ASSIGNMENT

**Problem 0: Part A –**

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);

**Question 1:** Add height and weight to Fluffy.

**Answer:** cat.height=50;

cat.weight=28;

**Question 2:** Fluffy name is spelled wrongly. Update it to Fluffyy.

**Answer:** cat.name= “Fluffyy”;

**Question 3:** List all the activities of Fluffyy’s catFriends.

**Answer:**  cat.catFriends.forEach((element) =>{ element.activities.forEach((value)=>{

console.log(value);

})

});

**Question 4:** Print the catFriends names.

**Answer:**  cat.catFriends.forEach((element)=>{

console.log(element.name); });

**Question 5:** Print the total weight of catFriends.

**Answer:** var tot=cat.catFriends.reduce((prv,current)=>{

return (prv+current.weight);

},0);

console.log(tot);

**Question 6:** Print the total activities of all cats.

**Answer:** var tot=cat.activities.length+(cat.catFriends.reduce((prv,current)=>{

return (prv+current.activities.length);

},0));

console.log(tot);

**Question 7:** Add 2 more activities to bar & foo cats

**Answer:** cat.catFriends.find((element)=>{ return element.name==="bar"}).activities.push("Drinking milk","Walking");

cat.catFriends.find((element)=>{ return element.name==="foo"}).activities.push("Sitting","Standing");

**Question 8:** Update the fur color of bar

**Answer:** cat.catFriends.find((element)=>{ return element.name==="bar"}).furcolor="black";

**Problem 0: Part B –**

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  
}  
]  
}

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

**Answer:** myCar.accidents.forEach((element)=>{

element.atFaultForAccident=false;

});

**Question 2.** Print the dates of my accidents

**Answer:** myCar.accidents.forEach((element)=>{

console.log(element.date);

});

**Problem 1:**

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

Input (Object):

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

[“RajiniKanth”, 33, false]

**Answer:** var obj = {name:'RajiniKanth', age:33, hasPets:false};

function printAllValues(obj) {

return (Object.values(obj));

}

console.log(printAllValues(obj));

**Problem 2:**

**Question:** 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’]

**Answer:** var obj = {name:'RajiniKanth', age:25, hasPets:true};

function printAllKeys(obj) {

return (Object.keys(obj));

}

console.log(printAllKeys(obj));

**Problem 3:**

**Question:** 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”]]

**Answer:**

var obj = {name: 'ISRO', age: 35, role: 'Scientist'};

function convertListToObject(obj) {

var List=[];

Object.keys(obj).forEach((element)=>{

List.push([element,obj[element]]);

});

console.log(List);

}

convertListToObject(obj);

**Problem 4:**

**Question:** 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”  
}

**Answer:** var arr = ['GUVI', 'I', 'am', 'a geek'];

function transformFirstAndLast(arr) {

var newObj={};

newObj[arr[0]]=arr[arr.length-1];

return newObj;

}

console.log(transformFirstAndLast(arr));

**Problem 5:**

**Question:** 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  
}

**Answer:** var arr = [['make', 'Ford'], ['model', 'Mustang'], ['year', 1964]];

function fromListToObject(arr) {

var newObject = {};

arr.forEach((element)=>{

newObject[element[0]]=element[1];

});

return newObject;

}

console.log(fromListToObject(arr));

**Problem 6:**

**Question:** 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”}  
]

**Answer:**

var arr= [[['firstName', 'Vasanth'], ['lastName', 'Raja'], ['age', 24], ['role', 'JSWizard']], [['firstName', 'Sri'], ['lastName', 'Devi'], ['age', 28], ['role', 'Coder']]];

function transformEmployeeData(arr) {

var tranformEmployeeList = [];

tranformEmployeeList=arr.map((element)=>{

var Employee={};

element.forEach((value)=>{

Employee[value[0]]=value[1];

});

return Employee;

});

return tranformEmployeeList;

}

console.log(transformEmployeeData(arr));

**Problem 7:**

**Question:** 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}

**Answer:**

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

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

function assertObjectsEqual(actual, expected, testName){

var act=JSON.stringify(actual);

var exp=JSON.stringify(expected);

if(act===exp){

console.log("Passed");

}

else{

console.log("FAILED "+testName+" Expected "+exp+", but got "+act);

}

}

assertObjectsEqual(actual,expected,"the Object equality test");

**Problem 8**

**Question:**

I have a mock data of security Questions and Answers. Your function should take the object and a pair of strings and should return if the question and it’s valid answer are present.

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”  
}  
]

**Answer:**

var question="What was your first pet’s name?";

var answer="FlufferNutter";

function chksecurityQuestions(securityQuestions,question,answer) {

if(securityQuestions.findIndex((element)=>{return (element.question===question&&element.expectedAnswer===answer)}) !== -1){

return true;

}

else{

return false;

}

}

console.log(chksecurityQuestions(securityQuestions,question,answer));

**Problem 9:**

**Question:** Write a function to return the list of characters below 20 age.

**Answer:** 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)

{

var studentArray=arr.filter((element)=>{

return (element.age<20);

});

return studentArray;

}

console.log(returnMinors(students));