Nome: Nicolau Roberto Wojcieszuk Junior

Candidato para a vaga de estágio – TI: Web Analytics

1. Importando módulo para ler e alterar arquivos:

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

2. Descrição das funções:

2.1. Função para ler o arquivo corrompido

Primeiramente é feito um try catch para tratar um erro caso haja problema em ler o arquivo corrompido. É criado uma variável para armazenar os dados do banco de dados, lendo o arquivo pelo método .readFileSync e transformando o json em objeto Javascript pelo método .parse.

```
function loadBrokenFile (){
  try{
    var database = fs.readFileSync("./broken-database.json", "utf8");
    return JSON.parse(database);
  }catch{
    console.log("Error");
  }
}
```

2.2. Função para corrigir os nomes dos produtos

A função recebe o nome do produto como parâmetro e é utilizado o método .replace para substituir os caracteres corrompidos pelos caracteres certos, e é usado o (/g) para substituir globalmente o caractere, independente da posição que ele estiver na string.

```
function fixingName (name){
  return name.replace(/æ/g, "a").replace(/ß/g, "b").replace(/¢/g, "c").replace(/ø/g, "o");
}
```

2.3. Função para corrigir os preços dos produtos

A função recebe o preço do produto como parâmetro e é utilizado o parseFloat para retornar o valor como tipo real.

```
function fixingPrice (price){
  return parseFloat(price);
}
```

2.4. Função para corrigir a quantidade dos produtos

A função recebe a quantidade do produto como parâmetro, caso a quantidade for maior que 0 retorna a própria quantidade, porém se não houver valor ou ser valor 0, retorna o próprio 0.

```
function fixingQuantity (quantity){
  if(quantity > 0){
    return quantity;
  }
  else{
    return 0;
  }
}
```

2.5. Função para salvar o arquivo corrigido

É feito um try catch para tratar um erro caso haja problema em salvar o arquivo corrigido A função recebe o arquivo corrigido como parâmetro, ele é transformado de objeto Javascript de volta para json pelo método .stringify e salvo com nome “saída.json” pelo método .writeFileSync.

```
function saveFixedFile (file){
  try{
    var databaseFixed = JSON.stringify(file);
    fs.writeFileSync("./saida.json", databaseFixed);
  }catch{
    console.log("Error");
  }
}
```

2.6. Função para carregar o arquivo corrigido

É feito um try catch para tratar um erro caso haja problema em ler o arquivo corrigido. É criado uma variável para armazenar os dados do banco de dados, lendo o arquivo pelo método .readFileSync e transformando o json em objeto Javascript pelo método .parse como foi feito na primeira função.

```
function loadFixedFile (){
  try{
    var database = fs.readFileSync("./saida.json", "utf8");
    return JSON.parse(database);
  }catch{
    console.log("Error");
  }
}
```

2.7. Função para ordenar por categoria e id

A função recebe 2 itens do banco de dados corrigido como parâmetro por meio do método .sort usado para chama-la. Caso a categoria do primeiro item seja igual ao do segundo item, será feita a comparação dos id's visto que já são da mesma categoria. Porém se for de categorias diferentes, será comparado se a categoria do primeiro item é menor que a categoria do segundo item.

```
function sortingCategories (item1, item2){
  if(item1.category == item2.category){
    if(item1.id < item2.id){
      return -1;
    }else{
      return 1;
    }
  }else if(item1.category < item2.category){
    return -1;
  }else{
    return 1;
  }
}
```

2.8. Função para calcular estoque por categoria

Primeiramente é criado uma variável de soma inicializada com 0, e 3 arrays vazios, onde vão armazenar todas as categorias, categorias únicas e o valor por categoria. O primeiro for é para criar um array com todas as categorias, onde é usado o método push para adicionar a categoria atual no array. O segundo for é para filtrar apenas categorias únicas, onde é feito a comparação entre a categoria atual e a próxima, caso elas sejam diferentes, a categoria atual será adicionada no array. O terceiro e quarto for é para fazer o cálculo em si, onde será percorrido as categorias únicas e em seguida os produtos, comparando a categoria do produto atual com a categoria no array. Caso as categorias sejam iguais será feito o cálculo e armazenado na variável soma. Ao percorrer todos os produtos, o valor da soma será adicionado ao array de valor por categoria e em seguida zerar a soma para a próxima categoria.

```
function stockValue(){
  var soma = 0 , allCategories = [], uniqueCategories = [], valueCategory = [];
  //For para criar um array com todas as categorias
  for(var i in fixedFile){
    allCategories.push(fixedFile[i].category);
  }

  //For para filtrar um array apenas com as categorias únicas
  for(var i = 0, j = 0; i < allCategories.length; i++){
    if(allCategories[i] != allCategories[i+1]){
      uniqueCategories.push(allCategories[i]);
    }
  }

  //For para percorrer as categorias únicas e calcular valor
  for(var i in uniqueCategories){
    for(var j in fixedFile){
      if(fixedFile[j].category == uniqueCategories[i]){
        soma += fixedFile[j].price * fixedFile[j].quantity;
      }
    }
    valueCategory[i] = soma;
    soma=0;
  }
  //For para imprimir categoria, valor de estoque
  for(var i = 0; i< uniqueCategories.length; i++){
    console.log(uniqueCategories[i],valueCategory[i]);
  }
}
```

3. Referências:

https://www.w3schools.com/nodejs/nodejs_filesystem.asp

https://nodejs.dev/learn/reading-files-with-nodejs

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

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

https://www.alura.com.br/artigos/convertendo-string-para-numero-em-javascript?gclid=CjwKCAiAx8KQBhAGEiwAD3EiP0STHdHnHYpdaXHFq5I4drkmiGE0NV8WIR39d8vOkMplYdeaD85xDhoCf0YQAvD_BwE

https://nodejs.dev/learn/writing-files-with-nodejs

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

https://www.alura.com.br/artigos/ordenacao-de-numeros-no-javascript-nao-funciona?gclid=CjwKCAiA6seQBhAfEiwAvPqu1wb_wnZ3ymSiSWHyYyWxjxoQAtrDKdgqd3dQquJ3qzQTCBM1Z47fNxoC2NAQAvD_BwE

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