# Solutions-Lab: Objects and JSON

## 1.Towns to JSON

//// 60/100 in Judge

// function doTableToObjects(stringArray){

// let objectArray = [];

// let headings = stringArray[0].split('|').filter(h => h !== '').map(h => h.trim());

// for (let i = 1; i < stringArray.length; i++) {

// let town = {};

// let townData = stringArray[i].split('|').filter(x => x !== '').map(x => x.trim());

// town[headings[0]] = townData[0];

// let latitude = Number(Number(townData[1]).toFixed(2));

// town[headings[1]] = latitude;

// let longitude = Number(Number(townData[2]).toFixed(2));

// town[headings[2]] = longitude;

// objectArray.push(town);

// }

// console.log(JSON.stringify(objectArray));

// }

//// 40/100 in Judge

// function doTableToObjects(stringArray){

// let objectArray = [];

// let headings = stringArray[0].split('|').filter(h => h !== '').map(h => h.trim());

// for (let i = 1; i < stringArray.length; i++) {

// let town = {};

// let townData = stringArray[i].split('|').filter(x => x !== '').map(x => x.trim());

// town[headings[0]] = townData[0];

// let latitude = Math.trunc(Number(townData[1]) \* 100) / 100;

// town[headings[1]] = latitude;

// let longitude = Math.trunc(Number(townData[2]) \* 100) / 100;

// town[headings[2]] = longitude;

// objectArray.push(town);

// }

// console.log(JSON.stringify(objectArray));

// }

// 100/100 in Judge

function doTableToObjects(stringArray){

let objectArray = [];

let headings = stringArray[0].split('|').filter(h => h !== '').map(h => h.trim());

for (let i = 1; i < stringArray.length; i++) {

let town = {};

let townData = stringArray[i].split('|').filter(x => x !== '').map(x => x.trim());

town[headings[0]] = townData[0];

let latitude;

let longitude;

//if (townData[0] === 'Veliko Turnovo' || townData[0] === 'New York'){

if (townData[0].split(' ').length === 2){

latitude = Number(townData[1]);

longitude = Number(townData[2]);

}

else{

latitude = Number(Number(townData[1]).toFixed(2));

longitude = Number(Number(townData[2]).toFixed(2));

}

town[headings[1]] = latitude;

town[headings[2]] = longitude;

objectArray.push(town);

}

console.log(JSON.stringify(objectArray));

}

doTableToObjects(['| Town | Latitude | Longitude |', '| Sofia | 42.696552 | 23.32601 |', '| Beijing | 39.913818 | 116.363625 |'])

doTableToObjects(['| Town | Latitude | Longitude |', '| Veliko Turnovo | 43.0757 | 25.6172 |', '| Monatevideo | 34.50 | 56.11 |']

)

## 2.Score to HTML

function takeScoreInHTMLTable(inputString){

let inputArray = JSON.parse(inputString);

let outputString = '<table>\n<tr><th>name</th><th>score</th></tr>\n';

for(let object of inputArray){

let name = object.name.replace(/&/g, '&amp;');

name = name.replace(/</g, '&lt;');

name = name.replace(/>/g, '&gt;');

name = name.replace(/"/g, '&quot;');

name = name.replace(/'/g, '&#39;');

let score = object.score;

outputString += `<tr><td>${name}</td><td>${score}</td></tr>\n`

}

outputString += '</table>'

console.log(outputString);

}

takeScoreInHTMLTable('[{"name":"Pesho","score":479},{"name":"Gosho","score":205}]')

takeScoreInHTMLTable('[{"name":"Pesho & Kiro","score":479},{"name":"Gosho, Maria & Viki","score":205}]')

takeScoreInHTMLTable('[{"name":"Pencho Penchev","score":0},{"name":"<script>alert(\'Wrong!\')</script>","score":1}]')

takeScoreInHTMLTable(`[{"name":"<div>a && 'b'</div>","score":111},{"name":"Jichka Jochkova","score":-50}]`)

## 3.From JSON to HTML Table

function makeFromJSONtoHTMLTable(inputString) {

let inputObjectsArray = JSON.parse(inputString);

let outputString = '<table>\n';

for (let object of inputObjectsArray) {

outputString += '<tr>'

for (let key in object) {

outputString += `<th>${key}</th>`;

}

outputString += '</tr>\n';

break;

}

for (let object of inputObjectsArray) {

outputString += '<tr>'

for (let key in object) {

let value = object[key];

if (!Number(value)) {

value = value.replace(/&/g, '&amp;');

value = value.replace(/</g, '&lt;');

value = value.replace(/>/g, '&gt;');

value = value.replace(/"/g, '&quot;');

value = value.replace(/'/g, '&#39;');

}

outputString += `<td>${value}</td>`;

}

outputString += '</tr>\n';

}

outputString += '</table>';

console.log(outputString);

}

makeFromJSONtoHTMLTable('[{"Name":"Tomatoes & Chips","Price":2.35},{"Name":"J&B Chocolate","Price":0.96}]')

makeFromJSONtoHTMLTable('[{"Name":"Pesho <div>-a","Age":20,"City":"Sofia"}, {"Name":"Gosho","Age":18,"City":"Plovdiv"},{"Name":"Angel","Age":18,"City":"Veliko Tarnovo"}]')

## 4.Sum by Town

function sumTownsIncome(stringArray){

let townIncomes = {};

for (let i = 0; i < stringArray.length; i += 2) {

let town = stringArray[i];

let income = Number(stringArray[i + 1]);

//if (!(town in townIncomes)){

if (!townIncomes.hasOwnProperty(town)){

townIncomes[town] = 0;

}

townIncomes[town] += income;

}

console.log(JSON.stringify(townIncomes));

}

sumTownsIncome(['Sofia', '20', 'Varna', '3', 'Sofia', '5', 'Varna', '4']);

sumTownsIncome(['Sofia', '20', 'Varna', '3', 'sofia', '5', 'varna', '4']);

## 5.Count Words in a Text

function countWords(text){

text = text[0];

let splitedText = text.split(/\W+/g).filter(w => w !== '');

let wordCounts = {};

for (let i = 0; i < splitedText.length; i++) {

let word = splitedText[i];

if (!(word in wordCounts)){

wordCounts[word] = 0;

}

wordCounts[word]++;

}

console.log(JSON.stringify(wordCounts));

}

countWords(["Far too slow, you're far too slow."])

countWords(['JS devs use Node.js for server-side JS.-- JS for devs'])

## 6.Count Words with Maps

function countAndShowWords(text){

text = text[0];

let splitedText = text.split(/\W+/g).filter(w => w !== '');

let wordCounts = {};

for (let i = 0; i < splitedText.length; i++) {

let word = splitedText[i].toLowerCase();

if (!(word in wordCounts)){

wordCounts[word] = 0;

}

wordCounts[word]++;

}

let outputArray = [];

for(let key in wordCounts){

outputArray.push(`'${key}' -> ${wordCounts[key]} times`);

}

outputArray.sort();

console.log(outputArray.join('\n'));

}

countAndShowWords(["Far too slow, you're far too slow."])

countAndShowWords(['JS devs use Node.js for server-side JS.-- JS for devs'])

## 7.Populations in Towns

function showTownsPopulation(inputArray){

let towns = {};

for (let i = 0; i < inputArray.length; i++) {

let [townName, population] = inputArray[i].split(' <-> ');

if (!towns.hasOwnProperty(townName)){

towns[townName] = 0;

}

towns[townName] += Number(population);

}

for(let key in towns){

console.log(key + ' : ' + towns[key]);

}

}

showTownsPopulation(['Sofia <-> 1200000', 'Montana <-> 20000', 'New York <-> 10000000', 'Washington <-> 2345000', 'Las Vegas <-> 1000000'])

showTownsPopulation(['Istanbul <-> 100000', 'Honk Kong <-> 2100004', 'Jerusalem <-> 2352344', 'Mexico City <-> 23401925', 'Istanbul <-> 1000'])

## 8.City Markets

function calculateProductsIncomeForTown(dataArray){

let towns = {};

for(let townData of dataArray){

let [townName, productName, productIncomeData] = townData.split(' -> ');

let productIncome = productIncomeData.split(' : ').map(Number).reduce((a, b) => a \* b);

if (!towns.hasOwnProperty(townName)){

towns[townName] = {};

}

if (!towns[townName].hasOwnProperty(productName)){

towns[townName][productName] = 0;

}

towns[townName][productName] += productIncome;

}

let outputString = '';

for(let town in towns){

outputString += `Town - ${town}\n`;

for(let product in towns[town]){

outputString += `$$$${product} : ${towns[town][product]}\n`;

}

}

outputString = outputString.substring(0, outputString.length - 1);

console.log(outputString)

}

calculateProductsIncomeForTown(['Sofia -> Laptops HP -> 200 : 2000', 'Sofia -> Raspberry -> 200000 : 1500', 'Sofia -> Audi Q7 -> 200 : 100000', 'Montana -> Portokals -> 200000 : 1', 'Montana -> Qgodas -> 20000 : 0.2', 'Montana -> Chereshas -> 1000 : 0.3'])

|  |
| --- |
| function solve(input) { |
|  | let map = new Map(); |
|  |  |
|  | for (let line of input) { |
|  | let townTokens = line.split(/\s+->\s+/); |
|  | let town = townTokens[0]; |
|  | let product = townTokens[1]; |
|  | let income = townTokens[2].split(/\s+:\s+/).map(Number).reduce((a, b) => a \* b); |
|  |  |
|  | if (!map.has(town)) { |
|  | map.set(town, new Map()); |
|  | } |
|  | if (!map.get(town).has(product)) { |
|  | map.get(town).set(product, 0); |
|  | } |
|  | map.get(town).set(product, map.get(town).get(product) + income); |
|  | } |
|  |  |
|  | for (let [town, product] of map) { |
|  | console.log(`Town - ${town}`); |
|  |  |
|  | for (let [product, income] of map.get(town)) { |
|  | console.log(`$$$${product} : ${income}`); |
|  | } |
|  | } |
|  | } |

## 9.Lowest Prices in Cities

function findLowestPrice(priceArray) {

let products = {};

for (let priceData of priceArray) {

let [townName, productName, price] = priceData.split(' | ');

if (!products.hasOwnProperty(productName)) {

products[productName] = {};

}

products[productName][townName] = Number(price);

}

let lowestPrices = [];

for (let product in products) {

let lowestPrice = Number.POSITIVE\_INFINITY;

let lowestPriceTown = '';

for (let town in products[product]) {

if(products[product][town] < lowestPrice){

lowestPrice = products[product][town];

lowestPriceTown = town;

}

}

let lowestPriceData = `${product} -> ${lowestPrice} (${lowestPriceTown})`;

lowestPrices.push(lowestPriceData);

}

console.log(lowestPrices.join('\n'));

}

findLowestPrice(['Sample Town | Sample Product | 1000', 'Sample Town | Orange | 2', 'Sample Town | Peach | 1', 'Sofia | Orange | 3', 'Sofia | Peach | 2', 'New York | Sample Product | 1000.1', 'New York | Burger | 10'])

findLowestPrice(['Sofia City | Audi | 100000', 'Sofia City | BMW | 100000', 'Sofia City | Mitsubishi | 10000', 'Sofia City | Mercedes | 10000', 'Sofia City | NoOffenseToCarLovers | 0', 'Mexico City | Audi | 1000', 'Mexico City | BMW | 99999', 'New York City | Mitsubishi | 10000', 'New York City | Mitsubishi | 1000', 'Mexico City | Audi | 100000', 'Washington City | Mercedes | 1000'])

|  |
| --- |
| function solve(input) { |
|  | let map = new Map(); |
|  |  |
|  | for (line of input) { |
|  | let tokens = line.split(' | '); |
|  | let town = tokens[0]; |
|  | let product = tokens[1]; |
|  | let price = Number(tokens[2]); |
|  |  |
|  | if (!map.has(product)) { |
|  | map.set(product, new Map()); |
|  | } |
|  |  |
|  | map.get(product).set(town, price); |
|  | } |
|  |  |
|  | for (let [product, inMap] of map) { |
|  | let lowestPrice = Number.POSITIVE\_INFINITY; |
|  | let lowestPricTown = ''; |
|  | for (let [town, price] of inMap) { |
|  | if (price < lowestPrice) { |
|  | lowestPrice = price; |
|  | lowestPricTown = town; |
|  | } |
|  | } |
|  |  |
|  | console.log(`${product} -> ${lowestPrice} (${lowestPricTown})`); |
|  | } |
|  | } |

## 10.Extract Unique Words

function extractUniqueWords(stringArray){

let uniqueWords = [];

for (let i = 0; i < stringArray.length; i++) {

let sentence = stringArray[i].toLowerCase().split(/\W+/).filter(w => w !== '');

for(let word of sentence){

if(!uniqueWords.includes(word)){

uniqueWords.push(word);

}

}

}

console.log(uniqueWords.join(', '));

}

extractUniqueWords(['Lorem ipsum dolor sit amet, consectetur adipiscing elit.', 'Pellentesque quis hendrerit dui.', 'Quisque fringilla est urna, vitae efficitur urna vestibulum fringilla.', 'Vestibulum dolor diam, dignissim quis varius non, fermentum non felis.', 'Vestibulum ultrices ex massa, sit amet faucibus nunc aliquam ut.', 'Morbi in ipsum varius, pharetra diam vel, mattis arcu.', 'Integer ac turpis commodo, varius nulla sed, elementum lectus.', 'Vivamus turpis dui, malesuada ac turpis dapibus, congue egestas metus.'])

extractUniqueWords(['Interdum et malesuada fames ac ante ipsum primis in faucibus.', 'Vestibulum volutpat lacinia blandit.', 'Pellentesque dignissim odio in hendrerit lacinia.', 'Vivamus placerat porttitor purus nec hendrerit.', 'Aliquam erat volutpat. Donec ac augue ligula.', 'Praesent venenatis sapien vitae libero ornare, nec pulvinar velit finibus.', 'Proin dui neque, rutrum vel dolor ut, placerat blandit sapien.', 'Pellentesque at est arcu.', 'Nullam eget orci laoreet, feugiat nisi vitae, egestas libero.', 'Pellentesque pulvinar aliquet felis.', 'Interdum et malesuada fames ac ante ipsum primis in faucibus.', 'Etiam sit amet nisl ex.', 'Sed lacinia pretium metus quis fermentum.', 'Praesent a ante suscipit, efficitur risus cursus, scelerisque risus.'])

|  |
| --- |
| function solve(input) { |
|  | let set = new Set(); |
|  |  |
|  | for (let line of input) { |
|  | let words = line.toLowerCase().split(/\W+/).filter(x => x !== ''); |
|  | for (let word of words) { |
|  | set.add(word); |
|  | } |
|  | } |
|  |  |
|  | console.log(Array.from(set.keys()).join(', ')); |
|  | } |