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

* Add height and weight to Fluffy

Answer:

Cat.height = 20

Cat.weight = 40

* Fluffy name is spelled wrongly. Update it to Fluffyy

Ans: cat.name=‘Fluffyy’

* List all the activities of Fluffyy’s catFriends.

Ans: cat.catFriends.map(item => {

item.activities.map(item2 => {

console.log(item2)

})

})

* Print the catFriends names.

Ans: cat.catFriends.map(item => {

console.log(item.name)

})

* Print the total weight of catFriends

Ans:

Let totalWeight = 0

cat.catFriends.map(item => {

totalWeight+= item.weight

})

Console.log(totalWeight)

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

Ans:

cat.activities.map(item => {

console.log(item)

})

cat.catFriends.map(item => {

item.activities.map(item2 => {

console.log(item2)

})

})

* Add 2 more activities to bar & foo cats

Ans:

Let temp1 = Cat.catFriends[0].activities

Let temp2 = Cat.catFriends[1].activities

Temp1.push(‘sleep’)

Temp1.push(‘walk’)

Temp2.push(‘sleep’)

Temp2.push(‘walk’)

Cat.catFriends[0].activities = temp1

Cat.catFriends[1].activities = temp2

* Update the fur color of bar

Ans: Cat.catFriends[0].furColor = ‘blue’

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: myCar.accidents.map((item, index) => {

myCar.accidents[index].atFaultForAccident = false

})

2. Print the dated of my accidents

Ans: myCar.accidents.map((item, index) => {

console.log(item.date)

})

# 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:

function printAllValues(obj){

Let newArray = Object.keys(obj).map(item => {

return obj[item]

})

return newArray

}

# 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:

function printAllKeys(obj){

Let newArray = Object.keys(obj).map(item => {

return item

})

return newArray

}

# 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:

function convertObjectToList(obj){

Let newArray = Object.keys(obj).map(item => {

Let tempArray = []

tempArray.push(item)

tempArray.push(obj[item])

return tempArray

})

return newArray

}

# 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:

function ‘transformFirstAndLast’(arr){

Let obj = {}

Obj[arr[0]] = arr[arr.length-1]

return obj

}

# 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:

function fromListToObject(arr){

Let obj = {}

arr.map(item => {

Obj[item[0]] = item[1]

})

return obj

}

# 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:

function transformGeekData(arr){

Let obj = {}

Let tempArray = arr1.map(arr2 => {

Let obj = {}

arr2.map(item => {

Obj[item[0]] = item[1]

})

Return obj

})

return tempArray

}

# Problem 7 (10 — 20 mins):

## Parsing two JSON objects and Compare:

Read this : <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON/stringify>

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:

function assertObjectsEqual(obj1, obj2){

if(JSON.stringify(obj1) === JSON.stringify(obj2){

console.log(‘passed’)

} else {

console.log(‘failed: expected ’+obj1+’ but got ‘+obj2)

}

}

# 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

Ans:

function chksecurityQuestions(secQues, que, ans){

if(secQues.findIndex(item => item.question == que && item.expectedAnswer == ans) != -1){

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

Ans:

function returnMinors(arr){

arr.filter(item => item.age < 20).map(item => {

console.log(item.name)

})

}