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
2. Fluffy name is spelled wrongly. Update it to Fluffyy
3. List all the activities of Fluffyy’s catFriends.
4. Print the catFriends names.
5. Print the total weight of catFriends
6. Print the total activities of all cats (op:6)
7. Add 2 more activities to bar & foo cats
8. Update the fur color of bar

Basic Tasks to play with JSON

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

   ANSWERS:

```
cat.height=20;
cat.weight=8;
console.log(cat.height);
console.log(cat. weight); // console.log(cat.height, cat.weight);
```

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

   ANSWERS:

```
cat.name="Fluffyy"
console.log(cat.name);
```

3. **List all the activities of Fluffyy's catFriends**

   ANSWERS:

```
1)  console.log(cat.catFriends[0].activities, cat.catFriends[1].activities );
```

OR

```
2)  console.log(cat.catFriends[0].activities);
    console.log(cat.catFriends[1].activities);
```

4. **Print the catFriends names.**

   ANSWERS:

```
console.log(cat.catFriends[0].name, cat.catFriends[1].name );
```

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

ANSWERS:

```
var totalweight =cat.catFriends[0].weight + cat.catFriends[1].weight ;

console.log(totalweight);
```

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

ANSWERS:

```
console.log(cat.activities.length +
 cat.catFriends[0].activities.length + cat.catFriends[1].activities.length );
```

7. **Add 2 more activities to bar & foo cats**

ANSWERS:

```
//ADDED TWO ACTIVITIESAS BELOW
(cat.catFriends[0].activities).push("roaming around", "jumping from chair");
(cat.catFriends[1].activities).push("Screaming around", "running inside home");
//PRINT O/P   (1)
console.log(cat.catFriends[0].activities);
console.log(cat.catFriends[1].activities);
                    OR
//PRINT O/P   (2)
console.log(cat.catFriends[0].activities , cat.catFriends[1].activities);
```

8. **Update the fur color of bar.**

ANSWERS:

```
cat.catFriends[0].furcolor = "Brown"
// only to print color
console.log(cat.catFriends[0].furcolor);
// to print the entire keys & values
console.log(cat.catFriends);
```

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.

   ANSWERS:

```
for ( let i=0; i<myCar.accidents.length; i++){

   myCar.accidents[i].atFaultForAccident=false;

}



console.log(myCar.accidents);
```

2. Print the dated of my accidents

ANSWERS:

```
for ( let i=0; i<myCar.accidents.length; i++)
{
console.log(myCar.accidents[i].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]

Sample Function proto:

```
var obj = {name : "RajiniKanth", age : 33, hasPets : false};function
printAllValues(obj) {
 // your code here
}
```

ANSWER:

```
function printAllValues (y) {
    return ( Object.values(y) );
}
```

// To print the entirevalues in a object

console.log(printAllValues(obj)); ➔ o/p : ["RajiniKanth", 33, false]

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']

Sample Function proto:

```
function printAllKeys(obj) {
 // your code here
}
```

ANSWER:

```
        function printAllKeys (x) {
            return ( Object.keys(x) );
        }
        // To print the  entirevalues in a object
    console.log(printAllKeys(obj)); ➔ O/P: ['name', 'age', 'hasPets']
```

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”]]

Sample Function proto:

```
var obj = {name: “ISRO”, age: 35, role: “Scientist”};
function convertListToObject(obj) {
 // your code here
}
```

ANSWER:

```
function convertListToObject(x)
{    // To convert an object to an array, we’ll use the .entries() method
   return (Object.entries(x))
}
   // print the function
console.log(convertListToObject(obj));
```

0/P: [['name', 'ISRO'], ['age', 35], ['role', 'Scientist']]

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”

}

Sample Function proto:

```
var arr = [“GUVI”, “I”, “am”, “a geek”];function
transformFirstAndLast(arr) {

 return newObject;
}
```

ANSWER:

```
function transformFirstAndLast(x) {
var newObject = x.reduce((acc,curr)=> (acc[arr[0]]=arr[3],acc),{});
return (newObject);
}
console.log(transformFirstAndLast(arr));  O/P➔ var object = {GUVI : “Geek”}
```

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}

Sample Function proto:

```
var arr = [[“make”, “Ford”], [“model”, “Mustang”], [“year”,
1964]];function fromListToObject(arr) {
 var newObject = {};

 return newObject;
}
```

ANSWER:

```
function fromListToObject(x){
   var obj={};
   for (var i=0; i<x.length;i++){
      obj[x[i][0]]=x[i][1]
   }
   return obj;
}
console.log(fromListToObject(arr));
```

o/p➔{ make: 'Ford', model: 'Mustang', year: 1964 }

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"]]];
[
{firstName: "Vasanth", lastName: "Raja", age: 24, role: "JSWizard"},
{firstName: "Sri", lastName: "Devi", age: 28, role: "Coder"}
]Output:

ANSWERS:

```
function transformGeekData(x)

{    var box = [];

  for(let i=0; i<x.length; i++) //FIRST LOOP FOR ARRAY LENGTH

{

 var obj = {};      //SECOND LOOP FOR ARRAY INDEX LENGTH

    for(let j=0;j<x[i].length; j++){

        obj[x[i][j][0]]=x[i][j][1];

      }

               box.push(obj);   //PUSH IN TO box

}

return(box);  //FINAL OUTPUT

}

console.log(transformGeekData(arr));
```

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}

```
var expected = {foo: 5, bar: 6};
var actual = {foo: 5, bar: 6}function assertObjectsEqual(actual, expected, testName){
 // your code here
}
```

ANSWER:

function assertObjectsEqual(actual,expected,testName){

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

console.log("Passed "+testName,"Expected"+JSON.stringify(actual),"and got" + JSON.stringify(expected))

 }else{console.log("Failed "+testName,"Expected"+JSON.stringify(actual),"but got" + JSON.stringify(expected))

}}

assertObjectsEqual(actual , expected,"Comparison");

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'   }  ]
```

ANSWER:

```
function chksecurityQuestions(securityQuestions,question,ans) {
   for(var i in securityQuestions){
     if (question == securityQuestions[i].question ){
       if(ans == securityQuestions[i].expectedAnswer){
         return true;
       }}
      else{ return 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); //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))
```

ANSWER:

```
function returnMinors(x){
    var  agebelow20 = x.filter(function (y)
    {
       return y.age <20;
    });
    console.log(agebelow20);
    }
returnMinors(students);
```