1 trong hai số không phải là số, thì trả về chuỗi

2. boolen

a. console.log ra thì ra true có in màu xanh

b. in biến string thì nó ra màu đen

3. biến trả về false

a. **0, Null**

**false**

**''**

**Undefined**

**NaN**

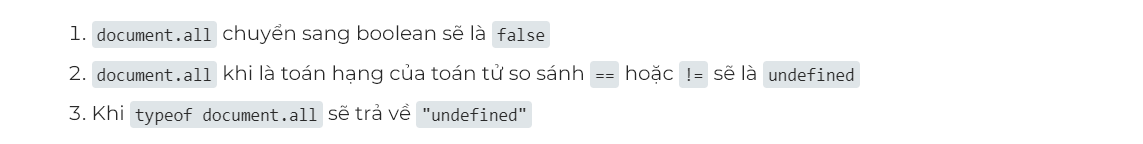
**b.** các kiẻu mà trả về fasle 🡪 gọi là fasly

4, biến trả về true

1. name : ‘miu’;

console.log(name)🡪 true;

1. các giá trị trả về true gọi là truthy

5. 

6. toán tử logic && ||

a. && tìm cho đến khi nào gặp fasly thì dừng lại

b. || tìm cho đén khi nào gặp truthy thì dừng lại

7. private : riêng

a. biến cục bộ trong hàm chỉ được sử dụng trong hàm

8.Argument

a. hạot động đưa các biến truyền vào , vào trong hàm và đặc trưng bởi từng index

b.for of

for( var temp of argument){

câu lệnh;

}

c.Chức năng : truyền tham số ở input nhưng không cần khai báo ở trong () của hàm

9. parameter(tham số)

10.arlet, comfirm

a.khi chạy đến dòng mã nãy , khi nào ng dùng thực thi kích hoạt từ web thì các đoạn mã ở sau nó mới được thực thi

1 1. toString

1. a = 1, b =2; a.toString() + b.toString() = 12

12. khai báo tên hàm giống nhau

a. cái hàm sau sẽ ghi đè cái hàm trước

13. khai báo hàm trong hàm

a. có thể , và hàm con đó cũng sẽ mang tính private

14.function:

-declaration ; khai báo

-Express: bày tỏ

a. declaration:

a1.

-- được hỗ trợ hosting

showMessage();(gọi hàm trước cả khai báo hàm)

Function showMessage(){

Console.log(“hello”);

}

🡪output: hello;

b.express function:

🡪 không hỗ trợ hosting

B1.Var showMessage = function(){

Console.log(“hello”);

}

B2. Var object = {

objectFunction =function{

console.log(“hello”);

}

Hoặc là truyền vào function một function

Vd. Function setTimeOut(function run(){});

}

15.string

a. khởi tạo

a1.var s = “chuỗi”

var s = new string(“chuỗi”)

* cách trên được ưu tiên hơn và trả về type là string
* cách dưới trả về object

16. Hiện thị ‘, \ trong string

a. thêm\ vào trước \\ -> \, \’-> ‘

# 17. Template String

# a = “lo”

# `he ${a}` = “he” + ‘ ‘ a == he lo

# 18. length

# a.length

# 19,find index

# a. indexOf

# s.indexOf (“string”)

# s.indexOf (“string”, vị trí bắt đầu tiềm kiếm)

# lastIndexof, truyền vào chuồi , tìm chuỗi đó vị trí cuối cùng nhất trong mảng

# search, I như indexOf, nhưng nó là tìm chính quy

# 20. slice : cắt

# a. s.slice(3) 🡪 cắt từ vị trí 3 cho đến cuối

# b.s.slice(3,5)🡪 cắt từ vị trí 3 đến 5

# trái phải đều bắt đầu từ 0 , canh mà cắt

# “hello” ,slice(2) == slice(-3)

# Slice số âm không nhận tham số 0 , nếu muốn nó cắt hết theo hướng gốc âm thì truyền một tham số như trên thôi

# 21.replace

# a. s.replace(‘js’, ‘javascript’);

# thay thế js đầu tiên thành javascript

# replace all: (dùng biểu thức chính quy)

# s.replace( /js/g,”javascript”)

# 22. upper case: chữ hoa

# Lower case : chữ thường

# s.toUpperCase() , s.toLowerCase()

# 23. TRIM : Loại bỏ khoảng trắng ở hai đầu:

# a. s.trim()

# 24.split: tách

# a. tách all:

# s.plit(‘’) –> truyền dấu hiệu chung là chuỗi rỗng

# b.tách theo điểm phân cách chung: ( nó tạo ra một array)

# “he lo”🡪s.split(‘, ’) 🡪 [‘he’,’lo’]

# 25. lấy kí tự

# “hello”

# A1. charAt 🡪 s.charAt(2) 🡪lấy kí tự số 1

# B1.s.charAt(20)🡪 không có 🡪 trả về string (chuỗi rỗng)

# A2. S[2]

# B2. S[20] 🡪 trả về undefined

# 26. Đối tượng number

# 1.Number.isFinite() 🡪 Ktra có hữu hạn không

# 2.Number.isInteger()

# 3.Number.parseFloat()

# 4.Number.parseInt()

# Parse: phân tích

# 27. isNaN

# Kiểm tra xem biểu thức , số đó có hợp lệ kô vd: 2/’abd’ = NaN-=> true

# 

# 5. thuộc tính chuyển thành string

# toFixed() (làm tròn)

# toString()

# 28.trong một Array có thể có tất cả kiểu dữ liệu đã được học kể cả

# A.Null, undefine, function(){},{} 🡪 một object vẫn được

# b. phân cách nhau bằng dấu phẩy

# 2 9. Kiểm tra array

# Array.isArray(truyền trực tiếp mảng vào hoặc tạo thẳng Array trong đây cũng được);

# Ex; Array.isArray(var) ==

# Độ dài mảng : .length;

# Lấy phần tử của mạng giống c++

# 30. Array method

# a. s.toString()🡪 chuyển một mảng thành một chuỗi, các thành phần mảng cũ được ngăn cách nhau bằng dấu phẩy

# a1. S.join(“kí tự ngăng cách”) 🡪 chức năng y toString nhưng mà nó ngăn cách theo thứ tự do mình đặt

# 31. s.pop ()

# a. xoá phần tử cuối mảng trả về phần tử vừa xoá

# s.shift()

# a. xoá một phần tử ở đầu mảng ,return phần tử vừa xoá

# 32.

# s.push( ‘hehe’, ‘hjhj’)

# Add hai phần tử ở cuối mảng 🡪 Return số lượng mới của mảng

# s.Unshift(‘hehe’,’hihi’)

# Add hai phần tử vào đầu mảng 🡪 Return số lượng phần tử mới của mảng

# 33. s.splice(index bắt đầu , số lượng muốn xoá từ hiện tại trở về sau, các kí tự muốn chèn vào...v.v chỗ này có thể có hoặc không )

# a. ví dụ:

# var s = [1,2,3,4]

# s.splice( 2,1, 33) 🡪 từ vị trí 2 là số 3, nó xoá một phần tử là 3, rồi chèn 33 vào 🡪 s = [1,2,33,4]

# return phần tử đã xoá

# 34. s1.concat(s2)

# a. nối s2 vào đuôi của s1

# return s1

# 35. s.slice 🡪 cắt (không tác động thực lên mảng)

# A0.Lấy xuôi

# s.slice(1,4) cắt từ nữa đoạn index thuộc [1,4) trả về phần tử đc cắt

# a1. Lấy ngược

# 

# b. s.slice(1) cắt từ 1 đến cuối mảng

# c. s.slice (0) 🡪 coppy mảng

# 36. đặt tên trong object:

# a. ten dạng vi phạm như : full-name : ‘dang cong tuan’

# 🡪 solve : đặt full-name trong cặp ngoặc đơn

# ‘full-name : ‘dang cong tuan’;

# 37. add biến ngoài vùng object :

# a.myInfor.email = ‘ta@gmail.com’

# b. nếu tên vi phạm ,hoặc

# 🡪 myInfor[‘my-email’] = ‘ta@gmail.com’

# 38. nếu duyệt một biến không có trong object 🡪 return về undefined

# 39. truyền tên biến gián tiếp từ vùng bên ngoài:

# 

# 40. truy xuất gián tiếp:

# a.

# 

# In ra birth-day

# 41. truyền function vào object:

# a.

# 

# 🡪 chú ý: cái function getName 🡪 là một thuộc tính

# cái name, ‘birth-day’ ..v.v gọi là phương thức

# 42.Con trỏ this nằm trong vùng hoạt động của object myInfor

# a.đóng vai trò là myInfor

# b.sau này có đỗi tên của object thì khỏi mấy công đỗi tên nếu trỏ theo kiểu truyền thống như trên là myInfor.name;

# 43 user : tác giả