Vaga: Estágio – TI: Web Analytics

Candidato: Caio Teruo Fuzita Rabelo da Silva

1. Módulos

1.1. File System

```
1    fs = require('fs');
```

Necessário para abrir arquivos, modificar e salvar.

2. Funções

2.1. Abrir arquivo .json

```
40   function readJson(){
41       const data = fs.readFileSync('./broken-database.json','utf8'); // abre o arquivo JSON
42       return JSON.parse(data); // transforma o JSON em objeto javascript
43   }
```

Cria uma constante com os dados do arquivo 'broken-database.json', e para que seja possível manipular os dados, é utilizado o método .parse, que transforma o arquivo .json em objeto Javascript.

2.2. Correção de nomes

```
46   function fixName(str){
47       return str.replace(/æ/g, 'a').replace(/¢/g, 'c').replace(/ø/g, 'o').replace(/ß/g, 'b'); //https://www.
         devmedia.com.br/javascript-replace-substituindo-valores-em-uma-string/39176
48   }
```

Utiliza o método .replace com o parâmetro global (/g) para substituir os caracteres corrompidos.

2.3. Correção das quantidades

```
51   function fixQuantity(quantity){
52       return quantity || 0; // retorna o valor de quantity, e se não houve um valor, retorna 0;
53   }
```

Retorna o próprio valor do 'quantity', se não houve um valor, ele retornará 0.

2.4. Correção dos preços

```
56   function fixPrice(price){
57       return parseInt(price); // transforma tipo string em tipo number
58   } //https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/parseInt
```

Retorna a variável como tipo ‘number’.

2.5. Salvar o arquivo

```
61  function writeJson(file){
62      var dados = JSON.stringify(file); // transforma o objeto javascript em .json
63      fs.writeFileSync('./saida.json', dados);
64  }
```

Primeiramente transforma o objeto Javascript de volta para o tipo .json, após isso, salva o arquivo com o nome ‘saida.json’.

2.6. Abrir o arquivo corrigido

```
67  function readFixedJson(){
68      const dataFixed = fs.readFileSync('./saida.json', 'utf8');
69      return JSON.parse(dataFixed);
70  }
```

Realiza a mesma operação da função ‘readJson’, mas dessa vez com o arquivo já corrigido ‘saida.json’.

2.7. Verificar e listar categorias únicas

```
73  function verifyCategory(){
74
75      var categorias = []; // criação de um array vazio
76      for (var i in fixedJson){
77          categorias.push(fixedJson[i].category); // atribui todas as categorias existentes no array
            'categorias'
78      }
79
80      var categoriasUnicas = []; // criação de um array vazio
81
82      for (var i = 0; i < categorias.length; i++){
83          if (categorias[i] != categorias[i+1]){
84              categoriasUnicas.push(categorias[i]); // varredura por todos os elementos da array
                'categorias', todos os valores únicos vão para o 'categoriasUnicas'
85          }
86      }
```

Criação do array ‘categorias’, para armazenar apenas os nomes das caterogias, junto com um laço para percorrer o arquivo .json, todas as categorias existentes no arquivo .json são atribuídas à array. Após isso, é criado o array ‘categoriasUnicas’, que, a partir de um laço em conjunto com um ‘if’, percorre o array onde está presente todas as categorias existentes. O ‘if’ verifica se as categorias são repetidas, como as categorias já estão em ordem alfabética, somente é necessário verificar se o elemento se repete no índice i+1, quando não há a repetição, é executado o algoritmo para atribuir o elemento para o array ‘categoriasUnicas’.

2.8. Calcular estoque

```
function calculadora(products){
    var soma = 0; // iniciador da soma
    for (var i in products){
        soma += products[i]['quantity'] * products[i]['price']; // multiplica a quantidade pelo preço
    }
    return soma
}
```

A função para calcular o estoque é iniciada atribuindo o valor 0 para a soma, após isso, um laço é utilizado para percorrer o objeto inteiro, multiplicando a quantidade pelo preço, e no fim, retornando o valor da soma.

3. Execução do algoritmo

```
var brokenJson = readJson();
```

O algoritmo é iniciado criando a variável 'brokenJson', que armazenará todo o objeto corrompido, a partir da execução da função 'readJson()'.

```
for (var i in brokenJson) {
    brokenJson[i]['name'] = fixName(brokenJson[i]['name']);
    brokenJson[i]['quantity'] = fixQuantity(brokenJson[i]['quantity']);
    brokenJson[i]['price'] = fixPrice(brokenJson[i]['price']);
}
```

Um laço é utilizado para percorrer o objeto inteiro, fazendo as devidas correções nos nomes, quantidades e preços do arquivo.

```
writeJson(brokenJson);
```

Após as devidas correções, o arquivo é salvo.

```
var fixedJson = readFixedJson();
```

Para que seja feita as devidas validações, é criada a variável 'fixedJson' já com o arquivo corrigido.

```
fixedJson.sort(function compare(a, b){
    return (a.category > b.category) ? 1 : (a.category < b.category) ? -1 : (a.id > b.id) ? 1 : (a.id
    < b.id) ? -1 : 0;
}); //https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array/sort
```

O método '.sort' é utilizado para ordenar os itens primeiramente por categoria, e logo em seguida por ID.

```
27  var categoriaProdutos = verifyCategory();
28
29  // realiza o cálculo de estoque de acordo com cada categoria ex
30  for (var i = 0; i < categoriaProdutos.length; i++) {
31      var produtos = fixedJson.filter((search) => {
32          return search.category == categoriaProdutos[i];
33      })
34      console.log(categoriaProdutos[i], calculadora(produtos))
35  }
```

Por fim, é realizado o cálculo de estoque de cada categoria existente, o laço percorre todas as categorias existentes, utilizando o método Search é possível obter todos os elementos que contém determinada categoria, e assim é passado como argumento para a função de calcular o estoque.

4. Fontes

https://nodejs.org/api/fs.html#fsreadfilepath-options-callback

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array/sort

https://www.devmedia.com.br/javascript-replace-substituindo-valores-em-uma-string/39176

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/parseInt