1) **Version's of HTTP .**

HTTP/0.9 - one line protocol used to transfer plain HTML file

HTTPS - Netscape created https to be used with SSL for its Browser

HTTP/1.0 - concept of header, version information , status codes were introduced

HTTP/1.1 - Introduced persistent connection ,pipelining ,cache control and many other features

HTTP/2 - Based on Google's SPDY allows multiplexing and server push

HTTP/3 - Based on Google's QUIC that user UDP instead of TCP

HTTP/1.1 uses text based request to a server by calling a method like GET or POST .

In response server sends a resource like an HTML page back to the client

HTTP/2 is developed by google primarily with the intention of reducing webpage load latency by using techniques such as compression, multiplexing and prioritization .

But we still use HTTP/1.1 .HTTP/2 still not yet fully developed .

2) **5 difference between Browser JS and NODE JS**

Browser JS:

- Browser JS has Window global class .
- It is basically used on the client side
- It is an upgraded version of ECMA script .
- It can run in any browser engine as like JS core in safari and Spidermonkey in Firefox.
- Browser JS is used in Frontend development

NODEJS

- It doesn’t has Window global class
- It is used in Server side development
- Node js run on V8 engine of google chrome
- It has the functionality of file management
- NodeJS is a javascript runtime environment

3) **what happen when you type a URL in the address bar in the browser?**

- when we hit the url in the address bar . URL is converted into IP address by DNS server.
- Then the request is acknowledged by server and provided with the requested files like HTML , CSS and JS files
- HTML and CSS file are processed by Rendering engine in the browser and DOM tree is formed
- And then rendering tree is constructed ,it is processed to form layout after which painting process executed .
- JS file is processed by respective js engine of the browser .
- After these processes we could the see the requested info in our browser .

4) **Write a write up on Difference between copy by value and copy by reference .**

Copy by value:

- Copy by value usually works on primitive data type .
- If one variable is assigned as the value of other example : var y =10 : var v =y;
- In this case value of y is copied to v .so v has the value of 10 ;
- Both are independent , changes in either one does not affect other .

Copy by Reference

- Copy by reference works on composite data type .
- In this case if we assign one variable as value for other, reference of one is copied to another one
- So both are dependent .
- Because both holds the address of the value .
- If any changes on one will reflect on other .

5) **How to copy by value a composite data type (array + objects) ?**

There are 3 ways to copy by value for composite data types.

1. Using the spread (…) operator
2. Using the object.assign() method
3. Using the JSON.stringify() and JSON.parse() methods

JSON Task

Playing with JSON object's Values:

Basic Tasks to play with JSON

1. Add height and weight to Fluffy
2. Fluffy name is spelled wrongly. Update it to Fluffy

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

Code :

```
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
 }
 ]
}
cat.height = "30cm";
cat.weight =  "10kg";
cat.name = "Fluffyy";
console.log(cat.catFriends[0].activities.concat(cat.catFriends[1].activities).join(" "));
console.log(cat.catFriends[0].name,cat.catFriends[1].name);
console.log(cat.catFriends[0].weight+cat.catFriends[1].weight);
console.log(cat.activities.concat(cat.catFriends[0].activities.concat(cat.catFriends[1].activities
    )).join(" "));
cat.catFriends[0].activities.push("play","jump");
cat.catFriends[1].activities.push("play","jump");
cat.catFriends[0].furcolor="brown";
console.log(cat);
```

Output :

Output:

```
be grumpy eat bread omblet sleep pre-sleep naps
bar foo
11
play eat cat food be grumpy eat bread omblet sleep pre-sleep naps
{ name: 'Fluffyy',
  activities: [ 'play', 'eat cat food' ],
  catFriends:
   [ { name: 'bar', activities: [Array], weight: 8, furcolor: 'brown' },
     { name: 'foo', activities: [Array], weight: 3 } ],
  height: '30cm',
  weight: '10kg' }
```

Iterating with JSON object's Values

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

2. Print the dated of my accidents

Code :

```
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
 }
 ]
}

for(var i in myCar.accidents)
{
    myCar.accidents[i].atFaultForAccident = false ;
    console.log(myCar.accidents[i].date);
}
console.log(myCar);
```

Output :

```
3/15/2019
7/4/2022
6/22/2021
{ make: 'Bugatti',
  model: 'Bugatti La Voiture Noire',
  year: 2019,
  accidents:
   [ { date: '3/15/2019',
       damage_points: '5000',
       atFaultForAccident: false },
     { date: '7/4/2022',
       damage_points: '2200',
       atFaultForAccident: false },
     { date: '6/22/2021',
       damage_points: '7900',
       atFaultForAccident: false } ] }
```

Parsing an JSON object's Values:

Write a function called "printAllValues" which returns an newArray of all the input object's values.

Code :

```
var object = {name:"RajiniKanth", age: 33, hasPets : false};
var result = [];
for(var i in object)
{
    result.push(object[i]);
}
console.log(result);
```

Output:

```
Output:
[ 'RajiniKanth', 33, false ]
```

Parsing an JSON object's Keys:

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

Code:

```
var object = {name:"RajiniKanth", age: 33, hasPets : false};
var result = printAllKeys(object);

function printAllKeys(obj) {
    var temp = [] ;
for(var i in object)
{
    temp.push(object[i]);
}
    return temp;
}
console.log(result);
```

Output:

Output:

```
[ 'RajiniKanth', 33, false ]
```

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.

```
var obj = {name: "ISRO", age: 35, role: "Scientist"}
var result = convertListToObject(obj);
function convertListToObject(obj) {
    var res = [];
    for(var i in obj)
    {
        var t = [];
        t.push(i);
        t.push(obj[i])
        res.push(t);
    }
    return res;
}
console.log(result);
```

Output:

```
Output:
[ [ 'name', 'ISRO' ], [ 'age', 35 ], [ 'role', 'Scientist' ] ]
```

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.

Code:

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

 newObject = {};
 newObject[arr[0]] = arr[3];

 return newObject;
}
console.log(transformFirstAndLast(arr))
```

Output :

```
Output:
{ GUVI: 'a geek' }
```

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.

Code :

```
var arr = [["make", "Ford"], ["model", "Mustang"], ["year", 1964]];

function fromListToObject(arr) {
 var newObject = {};
 for(var i in arr){
 newObject[arr[i][0]] = arr[i][1];
    //console.log(arr[i][1]);
 }
 return newObject;
}
console.log(fromListToObject(arr))
```

Output:

Output:

```
{ make: 'Ford', model: 'Mustang', year: 1964 }
```

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.

Code :

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

function transformGeekData(arr) {
  var res = [];
 for(var i in arr){
     var newObject = {};
  for(var j in arr[i])
  {
    newObject[arr[i][j][0]] = arr[i][j][1];
   // console.log(arr[i][j][0]);
  }
  res.push(newObject);    //console.log(arr[i][1]);
 }
 return res;
}
console.log(transformGeekData(arr))
```

Output :

Output:

```
[ { firstName: 'Vasanth',
    lastName: 'Raja',
    age: 24,
    role: 'JSWizard' },
  { firstName: 'Sri', lastName: 'Devi', age: 28, role: 'Coder' } ]
```

Parsing two JSON objects and Compare:

Write an “assertObjectsEqual” function from scratch.

```
var expected = {foo: 5, bar: 6};
var actual = {foo: 5, bar: 6}
function assertObjectsEqual(actual, expected, testName){
 var count = 0 ;
 var total = 0 ;
 for(var i in expected)
 {
      total++;
      if(i==i)
      {
     if(actual[i]===expected[i])
     {
         count++;
     }}
 }

 if(count==total)
 {
     console.log("Passed");
 }
 else
 {
     console.log("Failed");
 }

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

Output:

Output:

```
Passed
```

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

Code :

```
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,answer) {
 for(var i in securityQuestions)
 {
     if(securityQuestions[i].question==(question))
     {
         if(securityQuestions[i].expectedAnswer==(answer))
         {
            return true
         }
     }
 }
 return false ;
}
var ques = "What was your first pet's name?";
var ans  =  "FlufferNutter";
var status = chksecurityQuestions(securityQuestions, ques, ans);
console.log(status);
```

Output:

Output:

```
true
false
```

Parsing JSON objects and Compare:

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

Code :

```
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 res= [];
 for(var i in arr)
 {
     if(arr[i].age<20)
     {
         res.push(arr[i].name);
     }
 }
 return res;
}
console.log(returnMinors(students));
```

Output :

Output:

```
[ 'Maari', 'Bhallala Deva', 'Baahubali', 'Chulbul Pandey' ]
```

Try the rest countries api. Extract and print the total population of all the countries in the console .

Code :

```
JS script.js > ...
1   var connect = new XMLHttpRequest();
2   connect.open('GET',"https://restcountries.eu/rest/v2/all",true);
3   connect.send();
4   connect.onload = function()
5   {
6       var data = JSON.parse(this.response);
7       var sum = 0 ;
8       for(var i in data)
9       {
10        sum =sum+data[i].population ;
11      }
12      console.log(sum);
13
14
15  }
16
```

Output :