**Task 1;**

**Html code:**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<title>sample api data</title>**

**</head>**

**<body>**

**<script src = "script.js"> </script>**

**</body>**

**</html>**

**Script.js code:**

**var request = new XMLHttpRequest();**

**request.open('GET','https://restcountries.eu/rest/v2/all',true);**

**request.send();**

**request.onload = function() {**

**var data = JSON.parse(this.response)**

**for(var i in data)**

**{**

**console.log(data[i].name);**

**}**

**}**

**OUTPUT:**

Afghanistan

script.js:10 Åland Islands

script.js:10 Albania

script.js:10 Algeria

script.js:10 American Samoa

script.js:10 Andorra

script.js:10 Angola

script.js:10 Anguilla

script.js:10 Antarctica

script.js:10 Antigua and Barbuda

script.js:10 Argentina

script.js:10 Armenia

script.js:10 Aruba

script.js:10 Australia

script.js:10 Austria

script.js:10 Azerbaijan

script.js:10 Bahamas

script.js:10 Bahrain

script.js:10 Bangladesh

script.js:10 Barbados

script.js:10 Belarus

script.js:10 Belgium

script.js:10 Belize

script.js:10 Benin

script.js:10 Bermuda

script.js:10 Bhutan

script.js:10 Bolivia (Plurinational State of)

script.js:10 Bonaire, Sint Eustatius and Saba

script.js:10 Bosnia and Herzegovina

script.js:10 Botswana

script.js:10 Bouvet Island

script.js:10 Brazil

script.js:10 British Indian Ocean Territory

script.js:10 United States Minor Outlying Islands

script.js:10 Virgin Islands (British)

script.js:10 Virgin Islands (U.S.)

script.js:10 Brunei Darussalam

script.js:10 Bulgaria

script.js:10 Burkina Faso

script.js:10 Burundi

script.js:10 Cambodia

script.js:10 Cameroon

script.js:10 Canada

script.js:10 Cabo Verde

script.js:10 Cayman Islands

script.js:10 Central African Republic

script.js:10 Chad

script.js:10 Chile

script.js:10 China

script.js:10 Christmas Island

script.js:10 Cocos (Keeling) Islands

script.js:10 Colombia

script.js:10 Comoros

script.js:10 Congo

script.js:10 Congo (Democratic Republic of the)

script.js:10 Cook Islands

script.js:10 Costa Rica

script.js:10 Croatia

script.js:10 Cuba

script.js:10 Curaçao

script.js:10 Cyprus

script.js:10 Czech Republic

script.js:10 Denmark

script.js:10 Djibouti

script.js:10 Dominica

script.js:10 Dominican Republic

script.js:10 Ecuador

script.js:10 Egypt

script.js:10 El Salvador

script.js:10 Equatorial Guinea

script.js:10 Eritrea

script.js:10 Estonia

script.js:10 Ethiopia

script.js:10 Falkland Islands (Malvinas)

script.js:10 Faroe Islands

script.js:10 Fiji

script.js:10 Finland

script.js:10 France

script.js:10 French Guiana

script.js:10 French Polynesia

script.js:10 French Southern Territories

script.js:10 Gabon

script.js:10 Gambia

script.js:10 Georgia

script.js:10 Germany

script.js:10 Ghana

script.js:10 Gibraltar

script.js:10 Greece

script.js:10 Greenland

script.js:10 Grenada

script.js:10 Guadeloupe

script.js:10 Guam

script.js:10 Guatemala

script.js:10 Guernsey

script.js:10 Guinea

script.js:10 Guinea-Bissau

script.js:10 Guyana

script.js:10 Haiti

script.js:10 Heard Island and McDonald Islands

script.js:10 Holy See

script.js:10 Honduras

script.js:10 Hong Kong

script.js:10 Hungary

script.js:10 Iceland

script.js:10 India

script.js:10 Indonesia

script.js:10 Côte d'Ivoire

script.js:10 Iran (Islamic Republic of)

script.js:10 Iraq

script.js:10 Ireland

script.js:10 Isle of Man

script.js:10 Israel

script.js:10 Italy

script.js:10 Jamaica

script.js:10 Japan

script.js:10 Jersey

script.js:10 Jordan

script.js:10 Kazakhstan

script.js:10 Kenya

script.js:10 Kiribati

script.js:10 Kuwait

script.js:10 Kyrgyzstan

script.js:10 Lao People's Democratic Republic

script.js:10 Latvia

script.js:10 Lebanon

script.js:10 Lesotho

script.js:10 Liberia

script.js:10 Libya

script.js:10 Liechtenstein

script.js:10 Lithuania

script.js:10 Luxembourg

script.js:10 Macao

script.js:10 Macedonia (the former Yugoslav Republic of)

script.js:10 Madagascar

script.js:10 Malawi

script.js:10 Malaysia

script.js:10 Maldives

script.js:10 Mali

script.js:10 Malta

script.js:10 Marshall Islands

script.js:10 Martinique

script.js:10 Mauritania

script.js:10 Mauritius

script.js:10 Mayotte

script.js:10 Mexico

script.js:10 Micronesia (Federated States of)

script.js:10 Moldova (Republic of)

script.js:10 Monaco

script.js:10 Mongolia

script.js:10 Montenegro

script.js:10 Montserrat

script.js:10 Morocco

script.js:10 Mozambique

script.js:10 Myanmar

script.js:10 Namibia

script.js:10 Nauru

script.js:10 Nepal

script.js:10 Netherlands

script.js:10 New Caledonia

script.js:10 New Zealand

script.js:10 Nicaragua

script.js:10 Niger

script.js:10 Nigeria

script.js:10 Niue

script.js:10 Norfolk Island

script.js:10 Korea (Democratic People's Republic of)

script.js:10 Northern Mariana Islands

script.js:10 Norway

script.js:10 Oman

script.js:10 Pakistan

script.js:10 Palau

script.js:10 Palestine, State of

script.js:10 Panama

script.js:10 Papua New Guinea

script.js:10 Paraguay

script.js:10 Peru

script.js:10 Philippines

script.js:10 Pitcairn

script.js:10 Poland

script.js:10 Portugal

script.js:10 Puerto Rico

script.js:10 Qatar

script.js:10 Republic of Kosovo

script.js:10 Réunion

script.js:10 Romania

script.js:10 Russian Federation

script.js:10 Rwanda

script.js:10 Saint Barthélemy

script.js:10 Saint Helena, Ascension and Tristan da Cunha

script.js:10 Saint Kitts and Nevis

script.js:10 Saint Lucia

script.js:10 Saint Martin (French part)

script.js:10 Saint Pierre and Miquelon

script.js:10 Saint Vincent and the Grenadines

script.js:10 Samoa

script.js:10 San Marino

script.js:10 Sao Tome and Principe

script.js:10 Saudi Arabia

script.js:10 Senegal

script.js:10 Serbia

script.js:10 Seychelles

script.js:10 Sierra Leone

script.js:10 Singapore

script.js:10 Sint Maarten (Dutch part)

script.js:10 Slovakia

script.js:10 Slovenia

script.js:10 Solomon Islands

script.js:10 Somalia

script.js:10 South Africa

script.js:10 South Georgia and the South Sandwich Islands

script.js:10 Korea (Republic of)

script.js:10 South Sudan

script.js:10 Spain

script.js:10 Sri Lanka

script.js:10 Sudan

script.js:10 Suriname

script.js:10 Svalbard and Jan Mayen

script.js:10 Swaziland

script.js:10 Sweden

script.js:10 Switzerland

script.js:10 Syrian Arab Republic

script.js:10 Taiwan

script.js:10 Tajikistan

script.js:10 Tanzania, United Republic of

script.js:10 Thailand

script.js:10 Timor-Leste

script.js:10 Togo

script.js:10 Tokelau

script.js:10 Tonga

script.js:10 Trinidad and Tobago

script.js:10 Tunisia

script.js:10 Turkey

script.js:10 Turkmenistan

script.js:10 Turks and Caicos Islands

script.js:10 Tuvalu

script.js:10 Uganda

script.js:10 Ukraine

script.js:10 United Arab Emirates

script.js:10 United Kingdom of Great Britain and Northern Ireland

script.js:10 United States of America

script.js:10 Uruguay

script.js:10 Uzbekistan

script.js:10 Vanuatu

script.js:10 Venezuela (Bolivarian Republic of)

script.js:10 Viet Nam

script.js:10 Wallis and Futuna

script.js:10 Western Sahara

script.js:10 Yemen

script.js:10 Zambia

script.js:10 Zimbabwe

# Copy by value

when a variable is assigned a value

i.e var x =17

var y =’xyz’

var z = null

‘x’ contains value 17, ‘y’ contains ‘xyz’.

the **values** ‘x’ and ‘y’ are copied into the variables ‘a’ and ‘b’.

Here both ‘x’ and ‘a’ contain the same value 17. Both ‘y’ and ‘b’ contain the same value ‘xyz’. However, an important thing to understand here is that even though ‘x’ and ‘a’ as well as ‘y’ and ‘b’ contains the same value they are not connected to each other. It is so because the values are directly copied into the new variables.Changes taking place in one does not affect the other.

# 

# Copy by reference

# The values it is assigned is not stored directly in the box. The language itself assigns a different memory location to store the data. The address of this memory location is stored in the box created.

Where,

Let user = { name : “guvi” }

Let admi = user;

admin.name= ‘geek’ // value changed

alert(user.name); // name changed to ‘geek’

when the value of admin is changed it automatically changes the value of user as well.This happens because both ‘user’ and ‘admin’ are storing the address of the memory location. And when one changes the values in the allocated memory it is reflected in the other as well.

We can further elaborate it we can say that; copy by reference is like having two keys of the same room shared between ‘admin’ and ‘user’.

**How do you copy by value a composite data type?**

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

# 1. Using Spread

**Spread operator** allows an iterable to expand in places where 0+ arguments are expected. It is mostly used in the variable array where there is more than 1 value are expected. It allows us the privilege to obtain a list of parameters from an array.Using spread will clone your object. Note this will be a shallow copy.

var c=[...a]

console.log(a,c)

[1,2,100] [1,2,100]

c[2]=80

console.log(a,c)

Output: [1,2,100] [1,2,80]

when copied variable value is changed but original variable value remain same .

# 2. Using Object.assign()

The **Object.assign()** method copies all enumerable own properties from one or more *source objects* to a *target object*. It returns the target object.Note this will be a shallow copy.

var a=[1,2,3]

var b=Object.assign([ ],a)

console.log(a,b)

[1,2,3] [1,2,3]

b[2]=100

console.log(a,b)

[1,2,3] [1,2,100]

The empty *[]* as the first argument, this will ensure you don't mutate the original object.

# 3.Using JSON.parse() and JSON.stringify()

The JSON object, available in all modern browsers, has two useful methods to deal with JSON-formatted content: parse and stringify. JSON.parse() takes a JSON string and transforms it into a JavaScript object. JSON.stringify() takes a JavaScript object and transforms it into a JSON string.Using JSON.parse() and JSON.stringify() for copy performs deep copy .

a=[1,2,3]

var b=JSON.parse(JSON.stringify(a))

console.log(a,b)

[1,2,3] [1,2,3]

b[2]=100

console.log(a,b)

[1,2,3] [1,2,100]

The deep copy is a true copy for nested objects. Shallow copy copies only reference in case of nested objects.

Problem 0:

**Html code:**

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Script.js 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(i in myCar.accidents){

myCar.accidents[i]. atFaultForAccident = false

}

console.log(myCar.accidents)

for(let i in myCar.accidents) {

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

}

Output :

(3) [{…}, {…}, {…}]0: {date: "3/15/2019", damage\_points: "5000", atFaultForAccident: false}1: {date: "7/4/2022", damage\_points: "2200", atFaultForAccident: false}2: {date: "6/22/2021", damage\_points: "7900", atFaultForAccident: false}length: 3\_\_proto\_\_: Array(0)

script.js:31 3/15/2019

script.js:31 7/4/2022

script.js:31 6/22/2021

Problrm 1:

Script.js code:

var object = {name : 'RajiniKanth', age : 33, hasPets : false};

function printAllValues(obj) {

console.log(Object.values(object))

}

printAllValues();

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

script.js:4 (3) ["RajiniKanth", 33, false]

Problem 2:

Script.js code

var object = {name : 'RajiniKanth', age : 33, hasPets : false};

function printAllKeys(obj) {

console.log(Object.keys(object))

}

printAllKeys();

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

["name", "age", "hasPets"]

Problem 3:

script .js code:

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

function convertListToObject(obj) {

console.log(Object.entries(object))

}

convertListToObject()

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

* *(3) [Array(2), Array(2), Array(2)]*
  + 0: (2) ["name", "ISRO"]
  + 1: (2) ["age", 35]
  + 2: (2) ["role", "Scientist"]
  + length: 3
  + \_\_proto\_\_: Array(0)

Problem 4:

Script.js code:

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

function transformFirstAndLast(arr){

let newObject={}

let arrlength=arr.length

newObject[arr[0]]=arr[arrlength-1]

return newObject;

}

console.log(transformFirstAndLast(arr))

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

{GUVI: "a geek"}

Problem 5:

Script.js code

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

function fromListToObject(arr) {

var newObject = {};

for(i in arr){

newObject[arr[i][0]]=arr[i][1]

}

return newObject;

}

console.log(fromListToObject(arr))

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

{make: "Ford", model: "Mustang", year: 1964}

Problem 6:

Script.js code

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

function transformEmployeeData(arr) {

var tranformEmployeeList=[]

let newObject={}

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

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

var key= arr[i][j][0]

newObject[key]=arr[i][j][1]

}

console.log(newObject);

} }

console.log(transformEmployeeData(arr))

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

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

Problem 7:

Script.js code

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

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

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

function assertObjectsEqual(actual, expected, testName) {

// your code here

//convert object to string.

var actualString = JSON.stringify(actual);

var expectedString = JSON.stringify(expected);

if(actualString != expectedString) {

console.log('FAILED [' + testName + '] Expected "' + expected + '", but got "' + actual + '"');

} else {

console.log("passed");

}

}

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

Passed

Problem 8:

Script.js 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,ans) {

let answerGiven = false;

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

if(securityQuestions[i].question === question){

if(securityQuestions[i].expectedAnswer=== ans){

answerGiven=true;

}

}

}

return answerGiven;

}

var ques= 'What was your first pet’s name?';

var ans ='FlufferNutter'

console.log(chksecurityQuestions(securityQuestions,ques,ans))

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

true

Problem 9:

Script.js 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)

{

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

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

console.log(students[i].name)

}

}

}

returnMinors(students)

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Output :

Maari

script.js:14 Bhallala Deva

script.js:14 Baahubali

script.js:14 Chulbul Pandey

Task 4:

Find the total population the countries.

Html code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>sample api data</title>

</head>

<body>

<script src = "script.js"> </script>

</body>

</html>

Script.js code

var request = new XMLHttpRequest();

request.open('GET','https://restcountries.eu/rest/v2/all',true);

request.send();

request.onload = function() {

var data = JSON.parse(this.response)

var sum =0

for(let i=0;i<data.length;i++)

{

sum=sum+data[i].population;

}

console.log("total population"+" "+sum);

}

Output :

total population 7349137231