ES6 Assignments

**Q1. Constants:** Declare a constant & confirm its value cannot be changed.

Sol : const testVariable= 10;

console.log(testVariable);

testVariable = 11;

// This will throw a Read only Error

**Q2. Scoping:** Declare a variable inside if condition & make sure that it is not accessible outside if condition.

Sol : let g=1;

let h="variable Outside If";

if(g==1)

{

let h="variable Inside If";

console.log(h); // output : variable Inside If

}

console.log(h); //output : variable Outside If

**Q3. Enhanced object properties:** Create an ‘Order’ object having data members ‘id’, ‘title’, ‘price’. Add the methods printOrder() & getPrice(). Now, copy the order object using Object.assign().

Sol : let order1 = {

id : 2,

title :"Order1",

price :100,

getprice()

{

return this.price;

},

printOrder(){

console.log("ID : "+this.id+" Order name: "+this.title+" Price: "+this.price);

}

}

order1.printOrder();

let order2=Object.assign(order1);

order2.printOrder();

**Q4. Arrow functions:** Take an array of strings & convert it into another array of object which has two properties {string, string\_length}.

Sol: let fruits = ["Orange","Banana","Apple","Grape"];

console.log(fruits);

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

let object = {name : arr[i], length : arr[i].length};

arr[i]=object;

}

};

operator(fruits);

console.log(fruits);

**Q5. Extended parameter handling:**

* 1. Write a add() with default values.
  2. Write a function userFriends() that takes 2 arguments username & array of user friends. The function should print username & his list of friends. (Use rest parameters)
  3. Write a function printCapitalNames() that takes five names as argument & prints them in capital letters. Use spread operator in order to call printCapitalNames() function.

Sol : //add function with default value

var add= function(a = 10,b = 10){

return a+b;

}

console.log(add());

//userFriends function with rest argument

var userFriends = function(user,...friends)

{

console.log("User : "+user);

console.log("User's friend : ");

for(let friend of friends)

{

console.log(friend);

}

}

userFriends("A","B","C","D");

//printCapitalNames function using spread operator

var printCapitalNames = function(first,second,third)

{

console.log("First : "+first.toUpperCase());

console.log("Second : "+second.toUpperCase());

console.log("Third : "+third.toUpperCase());

}

let arg = ["ram","shyam","mohan"];

printCapitalNames(...arg);

**Q6. Template literals:** Draft a ticket to Sysnet that describes problem with your laptop. Use ‘template literals’ to add value of laptop model, your desk no, your name etc.

Sol : var problem = `Laptop is crashing everytime i open jarik application

Details :

Laptop model : Acer Aspire 5

Desk number : j5

Name : Shashank`;

console.log(problem);

**Q7. De-structuring assignment:**

* 1. Suppose there is a javascript array with 4 elements. Print the value of 3rd element using array matching.
  2. Create an organization object having attributes name, address. Write a program to retrieve pin code of an address using object deep matching.

Sol : let arr=[1,2,3,4];

let [,,third,]=arr;

console.log(third);

let organisation = {

name : 'Raghav',

address : {

city : 'A',

state : 'S',

pincode : 110110

}

}

console.log(organisation);

let {name,address} = organisation;

let {city,state,pincode} =address;

console.log(pincode);

**Q8. Classes & Modules:** Write a class Account with attributes id, name, balance. Add two sub classes SavingAccount & CurrentAccount having specific attribute interest & cash\_credit respectively. Create multiple saving & current account objects. Write a functionality to find out total balance in the bank

Sol : //Declaring a parent class to store general information

class Account{

constructor(name,age,balance){

this.name=name;

this.age=age;

this.balance=balance;

}

getBalance(){

return this.balance;

}

}

//Making a Saving class inheriting Account class and having a extra interest field

class Saving extends Account{

constructor(name,age,balance,interest)

{

super(name,age,balance);

this.interest=interest;

}

}

//Making a Current class inheriting Account class and having a extra field cash\_credit

class Current extends Account{

constructor(name,age,balance,cash\_credit)

{

super(name,age,balance);

this.cash\_credit=cash\_credit;

}

}

//Defining two array to store object of saving and current class

let currentAccount=[];

let savingAccount=[];

//Instantiating different object of Saving class and pushing it in savingAccount array

savingAccount.push(new Saving("A",21,100,5));

savingAccount.push(new Saving("B",20,200,5));

savingAccount.push(new Saving("C",15,300,5));

savingAccount.push(new Saving("D",29,400,5));

console.log(savingAccount);

//Instantiating different object of Current class and pushing it in currentAccount array

currentAccount.push(new Current("A",30,5000,100));

currentAccount.push(new Current("B",35,5020,100));

currentAccount.push(new Current("c",36,50,100));

currentAccount.push(new Current("s",39,5500,100));

currentAccount.push(new Current("g",38,6000,100));

console.log(currentAccount)

/\*Defining function to calculate total balance in bank.

It takes saving account and current account object array as input

\*/

var balanceInBank = function(savingArr,currentArr)

{

let total=0;

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

total=total+savingArr[i].getBalance();

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

total=total+currentArr[i].getBalance();

console.log(total);

}

balanceInBank(savingAccount,currentAccount);