Q1. What is JSON and difference between JSON.stringify & JSON.parse ?

Json stands for javascript object notation.

Json is used to transfer data from one application to another.

Json is pureply data which hold json object.

Json start with {} , key should always string and in “ “

Json cant have method.

**Javascript object:**

Vat r = {

Name: “ROHEAT”,

Age : 29,

}

**Json object:**

{

“name” : “ROHEAT”

“age” : “29”

}

JavaScript engine work inside and json work outside.

In JavaScript, for every last line separated with (,) comma and In json we wont use comma(,) , if comma used it will gives error.

**It has a two predefined methods json.stringify and json.parse**

* Json.stringify method JavaScript object used to convert JavaScript object to json object.

Var js = {

Name: “rohit”

City : “nagpur”

}

Const exe = json.stringify(js);

* Json.parse method is used to convert json object to javascript object

Var js = {

Name: “rohit”

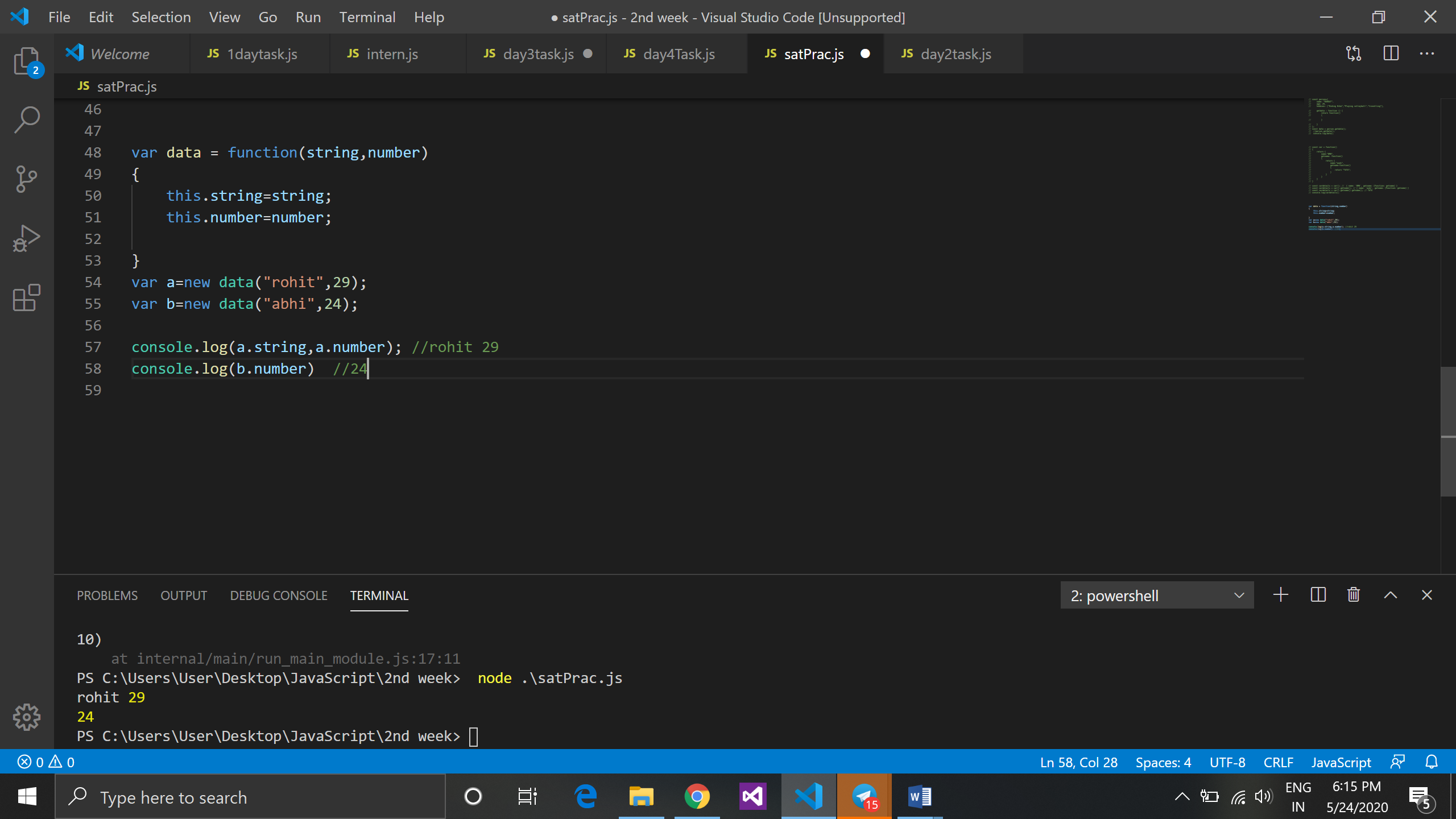
City : “nagpur”

}

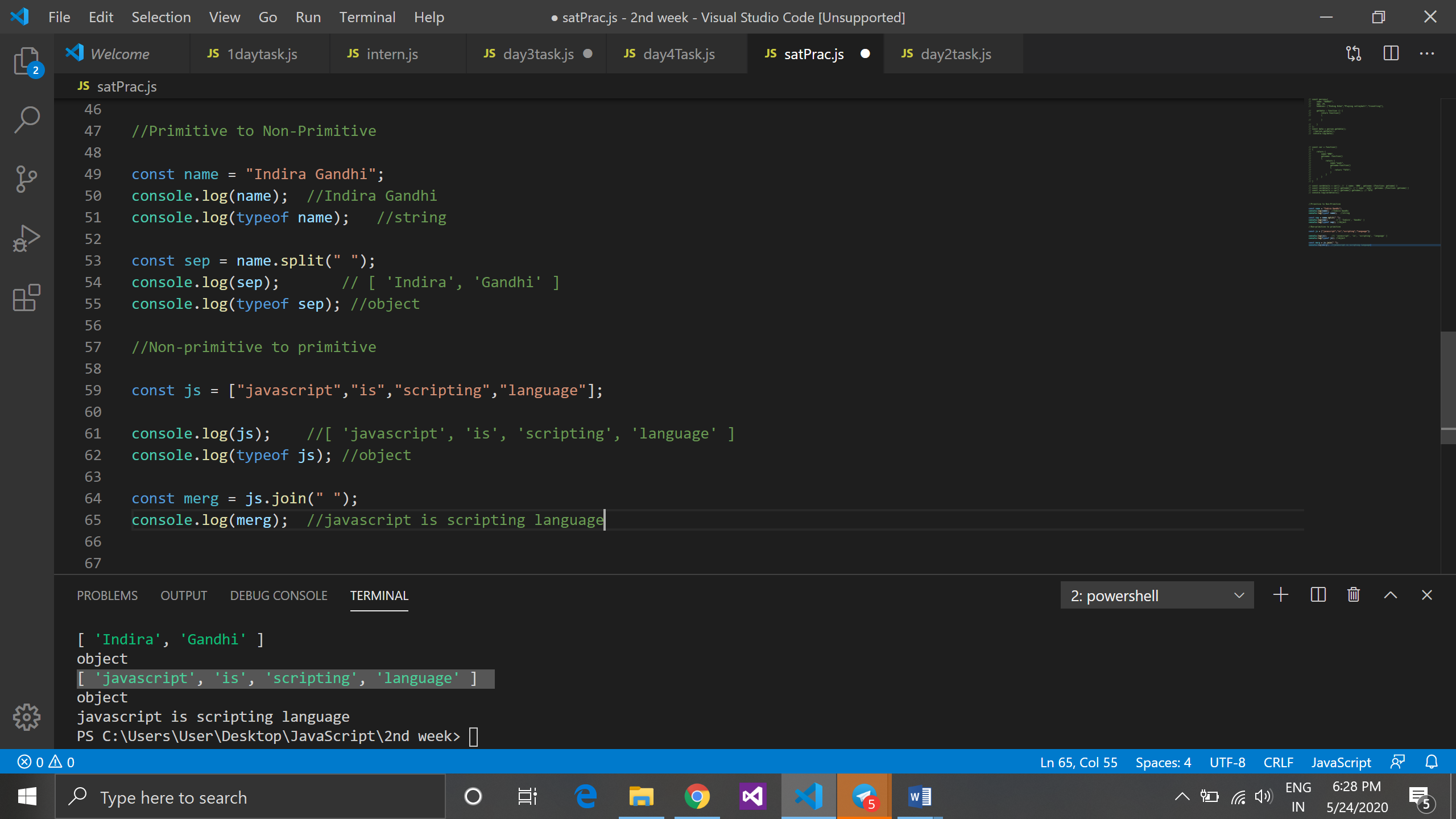
Const exe = json.parse(js);

Q2. What is constructor function?

The function statement is not the only way to define a new function; you can define your function dynamically using **Function()** constructor along with the **new** keyword.



Q3. How primitive value can be converted into non-primitive and vice-versa?



Q4. Deconstruct in plain English and write pseudo code.

(i) const result = total(1)(2)(3);

console.log(result); //1+2+3 = 6 [Expected Output is 6];

=> const total = function(param)

{

return function(para)

{

return function(par)

{

return param+para+par;

}

}

}

const result = total(1)(2)(3);

console.log(result);

-----------------------------------------

(ii) fnOne(1).fnTwo(2).then(function(result) { console.log(result)});

// Expected Output is 1+2=3;

const fnone = function(a)

{

return fntwo: function(b)

{

return {

then: function(result)

{

var output= a+b;

result(output);

}

}

}

}

-----------------------------------------

(iii) Schema.model.add(40,60).then(function(result) { console.log(result) }).catch(function(err) { console.log(err) });

// pass any two numbers in "add method" and addition of two numbers is 100, if addition is not 100 it should print "wrong arguments" in "catch method"

const schema = {

model:

{

add:function (x,y)

{

var out = x+y;

return : function(res)

{

return

{

catch:function(err)

{

if(out == 100)

{

res(out);

}

else {

console.log("plz enter properly");

}

}

}

}

}

}

}