**OOJS Assignment**

Q1. Create a hierarchy of person, employee and developers.

Ans.

Code:

function person(name, age){

this.name=name;

this.age=age;

}

function employee(empid, salary) {

this.empid=empid;

this.salary=salary;

}

function department(empnum, depname){

this.depname=depname;

}

employee.prototype = new person('zafeer' , 22);

department.prototype = new employee(3260,30000);

var dept=new department('FEEN');

console.log(dept);

….



Q2. Given an array, say [1,2,3,4,5]. Print each element of an array after 3 secs.

Ans.

var say1=[1,2,3,4,5];

var j=0;

function intar(){

if(j<say1.length)

{

console.log(say1[j]);

j++;

}

}

setInterval(intar,3000);

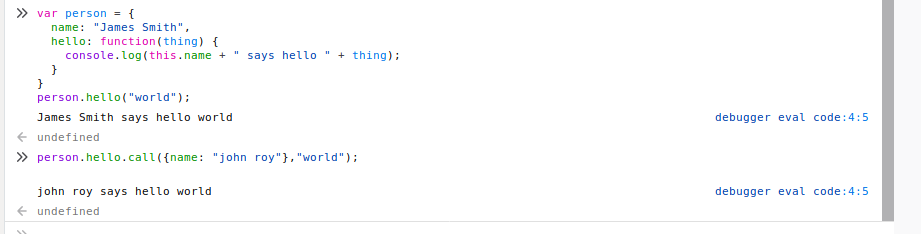
Q3. Explain difference between Bind and Call (example).

Ans.

call(): It is an alternate means to invoke the function.

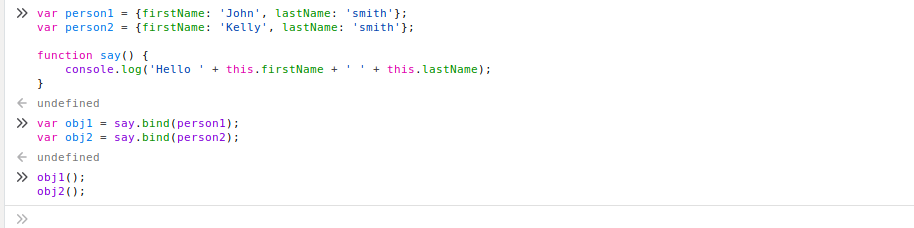
Call() accepts an array of parameters and a parameter itself.

Example of call(): -



Bind(): The bind() method creates a new function where “this” refers to the parameter in the parenthesis . This way the bind() method enables calling a function with a specified “this” value.

Example of passing argument with bind() : -



Q4. Explain 3 properties of argument object.

Ans.

* arguments is an Array-like object accessible inside functions that contains the values of the arguments passed to that function.
* You can refer to a function's arguments inside that function by using its arguments object
* Argument caller : Reference to the currently executing function that the arguments belong to.

Q5. Create a function which returns number of invocations and number of instances of a function.

Ans.

Code:

var invo=0; var inst=0;

function myFun(){

invo++;

alert("hi , i am invocated " +invo);

}

function check(c){

if(c==1)

{

var i=new myFun();

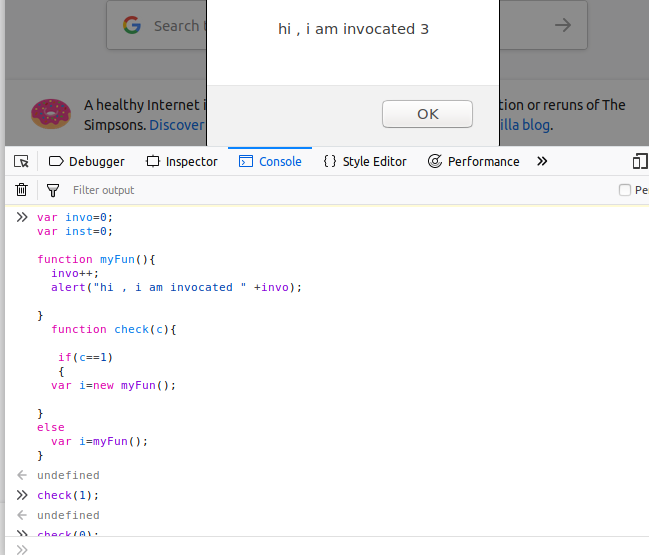
}

else

var i=myFun();

}

Output:



Q6. Create a counter using closures.

Closures are functions that refer to independent variables (variables that are used locally, but defined in an enclosing scope). In other words, these functions 'remember' the environment in which they were created.

Code:

function outer(){

var count=0;

function middle(){

count=++count;

function inner(){

console.log("total closure" + ++count); }

inner(); }

middle(); }

outer();

