

**GRUPO 1 [7 valores]**

NOTA: Neste grupo, responda com um máximo de 5 linhas em cada pergunta.

1. [1,5] A especificação do protocolo HTTP indica que o método PUT tem de ser idempotente. Qual a relevância desta propriedade, dada a funcionalidade deste método?
2. [1] Na reestruturação de uma aplicação web, decidiu-se acrescentar o header Location às respostas com status code 404 correspondentes a todos os recursos que mudaram de localização. Comente esta decisão.
3. [2] Considere o seguinte excerto de código em JavaScript:

```
1 function createObj () {  
2     const items = []  
3     return { addItem: items.push }  
4 }  
5 const obj = createObj()  
6 obj.addItem("SLB")
```

- a. [1] A instrução da linha 6 não adiciona a string "SLB" ao array items declarado na linha 2. Justifique.
- b. [1] Realize as alterações necessárias ao código, de modo a que o método addItem adicione ao array items o objeto que recebe como argumento.
4. [1,5] Na versão corrente da especificação HTML5 não é válido o antigo elemento <font>, que já estava desaconselhado há bastante tempo e que permitia alterar o tamanho e a cor de um bloco de texto. Porque razão se entendeu que este elemento não se enquadra no âmbito da especificação HTML? ~~Justifique~~
5. [1] Justifique qual das seguintes estruturas de URL é adequada para referir um recurso numa single-page application  
<http://example.com/docs/group/123/id/456>      <http://example.com/#docs/group/123/id/456>

**GRUPO 2 [7 valores]**

6. [6] Implemente as seguintes funções em JavaScript:

- a. [1,5] filterProperties(propNames, obj) que recebe um array de strings em propNames e um objeto em obj, retornando um novo objeto com as propriedades de obj cujos nomes estão presentes em propNames. Se em propNames existirem nomes que não correspondem a qualquer propriedade em obj, esta propriedade não é adicionada ao objeto retornado.
- b. [1,5] filterPropertiesN(propNames, objs) que recebe um array de strings em propNames e um array de objetos em objs, retornando um novo array de objetos correspondente à aplicação da função filterProperties com propNames a cada um dos elementos de objs.

NOTA: Na implementação desta função, a utilização de qualquer de ciclo for/while ou do método Array.forEach reduz a cotação em 50%.

- c. [2,5] Implemente um middleware Express que substitui o método json() do segundo argumento (response), de modo a ter um comportamento especializado quando a query string do uri inclui o nome filter. Nesse caso, o valor de filter é tomado como uma sequência de strings separadas por vírgulas, filtrando-se a resposta com a função filterProperties ou filterPropertiesN, de modo a que o objeto ou objetos retornados apenas contenham as propriedades identificadas em filter.

NOTAS:

- obj instanceof Array permite determinar se obj é uma instância de Array
- Veja exemplos na página seguinte.

Exemplos:

Uri: <http://somehost/example.com?filter=a,c,e>

Resposta produzida por outros *middlewares*: { a: 123, b: 234, c: 345, d: 456 }

Resposta enviada ao cliente: { a: 123, c: 345 }

Uri: <http://somehost/example.com?filter=a,c,e>

Resposta produzida por outros *middlewares*:

[ { a: 123, b: 234, c: 345, d: 456 }, { a: "aa", b: "bb", c: "cc", d: "dd" } ]

Resposta enviada ao cliente: [ { a: 123, c: 345}, { a: "aa", c: "cc" } ]

7. [1.5] Indique, justificando, o erro grave presente na função get e apresente uma versão corrigida:

```
1 function get(url, cb) {
2     let result = null
3     fetch(url).then(resp => { result = resp; }).catch(err => cb(err))
4     while (!result) ; /* waiting */
5     cb(null, result)
6 }
7 }
```

### GRUPO 3 [6 valores]

8. [6] Pretende-se implementar na aplicação CIBORG, desenvolvida no trabalho prático, a funcionalidade de adicionar vários jogos a um grupo.

NOTAS:

- Nas alíneas que requeiram implementação de código, considere apenas os casos de sucesso.
- O método `Promise.all(array)` recebe um *array* de `Promise` e retorna uma `Promise` que se resolve quando todas estiverem resolvidas.

a. [1,5] Especifique o *endpoint* que está disponível na componente servidora da aplicação que suporta esta funcionalidade. No pedido são enviados os identificadores dos jogos a acrescentar, bem como o identificador do grupo. Na definição do *endpoint*, indique o que for necessário para que quem usa a API tenha a informação para aceder a esse *endpoint* e conseguir adicionar vários jogos a um grupo, bem como para saber identificar e interpretar as situações de erro.

NOTAS:

- O método deste *endpoint* é `POST`
- A resposta tem o formato Json, quer represente conclusão com sucesso ou não.
- Os erros suportados são: pedido inválido, recurso não encontrado e erro no servidor.

b. [2] Implemente o método `addGamesToGroup(req, rsp)` do módulo `ciborg-web-api`, que chama o método `addGamesToGroup(groupId, gamesIds)` do módulo `ciborg-services`. Este último recebe o identificador do grupo em `groupId` e um *array* com os identificadores dos jogos em `gamesIds`.

c. [2,5] Implemente o método `addGamesToGroup(groupId, gamesIds)` do módulo `ciborg-services`. Na implementação deste método use os métodos `getGames(gamesIds)` do módulo `board-games-data` e o método `addGameToGroup(groupId, game)` do módulo `ciborg-db`. O método `getGames(gamesIds)` recebe um *array* de identificadores dos jogos e retorna um *array* de `Promise` correspondente a cada jogo. O método `addGameToGroup(groupId, game)` recebe o identificador do grupo em `groupId`, o objeto com a informação do jogo (tal como retornado por `getGames`) em `game` e retorna uma `Promise`.