# GUVI : Zen Code-Sprint : JavaScript Practice problems in JSON(Objects) and List

If you practice all these problems you will be strong in JS objects manipulations. Before starting this have a look at the [basics workouts in JS](https://medium.com/@reach2arunprakash/guvi-zen-class-javascript-warm-up-programming-problems-15973c74b87f)

# Problem 0 : Part A (15 mins):

## Playing with JSON object’s Values:

Fluffy sorry, Fluffyy is my fav cat and it has 2 catFriends  
Write a code to get the below details of Fluffyy so that  
I can take him to vet.

var cat = {  
 name: ‘Fluffy’,  
 activities: [‘play’, ‘eat cat food’],  
 catFriends: [  
 {  
 name: ‘bar’,  
 activities: [‘be grumpy’, ‘eat bread omblet’],  
 weight: 8,  
 furcolor: ‘white’  
 },   
{  
 name: ‘foo’,  
 activities: [‘sleep’, ‘pre-sleep naps’],  
 weight: 3  
 }  
 ]  
}console.log(cat)

# **Basic Tasks to play with JSON**

1. Add height and weight to Fluffy

Ans) cat.height = ”20 inches”;

cat.weight = “8kg”;

1. Fluffy name is spelled wrongly. Update it to Fluffyy

Ans) cat.name="Fluffyy"

1. List all the activities of Fluffyy’s catFriends.

Ans) var l=cat.catFriends.length;

var activities =[];

for (var i=0;i<l;i++){

k=cat.catFriends[i].activities.length

for (var j=0;j<k;j++){

activities.push(cat.catFriends[i].activities[j]);

}

}

console.log(activities);

1. Print the catFriends names.

Ans) var l=cat.catFriends.length;

var names=[];

for (var i=0;i<l;i++){

names.push(cat.catFriends[i].name)

}

console.log(names);

1. Print the total weight of catFriends

Ans) var l=cat.catFriends.length;

var totalWeight=0;

for (var i=0;i<l;i++){

totalWeight= totalWeight + +cat.catFriends[i].weight;

}

console.log(totalWeight);

1. Print the total activities of all cats (op:6)

Ans) var l=cat.catFriends.length;

var activities = cat.activities;

for (var i=0;i<l;i++){

k=cat.catFriends[i].activities.length

for (var j=0;j<k;j++){

activities.push(cat.catFriends[i].activities[j]);

}

}

console.log(activities);

1. Add 2 more activities to bar & foo cats

Ans) cat.catFriends[0].activities.push("licking","scratching")

cat.catFriends[1].activities.push("biting","slapping")

1. Update the fur color of bar

Ans) cat.catFriends[0].furcolor="black";

# Problem 0 : Part B (15 mins):

## Iterating with JSON object’s Values

Above is some information about my car. As you can see, I am not the best driver.  
I have caused a few accidents.  
Please update this driving record so that I can feel better about my driving skills.

var myCar = {  
 make: ‘Bugatti’,  
 model: ‘Bugatti La Voiture Noire’,  
 year: 2019,  
 accidents: [  
 {  
 date: ‘3/15/2019’,  
 damage\_points: ‘5000’,  
 atFaultForAccident: true  
 },  
 {  
 date: ‘7/4/2022’,  
 damage\_points: ‘2200’,  
 atFaultForAccident: true  
 },  
 {  
 date: ‘6/22/2021’,  
 damage\_points: ‘7900’,  
 atFaultForAccident: true  
 }  
 ]  
}

1. Loop over the accidents array. Change atFaultForAccident from true to false.

Ans) var l=myCar.accidents.length;

for (var i=0;i<l;i++){

myCar.accidents[i].atFaultForAccident = false;

}

console.log(myCar)

2. Print the dated of my accidents

Ans) var l=myCar.accidents.length;

for (var i=0;i<l;i++){

console.log(myCar.accidents[i].date)

}

# **Real challenges starts here**

# :bowtie:

# **Problem 1 (5 mins):**

## **Parsing an JSON object’s Values:**

Write a function called “printAllValues” which returns an newArray of all the input object’s values.

Input (Object):

var object = {name: “RajiniKanth”, age: 33, hasPets : false};  
Output:

[“RajiniKanth”, 33, false]

Ans) var obj = {

name: "RajiniKanth",

age: 33,

hasPets : false

};

function printAllValues(obj){

console.log(Object.values(obj))

}

printAllValues(obj);

# Problem 2(5 mins) :

## Parsing an JSON object’s Keys:

Write a function called “printAllKeys” which returns an newArray of all the input object’s keys.

Example Input:  
{name : ‘RajiniKanth’, age : 25, hasPets : true}  
Example Output:  
[‘name’, ‘age’, ‘hasPets’]

Ans) var obj = {

name: "RajiniKanth",

age: 33,

hasPets : false

};

function printAllKeys(obj){

console.log(Object.keys(obj))

}

printAllKeys(obj);

# Problem 3( 7–9 mins):

## Parsing an JSON object and convert it to a list:

Write a function called “convertObjectToList” which converts an object literal into an array of arrays.  
Input (Object):  
var object = {name: “ISRO”, age: 35, role: “Scientist”};  
Output:  
[[“name”, “ISRO”], [“age”, 35], [“role”, “Scientist”]]

Ans) var obj = {

name: "ISRO",

age: 35,

role : "scientist"

};

function convertObjectToList(obj){

console.log(Object.entries(obj))

}

convertObjectToList(obj);

# Problem 4( 5 mins):

## Parsing a list and transform the first and last elements of it:

Write a function ‘transformFirstAndLast’ that takes in an array, and returns an object with:  
1) the first element of the array as the object’s key, and  
2) the last element of the array as that key’s value.  
Input (Array):  
var array = [“GUVI”, “I”, “am”, “Geek”];  
Output:  
var object = {  
 GUVI : “Geek”  
}

Ans) var arr = ["GUVI", "I", "am", "Geek"];

var obj={};

function transformFirstAndLast(arr){

obj[arr[0]]=arr[3]

console.log(obj);

}

transformFirstAndLast(arr)

# Problem 5 ( 7 -9 mins):

## Parsing a list of lists and convert into a JSON object:

Write a function “fromListToObject” which takes in an array of arrays, and returns an object with each pair of elements in the array as a key-value pair.  
Input (Array):  
var array = [[“make”, “Ford”], [“model”, “Mustang”], [“year”, 1964]];  
Output:  
var object = {  
make : “Ford”  
model : “Mustang”,  
year : 1964  
}

Ans) var array = [['make', 'Ford'], ['model', 'Mustang'], ['year', 1964]];

var newObj={}

function fromListToObject(arr){

for (var i=0;i<arr.length;i++){

for (var j=0;j<arr[i].length;j=j+2){

newObj[arr[i][j]]=arr[i][j+1]

}

}

console.log(newObj)

}

fromListToObject(array)

# Problem 6 (10 mins):

## Parsing a list of lists and convert into a JSON object:

Write a function called “transformGeekData” that transforms some set of data from one format to another.

Input (Array):  
var array = [[[“firstName”, “Vasanth”], [“lastName”, “Raja”], [“age”, 24], [“role”, “JSWizard”]], [[“firstName”, “Sri”], [“lastName”, “Devi”], [“age”, 28], [“role”, “Coder”]]];  
Output:  
[  
{firstName: “Vasanth”, lastName: “Raja”, age: 24, role: “JSWizard”},  
{firstName: “Sri”, lastName: “Devi”, age: 28, role: “Coder”}  
]

Ans) var newArr=[];

function fromListToObject(arr){

for (var i=0;i<arr.length;i++){

let newObj={}

for (var j=0;j<arr[i].length;j++){

for (var k=0;k<arr[i][j].length;k=k+2){

newObj[arr[i][j][k]]=arr[i][j][k+1]

}

}

newArr.push(newObj)

}

console.log(newArr)

}

fromListToObject(array)

# Problem 7 (10 — 20 mins):

## Parsing two JSON objects and Compare:

Write an “assertObjectsEqual” function from scratch.  
Assume that the objects in question contain only scalar values (i.e., simple values like strings or numbers).  
It is OK to use JSON.stringify().  
Note: The examples below represent different use cases for the same test. In practice, you should never have multiple tests with the same name.  
Success Case:  
Input:  
var expected = {foo: 5, bar: 6};  
var actual = {foo: 5, bar: 6}  
assertObjectsEqual(actual, expected, ‘detects that two objects are equal’);  
Output:  
Passed  
Failure Case:  
Input:var expected = {foo: 6, bar: 5};  
var actual = {foo: 5, bar: 6}  
assertObjectsEqual(actual, expected, ‘detects that two objects are equal’);  
Output:  
FAILED [my test] Expected {“foo”:6,”bar”:5}, but got {“foo”:5,”bar”:6}

Ans) var expected = {foo: 5, bar: 6};

var actual = {foo: 6, bar: 5}

function assertObjectsEqual(actual, expected, testName){

if (JSON.stringify(expected)===JSON.stringify(actual)){

console.log("Passed")

}else {

console.log("Failed")

}

}

assertObjectsEqual(actual, expected,"detects that two objects are equal")

# Problem 8(10 mins):

## Parsing JSON objects and Compare:

I have a mock data of security Questions and Answers. You function should take the object and a pair of strings and should return if the quest is present and if its valid answer

var securityQuestions = [  
 {  
 question: “What was your first pet’s name?”,  
 expectedAnswer: “FlufferNutter”  
 },  
 {  
 question: “What was the model year of your first car?”,  
 expectedAnswer: “1985”  
 },  
 {  
 question: “What city were you born in?”,  
 expectedAnswer: “NYC”  
 }  
]function chksecurityQuestions(securityQuestions,question) {  
  
 // your code here return true or false;   
}//Test case1:var ques = “What was your first pet’s name?”;  
var ans = “FlufferNutter”;var status = chksecurityQuestions(securityQuestions, ques, ans);console.log(status); // true//Test case2:var ques = “What was your first pet’s name?”;  
var ans = “DufferNutter”;var status = chksecurityQuestions(securityQuestions, ques, ans);console.log(status); // flase

Ans) functionchksecurityQuestions(securityQuestions,question,answer) {

for (var i=0;i<securityQuestions.length;i++){

if (securityQuestions[i].question===question && securityQuestions[i].expectedAnswer===answer) {

return true

}else {

return false

}

}

}

# Problem 9(20 mins):

## Parsing JSON objects and Compare:

Write a function to return the list of characters below 20 age

var students = [  
 {  
 name: “Siddharth Abhimanyu”, age: 21}, { name: “Malar”, age: 25},  
 {name: “Maari”,age: 18},{name: “Bhallala Deva”,age: 17},  
 {name: “Baahubali”,age: 16},{name: “AAK chandran”,age: 23},{name:“Gabbar Singh”,age: 33},{name: “Mogambo”,age: 53},  
 {name: “Munnabhai”,age: 40},{name: “Sher Khan”,age: 20},  
 {name: “Chulbul Pandey”,age: 19},{name: “Anthony”,age: 28},  
 {name: “Devdas”,age: 56}   
];

function returnMinors(arr){

}

console.log(returnMinors(students));

Ans) function returnMinors(arr){

let list=[];

for (var i=0;i<students.length;i++){

if (students[i].age<20){

list.push(students[i].name)

}

}

console.log(list)

}

returnMinors(students);