undefined.
d) Uma string é uma sequência de zero ou mais caracteres unicode envolto por aspas duplas.
e) Os espaços em branco são proibidos entre qualquer par de tokens.

Comentários:

Vamos analisar cada alternativa.

a) Errado. Os objetos são desordenados, isso é, não possuem uma ordem definida.
b) Errado. Os valores numéricos só podem adotar a base decimal, seja inteiro ou float.
c) Errado. A ausência de valores é representada por null.
d) Certo. Essa é a definição de uma *string*.
e) Errado. Os espaços em branco entre os tokens (chaves ou valores) são ignorados.

Gabarito: Letra D

---

25. (CEBRASPE/TCE RJ/2022) Julgue o item subsequente, a respeito dos padrões XML, XSLT, SOAP, REST e JSON.

JSON é uma alternativa ao XML para representar dados, baseado em texto e com pares nome e valor para representar as informações.

Comentários:

Perfeito! JSON é uma alternativa mais leve e menos verbosa ao XML, representando dados em pares de chaves e valores.

Gabarito: Correto

QUESTÕES COMENTADAS

01. (CEBRASPE/TST/2024)

```
<html>
<head>
   <script src="https://ajax.googleapis.com/ajax/libs/jquer y/3.7.1/jquery. min.js"></script>
   <script>
      $(document).ready(function(){
         $("button").click(function(){
            $("#nome").hide();
         });
      });
   </script>
</head>
<body>
   <h2>Poder Judiciário</h2>
   <p>TST</p>
   <p id="nome">Tribunal Superior do Trabalho</p>
   <button>Clique Aqui</button>
</body>
</html>
```

A execução do código precedente, desenvolvido com HTML, JavaScript e jQuery, provê o resultado apresentado a seguir.

Com base nessas informações, assinale a opção que representa corretamente o que será apresentado se o usuário clicar no botão Clique Aqui
.

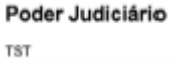
a)

b)

c)

d)

e)



02. (VUNESP/UNICAMP/2023) Segundo a sintaxe do jQuery, um determinado caractere é utilizado para acessar suas funcionalidades (identificar uma função jQuery). O caractere em questão é o:

a) $
b) /
c) ^
d) ~
e) *

03. (VUNESP/UNICAMP/2023) A seguir é apresentado um pequeno trecho de código escrito em Javascript com jQuery.

```
$(function() {
    // Algum codigo aqui
});
```

É correto afirmar que o código apresentado será executado quando

a) a página web for aberta.
b) a página web for fechada.
c) a página web estiver pronta.
d) o jQuery for baixado para a página web.
e) um timeout definido anteriormente no código for alcançado.

04. (FGV/DPE RS/2023) Considere o código JQuery apresentado a seguir.

```
$("button.mudar").on("click",(evt)=>
  { $("#mensagem").html("Clicado"); });
```

Sobre o funcionamento do código apresentado, é correto afirmar que:

a) associa a todos os botões com a classe "mudar" uma resposta ao evento de clique;
b) modifica o conteúdo interno do botão que foi clicado;
c) aciona o clique em um botão cujo id é "mudar";
d) altera o atributo de estilo do componente cujo id é "mensagem";

e) muda o texto "#mensagem" para "Clicado" nos botões da classe "mudar

05. (FGV/TJ SE/2023) No contexto dos seletores (selectors) do jQuery, a expressão

$(".intro")

aplica-se à seleção de:

a) o elemento com id="intro";
b) todos os elementos <intro>;
c) todos os elementos com class="intro";
d) todos os elementos com display="intro";
e) todos os elementos com type="intro".

06. (FGV/CGU/2022) Observe o código a seguir.

```
<!DOCTYPE html>
<html>
 <head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"> </script>
 <script>
   //Inserir código aqui
 </script>
</head>

<body>
   <h3>Lorem Ipsum</h3>

<div>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
  <p>Pellentesque ut nunc elit. Nullam rutrum nibh placerat.</p>
</div>

<p>Fusce rutrum, lacus in laoreet egestas, ex lacus laoreet quam</p>

<p>Vestibulum ante ipsum primis in posuere cubilia curae</p>

<button>Remover</button>
        </body>
     </html>
```

Para adicionar ao botão o comportamento de remover apenas os elementos-filhos e o conteúdo do elemento div, pode-se utilizar o seguinte trecho de código no elemento script:

a)

```
$(document).ready(function(){
      $("button").click(function(){
       $("div").empty();
```

```
    });
  });
```

b)

```
$(document).ready(function(){
    $("button").click(
  $("div").clear());
 });
```

c)

```
$(document).ready(function(){
     $("Remover").click(function(){
       $("div").removeContent();
     });
  });
```

d)

```
$(document).ready(function(){
      $("button").click(function(){
        $("div").remove());
     });
```

e)

```
$(document).ready(function(){
     $("Remover").click(
      $("div").empty(“p”));
    });
```

07. (FGV/SEFAZ AM/2022) Analise o código de uma página web a seguir.

```
1  <html>
2      <head>
3          <script type="text/javascript"
4                  src="/jquery/jquery-3.6.0.min.js">
5          </script>
6
7          <script>
8              $(document).ready(function(){
9                $("button").click(function(){
10
11               });
12             });
13             </script>
14     </head>
15     <body>
16         <div id="titulo">Meu Site</div>
17         <button>Clique aqui</button>
18     </body>
19 </html>
```

Assinale a opção que indica o código JQuery que pode ser incluído na linha 10 para tornar vermelha todas as letras da palavra "Meu Site", quando o botão rotulado com o texto "Clique Aqui" é pressionado.

a) $("div").css('color', '#FF0000')
b) $("div").html({'background-color': '#FF0000'})
c) $("#titulo").append("{'color': '#FF0000'}")
d) $("#titulo").attr('foreground-color', '#FF0000')
e) $("div").style('color', '#FF0000')

08. (FCC/PGE AM/2022) A instrução jQuery para aplicar a cor de fundo vermelha em todos os elementos de uma página HTML é

a) $("div").css("background-color", "#ff0000");
b) $("div").style("background-color", "#ff0000");
c) $("#div").css("background-color", "#0000ff");
d) $("div").style("background-color", "#00ff00");
e) $("div").css("background-color", "#0000ff");

09. (VUNESP/ALESP/2022) No jQuery, se o método .attr() for utilizado sob um conjunto de elementos retornados por um seletor, o resultado será

a) o resultado da função .attr() aplicado ao primeiro elemento do conjunto.
b) um iterador que a cada iteração retorna o resultado do método .attr() aplicado a um elemento do conjunto.
c) uma função que recebe como parâmetro um elemento do conjunto e que retorna o resultado do método .attr() aplicado ao elemento.
d) uma lista contendo o resultado do método .attr() aplicado a cada um dos elementos do conjunto.
e) o resultado da função .attr() aplicado ao último elemento do conjunto.

10. (FCC/TRT 4/2022) A instrução jQuery para colocar o conteúdo de todos os elementos p da página HTML com letras de tamanho 40px é

a) $("p.font").style("size", "40px");
b) $("p").css("font-size", "40px");
c) $("p").css("size", "40px");
d) $("p").style("font-size", "40px");
e) $("p").css("font-size:40px");

11. (FGV/TCE TO/2022) Analise o código jQuery a seguir.

```
<script>
$(document).ready(function(){
   $("p").click(function(){
      $(this).hide();
    });
 });
</script>
```

É correto concluir que o termo this refere-se:

a) a um elemento HTML cujo id é "p", que tenha sido clicado;
b) ao documento HTML recém-carregado;
c) ao método ready;
d) a um elemento HTML cuja tag é "p", que tenha sido clicado;
e) a um elemento HTML cuja classe é "p", que tenha sido clicado.

12. (FGV/TRT 13/2022) Analise o código da página HTML a seguir:

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script type="text/javascript" src="jquery.min.js"></script>
5          <script type="text/javascript" language="javascript">
6              $(function() {
7
8              });
9          </script>
10     </head>
11     <body>
12         <p class="title">TEXTO_A</p>
13         <p id="title">TEXTO_B</p>
14         <p>TEXTO_C</p>
15     </body>
16 </html>
```

A instrução jQuery que, ao ser incluída na linha 7, irá alterar para vermelho somente a cor da palavra TEXTO_A nessa página é

a) $("p > title").css("color", "red").
b) $("#title").css("color", "red").
c) $(".title").css("color", "red").
d) $("p , title").css("color", "red").
e) $(":title").css("color", "red").

13. (FCC/TRT 14/2022) Utilizando jQuery, para remover somente os elementos filho do contêiner cujo valor do atributo id é caixa, utiliza-se a instrução

a) $(".caixa").empty();
b) $("#caixa").remove();
c) $("#caixa").empty();
d) $(".caixa").delete();
e) $("#caixa").clear();

14. (FCC/TRT 5/2022) Considere o bloco de código jQuery abaixo, parte de uma página HTML, em condições ideais.

```
$(document).ready(function () {
    $("#ocultar").click(function () {
    ......I......;
  });
});
```

Considere a existência de um botão criado por meio da tag button, cujo atributo id possui o valor ocultar. Considere, ainda, que há na página diversos parágrafos criados por meio da tag p. Nestas condições, para que, ao clicar no botão, o conteúdo de todos as tags p seja ocultado, a lacuna I deve ser corretamente preenchida com

a) $("p").display(none)
b) $("p").hide()
c) $("p").visible(false)
d) $("p").show(false)
e) $("p").setVisible(false)

15. (GP CP2/CP II/2022) A jQuery faz uso do construtor $() para encontrar os elementos HTML dentro da página e utilizar as funções da biblioteca. A sintaxe básica é $(selector).action().

A seguir são listados alguns exemplos de seletores com a sua sintaxe:

I. $(this).hide()
• Demonstra o método jQuery hide() usado para esconder o elemento HTML atual.

II. $("#test").hide()
• Demonstra o método jQuery hide() usado para esconder o elemento da classe="test".

III. $("p").hide()
• Demonstra o método jQuery hide() usado para esconder todos os elementos dentro das tags <p>.

IV. $(".test").hide()
• Demonstra o método jQuery hide() usado para esconder todos os elemento da classe "test".

V. $(":button")
• Seleciona todos os elementos <button> e também <input type="button">

Dentre estes seletores, os que apresentam a sintaxe corretamente correspondente são

a) I, II, III e IV.
b) I, II,IV e V.
c) I, II, III e V.
d) I, III, IV e V

16. (CEPUERJ/UERJ/2021) Um arquivo HTML será carregado com a implementação a seguir em um navegador web:

```
<html>
<head>
   <script src=https://code.jquery.com/jquery-3.6.0.js crossorigin="anonymous">
   </script>
   <script type="text/javascript">
      $(document).ready(function() {
         $("#btn1").click(function() {
            alert(processa($("#n1").val()))
      });
      $(".btn").click(function() {
         var a = 2;
         var b = [4, function() {}, 'A'];
         var c = ($("#n2").html() / a) != 2;
         var x = $("#n1").val();
         alert((jQuery.isFunction(b[1]) && c) ? x * a : x / a);
      });
    });
    function processa(x) {
       return x ** 3;
    }
  </script>
</head>
<body>
   <input id="n1" type="hidden" value="2">
   <div id="n2">4</div>
   <button id="btn1" class="btn-primary">A</button>
   <button id="btn2" class="btn">B</button>
</body>
</html>
```

Considerando os conceitos de jQuery, o acionamento do botão "B" irá exibir em tela o valor:

a) 8
b) 6
c) 4

d) 1

17. (FGV/TJ RO/2021) No contexto da jQuery, o código

```
$(document).ready( ){
.....
})
```

previne que as funções jQuery sejam executadas antes da carga total da página.

A primeira linha desse script pode ser substituída por um método mais abreviado:

a) $(begin{
b) $(function( ){
c) $(go( ){
d) $(main{
e) $(ready( ){

18. (FGV/FUNSAÚDE CE/2021) Considere o trecho de código jQuery.

```
$(document).ready(function(){
  $("button").click(function(){
    $("#xpto").hide();
  });
});
```

Assinale o efeito da execução desse trecho.

a) Na carga da página, todos os elementos com tag = "button" são ocultados.
b) O elemento com o atributo id = "xpto" é ocultado quando qualquer botão for clicado.
c) Qualquer elemento com name = "xpto" é ocultado quando o próprio elemento for clicado.
d) Qualquer elemento com tag = "button" é ocultado quando o próprio elemento for clicado.
e) Todos os elementos com class = "xpto" são ocultados quando qualquer botão for clicado.

19. (FGV/BANESTES/2021) Analise o fragmento de código jQuery a seguir.

```
$("#x1").click(function(){
   $("#x2").html("<b>Hello world!</b>");
});
```

Numa página web, esse trecho faz com que um clique:

a) em qualquer elemento cuja tag é x1, cause uma alteração no conteúdo de todos os elementos com a tag x2;
b) em qualquer elemento da classe x1, cause uma alteração no conteúdo do elemento cujo id é x2;

c) no elemento cujo atributo id é x1, cause uma alteração no conteúdo de todos os elementos da classe x2;
d) no elemento cujo atributo id é x1, cause uma alteração no conteúdo do elemento cujo id é x2;
e) no elemento cujo atributo id é x1, cause uma alteração no conteúdo de todos os elementos com a tag x2.

20. (AOCP/FUNPRESP-JUD/2021) Quanto ao desenvolvimento web, julgue o seguinte item.

JavaScript é uma linguagem que sofre muito com compatibilidade entre navegadores. A jQuery sofre com o mesmo problema. Animações, manipulação de DOM e outra tarefas corriqueiras são mais complexas e menos produtivas ao usar o jQuery.

21. (AOC/SANESUL/2021) Com a utilização do jQuery como sendo uma biblioteca JavaScript, existem alguns métodos simples que são adicionados por conveniência. Dentre eles, temos um que exibe ou oculta elementos correspondentes, usando efeitos personalizados. Qual é o nome desse método?

a) .toggle()
b) .show()
c) .scrollParent()
d) .labels()
e) .enableSelection()

22. (FGV/IMBEL/2021) Analise o script jQuery exibido a seguir.

```
$(document).ready(function(){
  $("button").click(function(){
$("p").hide(500);
  });
});
```

Numa página Web, esse código faz com que

a) se um elemento com id=”p” for clicado, todos os elementos com id=”button” são ocultados.
b) se um elemento com id=”button” for clicado, todos os elementos com id=”p” são ocultados.
c) todos os parágrafos são ocultados quando a página é carregada, e voltam a ficar visíveis quando algum botão for clicado.
d) um clique, em qualquer botão presente na página, oculta todos os parágrafos.
e) um clique, em qualquer parágrafo presente na página, oculta todos os botões.

23. (FCC/ALAP/2020) Em um fragmento de código jQuery, para representar um elemento HTML cujo valor de id é bloco e um elemento cujo valor de classe é centro, utilizam-se, respectivamente:

a) $(".bloco") e $("#centro")

b) iQuery("#bloco") e jQuery(".centro")
c) #("bloco") e $("centro")
d) $("#bloco") e $(".centro")
e) $Id("bloco") e $class("centro")

24. (COMPERVE/UFRN/2020) O jQuery é uma biblioteca Javascript criada para otimizar o desenvolvimento de aplicações web. Em relação às funções disponíveis na versão 3.3.1, é correto afirmar:

a) .add() adiciona uma nova classe passada por parâmetro a cada elemento da seleção.
b) .append() adiciona o conteúdo passado por parâmetro no final de cada elemento da seleção.
c) .each() remove cada elemento da seleção.
d) .find() busca por um termo em toda a página HTML.

25. (COMPERVE/UFRN/2020) O jQuery pode selecionar elementos HTML a partir de seletores, otimizando o desenvolvimento. Sobre a utilização de seletores no jQuery é correto afirmar:

a) $('td, th'); retornará todos os elementos th que são filhos de td.
b) $('#formulario'); retornará todos os elementos da classe "formulario".
c) $('p:first'); retornará o primeiro dos elementos p.
d) $('.paragrafo'); retornará o elemento com id igual a "paragrafo".

26. (FCC/DPE RJ/2019) Analise o trecho de código a seguir, extraído de uma página na qual foram inseridos os devidos links para o acionamento da biblioteca JQuery.

```
...
<head>
...
<script>
$(document).ready(function(){
  $("#2").click(function(){
    $("p").hide();
  });
});
</script>
</head>
<body>
<p id="1">Clique 1</p>
<p id="2">Clique 2</p>
<p id="3">Clique 3</p>
<body>
...
```

Sabendo-se que essa página exibe inicialmente três linhas, é correto afirmar que um clique:

a) no primeiro parágrafo oculta somente a primeira linha originalmente exibida;

b) no segundo parágrafo em nada altera o que é exibido pela página;
c) no terceiro parágrafo oculta a primeira e a terceira linhas originalmente exibidas;
d) no segundo parágrafo oculta somente a segunda linha originalmente exibida;
e) no segundo parágrafo oculta as três linhas originalmente exibidas.

27. (FCC/ISS MANAUS/2019) Considere o bloco jQuery abaixo, em uma página web onde as referências às bibliotecas necessárias estão corretas.

```
<script>
$(document).ready(function(){
 $("button").click(function(){
  ..I..;
 });
});
</script>
```

Para adicionar ao div com id="caixa" as classes principal e branca, a lacuna I deve ser preenchida por

a) $("div#caixa").class("principal branca").
b) $("#caixa").addClass("principal branca").
c) $("div.caixa").addClass("principal").addClass("branca").
d) $("div#caixa").appendClass("principal branca").
e) $(".caixa").addClass("principal branca").

28. (FCC/PREF. MANAUS/2019) Considere o fragmento jQuery abaixo, criado em uma página web em condições ideais.

```
<script>
 $(document).ready(function(){
  ..I..
 });
</script>
```

Para aplicar a todos os elementos HTML com atributo class="par" a cor de letra azul, a lacuna I deve ser preenchida por

a) $("p").class(".intro").style("color", "#0000ff");
b) $("p").filter(".par").style("color", "#0000ff");
c) $("p").class("intro").css("font-color", "#0000ff");
d) $("p").filter(".par").css("color", "#0000ff");
e) $("p").eq(".intro").css("color", "#0000ff");

29. (FCC/SANSA/2019) Considere a página abaixo, desenvolvida utilizando-se jQuery.

```
<!DOCTYPE html>
<html>
<head>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.0/jquery.min.js">
</script>
<script>

$ ( document ) .ready(function ( ) {
      $ ( "button" ) .click(function ( ) {
          ...I... ("sanasa.php", function(data, status) {
                alert ( "Retorno: " + data + "\nStatus: " + status ) ;
           } ) ;
      } ) ;
} ) ;

  < /script >
< /head >
< body >
     < button>Obter dados</button >
< /body >
< /html >
```

Para que, ao clicar no botão, seja feita uma requisição ao arquivo sanasa.php que está no servidor, de forma que o retorno seja recebido em data e o status da operação em status, a lacuna I deve ser corretamente preenchida por

a) $.get
b) $request
c) $post
d) $.send
e) $submit

30. (CEBRASPE/STM/2018) Julgue o item seguinte, a respeito de JQuery.

Em JQuery, o método $.get() é usado para executar uma solicitação HTTP.

31. (CEBRASPE/STM/2018) Julgue o item seguinte, a respeito de JQuery.

O seletor JQuery $(":disabled") seleciona todos os elementos que estão desativados (disabled).

32. (FCC/DPE AM/2018) Um Analista de Sistemas criou, em uma página em condições ideais, o bloco jQuery abaixo.

```
$(document).ready(function(){
    $("input[name^='lei']").css("background-color", " #0000FF");});
</script>
```

Esse fragmento aplica a cor de fundo

a) verde somente nos elementos input cujo atributo name contenha a palavra lei.

b) azul em todos elementos input cujo atributo name contenha um valor terminado pela palavra lei.
c) vermelha somente nos elementos input cujo atributo name contenha a palavra lei.
d) azul em todos os elementos input cujo conteúdo do atributo name inicie com a palavra lei.
e) verde em todos elementos input cujo atributo name contenha a palavra lei.

33. (FCC/DPE AM/2018) Considere o fragmento de código jQuery abaixo.

```
<script>
$(document).ready(function(){
    $("tr:gt(3)").css("background-color", "red");
});
</script>
```

O código aplica a cor de fundo vermelha em todas as tags <tr> cujo índice seja

a) igual a 3.
b) maior ou igual a 3.
c) maior que 3.
d) menor ou igual a 3.
e) menor que 3.

34. (FCC/SEGEP MA/2018) Um Programador de Sistemas está desenvolvendo um site e deseja esconder os elementos de uma lista não ordenada cujos índices sejam menores do que 2. Para isso terá que utilizar no bloco jQuery a instrução

a) $("ul li[index<2]").hide();
b) $("ol li[index<2]").hide();
c) $("ul li:gt(2)").hide();
d) $("ul li:lt(2)").hide();
e) $("ol li:eq(2)").hide();

35. (FGV/MPE AL/2018) Observe o trecho de código JQuery a seguir.

```
$("button").click(function(){
   $("p").hide("slow", function(){
      alert(".....");
   });
});
```

Essa sintaxe ilustra o uso de funções que previnem certos comportamentos anômalos no processamento de efeitos.

Assinale o termo que caracteriza esse uso.

a) Callback parameter.
b) Event parameter.
c) Message parameter.
d) Script parameter.
e) Trigger parameter.

GABARITO

1. Letra A
2. Letra A
3. Letra C
4. Letra C
5. Letra C
6. Letra A
7. Letra A
8. Letra A
9. Letra A
10. Letra B
11. Letra D
12. Letra C
13. Letra C
14. Letra B
15. Letra D
16. Letra D
17. Letra B
18. Letra B
19. Letra B
20. Errado
21. Letra A
22. Letra A
23. Letra D
24. Letra B
25. Letra B
26. Letra E
27. Letra B
28. Letra D
29. Letra A
30. Correto
31. Correto
32. Letra D
33. Letra C
34. Letra D
35. Letra A

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.