

JavaScript Assertions Guide

Interview → Confidence लोना चाहिए
→ मुझे जो दोनों में उत्तरा मुझे प्रश्नों पर उत्तरा।
परों का 8 बैटे पर में उत्तरा।
, में उत्तर दुँगा क्यों की

06 - array.js

Canot Carriage I = ["Voor", "Agyah", "Rovi"]

cout <> emptyCarriage = [];

Count - three empty arrays = Array (3);

Geology (fossils, flint types etc.)

6

[<3 empty items>]
Length : 3

Cast Passanger = Array ("Veer", "Aayush", "Rovi")

Cost Dir y passenger = Array. of (3)

Carabidae. 109 (Singletons & others)

6 [37]

Array of (3)

٦

七

1

Arg(3)

$\{ \}$ (empty set)

dearer than

Array, from

Const ~~for~~ arr~~able~~ = Array. from ("DUST")

Const~~able~~ (arr~~able~~):

Const ~~for~~ arr~~able~~ = Array. for ("DUST")

Const temparr = ["A", "B", "C", "D", "E"]

→ temparr.length = ?

Const~~able~~ (temparr):

temparr.length = ?

Const~~able~~ (temparr):

// push, pop, shift, unshift, splice

// concat, slice, ~~flat~~

↳ not mutate original Array

Const toinGpy = ^{Array} wholepart · slice()

↳ Grammar tactics to Gpy an Array

// searching: index, indexOf, includes - first, findIndex,

Const~~able~~ (typ_of [])

Const~~able~~ (Array. isArray ([]))

Const~~able~~ (Array. isArray ("Rov. "))

Key Points

1. C++ ↗ ~~but~~ Array (4)
↳ qf rule
entre & of
↳ fib fib fib
element exist
fib fib exist
return of

2. Array are 0 based

↳ 0वां शुल्क

↳ means ~~1~~ element के ना
index 0 के 6 + 1
ना ~~1~~ की value undefined

3. mutating methods: push, pop,

diff, shift, splice

4. Non-mutating: concat,

filter, filter [1, 2, 3, 4, 5]

5. Splicing includes

6. Array.isArray()

Obj. Array methods.js

Cont. orders = [

- obj: "Pasta Carbonara", price: 14, spicy: false, qty: 2;
- obj: "Droge Ratte", price: 12, spicy: true,
- obj: "Cesar Salat", price: 9, spicy: false, qty: 3;
- obj: "Inferno Wings", price: 11, spicy: true, qty: 2

];

Cont myData = orders.forEach((order, index) => {

console.log(`#\${index + 1}: \${order.qty}x
\${order.dish}`);

});

console.log(myData);

↳ undefined

↳ forEach nothing return

Cont receipts = orders.map((o) => `\${o.dish}:

`\${o.price * o.qty};`

console.log(receipts)

Cont spicyOrders = orders.filter(o => o.spicy)

↳ filter is giving condition

NOTE :-

console.log(spicyOrders)

Reduce

जो क्या Reduce करा पाएगा

decide करा है।

Cost total Revenue = order.reduce(
(sum, order) => {

return sum + order.quantity
* order.Price

}, 0);

CostWithTotalRevenue

Single Line Gorunt
return arr[0]

need -> (1 =>)

arr [1 => ()]

return keyword ->

need ->

(1 => f3) | 8

return keyword
8 need से है

Cost grouped = orders.reduce ((c => c), initialValue)

पर तो क्या ही हो

किसी है नहीं

initialValue 0 है

लेणा mean है

विभिन्न datatype है

हो जाएगा है

Reduce Right
जो cost के element को arr से करा है।

Ques 1 ticket Number 123 [100, 25, 3, 42, 8] Ticket

Ans. Sorted Inv = [... ticketNumber J. sort ()]

Grade.100 (Sorted Inv)

[100, 25, 3, 42, 8]

ये क्या Sort होता था शायद

Sort क्या है। |

हो से Really मे Sort करना है।

Sort के Global क्या

इसका (toSorted) पर होता है जिसका
(a, b) के Subcase कि देना पड़ता
है by default होता है।

except & Synchronous function

इसमें break, continue का नहीं

करता है

इसको लिये मे Stop करता है।

किसी गा करता है।

only, await इसमें का गवाता है।

V.V.I

Note :- Function

इसके बारे में Continue का नहीं

करता है

इसको लिये मे Stop करता है।

किसी गा करता है।

only, await इसमें का गवाता है।

Cust Kitchen Orders = [

{ dish : "pasta Carbonara", Price: 14, Spicy: false,
Qty: 2},

{ dish : "Dragon Roll", Price: 2, Spicy: true, Qty: 13},

{ dish : "Caesar Salad", Price: 9, Spicy: false, Qty: 3},

{ dish : "Inferno wings", Price: 11, Spicy: true,

Qty: 2}

]

Cust Mild Report = KitchenOrderFilter (order =>
! order.Spicy), Map (order => {

dish: order.Dish,

total: order.Price * order.Qty,
})

Note: use ToSorted() instead of Sort()

QS-Object ->

Cust Hero = &

name: "Luna the brave",

Class: "Mage",

Level: 12,

Health: 85

Mana: 120,

IsAlive: true

hero.weapon = "Pise"

delete hero.level;

Got danger = {

name : " ",

agility : 80,

stealth : undefined

}

check("name" in danger); true

check("agility" in danger); true

check("stealth" in danger); true

check("hostile" in danger); true

but testing न है

नहीं पर testing

inner property object

जो क्षमा है मानव

true आए

इनमें hasOwnProperty असाक रख

अपनी property को inherit

property को होता

Constructors (C)

obj - Object methods (J)

```
const artifact = {  
    name: "Obsidian Crown",  
    era: "Ancient",  
    value: 80000,  
    material: "Volcanic glass"  
};
```

Get keys = Object.keys (artifact)

Get value = Object.values (artifact)

Get entries = Object.entries (artifact)

Object.keys (Keys)

↳ ['name', 'era', 'value', 'material']

Object.values (Value)

Object.entries (Entries);

~~for (Object entries (artifact))~~

for (Object [key, value] of object.entries (artifact))
Grafolo.Log (`\${key} : \${value}`)
}

q.24

pm
video

Graf Object = [

[{"Obidian Rock", 5000},
 {"Ruby Pendant", 30000},
 {"Iron Shield", 5000},

] :

Graf PriceObject = Object. fromEntries (Object)

Graf displayCase = {
 artifact: "Obidian",
 location: "Hall A, Corridor",
 locked: true
 }

Object.freeze (displayCase);

Delete displayCase. locked;

displayCase. updated = "test";

Grafolo (displayCase);

Grat.GatalogEntry = {

id : "ART-001",

Description : "Ancient Crystal".

Verifiable : true.

}

Object - Grat (GatalogEntry)

{
 → ID odd entry → ✓
 } Edit entry → ✓

CatalogEntry.id = "ART-002"

↳ Allow & Edit

Object CatalogEntry.id ;

↳ Not Allow

Grat SecurArtifacts = { name : "Ruby Pendant" }

N.V.T
Object.defineProperty (SecurArtifacts, "GatalogID", {
 value : "SEC-999",

 writable : false,

 enumerable : true,

 ↳ true मात्रा गोपनीयता
 पर्याप्त नहीं

 Configurable : false,

})

 ↳ delete या edit -/✓
 ेट्ट.

Controlling (SecureArtifacts CatalogId):

↳ See - 999

SecureArtifacts. CatalogId = "Hacker";

Gathering (SecureArtifacts. CatalogId);

~~for (Object entries (SecureArtifacts)~~

~~for (Grou [key, value] of Object.entries (Secure-
Artifacts)) {~~

Gathering ({key : value});

}

Get Desc. = Object.getOwnPropertyDescriptor (SecureArtifact, "CatalogId");

Gathering (desc);

↳ name : Ruby Perlone

{ value : "Ruby Perlone" }

II Loop key points

II

II 1. for()

II 2. while

II 3. do while

II 4. for...in → objects पर ✓

→ Array पर avoid से

II 5. for...of → Aggregate

→ String पर

II 6. map, forEach, filter, every, reduce

Customized Loop

Note: for each return नहीं करा दे पाए

for each return करेंगे तो भी

नहीं कुछ नहीं मिलेगा।।।

undefined ही मिलेगा

② accumulator decide which datatype

return by reduce

O-function

function brewPotion(ingredient, dose) {

return "Brewing potion with \${ingredient} (\$s of
dose). - potion ready!"

}

get mixElixir = function(ingredient) {

return "Mixing elixir with \${ingredient}"

}

// no own 'this', no 'arguments' object

get distillElixir = (ingredient) => {

return "Mixing elixir with \${ingredient}"

}

function oldBrewingEgs() {

ChokeLog("Type:", typeOfArguments)
ControlLog(arguments)

3

oldBrewingEgs("Sage", "Rosemary");

Type: Object

[Arguments] { '0': 'Sage', '1': 'Rosemary' }

Arrow function & arguments Object का बारे में

Concise arrow function = () {
 ↳ Global (arguments)

}

arrow function ()
 ↳ agar program को ही GTE
 नहीं फैला सकता कि विल
 + क्या GTE का दौरा

Global arrow function = () {
 → क्या है

 ↳ Global (arguments)

} GTE (e)

 ↳ Global (e)

}

}

arrow function ()

{ let globalCount = 0;

 function brewAndCount (name) {
 globalCount += 1

}

→ impure function

Pure function
पृष्ठीय अस्ति नहीं
+ variable change
तो,
impure function
अस्ति अस्ति की
variable change वाली

IIFE

)();

Get positionShop = (function () { })()

(function () {})()

- एक function में string पास हो लगा;
- एक function में Number पास हो लगा;
- एक function string या number या और भी को भी लगा;
- सतत वर्तने function, function on a Parameter

जो लगा है उसे function

return नहीं कर पाया है,

() की HOF कल्पाना है

↳ Higher order

function

या नई argument function के वाले
return की function

Note normal function are arrow function

मेरा था तुम cliff. की होगा

जब this keyword से arguments,

तुम आपना तुम cliff. की होगा,

Ques potionShop = (function () {})

Ques potionShop = (function () {

let inventory = 0;

return {

brew () {

inventory += 1;

return "Brew Potion # \$ { inventory }";

}

getStock () {

return inventory;

}

}

}();

Ques log (potionShop);

Ques log (potionShop . brew ());

Ques log (potionShop . inventory);

function makeFunc () {

const name = "mozilla";

function displayName () {

console.log (name);

}

return displayName;

}

const myfunc = makeFunc();

myfunc();

जब function GII रहा

display हो जाएगा त

after the GII

जब ~~function~~ खत्या

function return हो रहा

वही displayName

उपरी displayName

Name variable

Access कर पाएंगे

उपरी करना नहीं पाएंगे

अब makefunc का GII हो चुका है तब

name clear हो जाएगा

एवं उपरी accessible हो जाएगा

displayName GII नहीं होता।

ये name उपरी
wipe out हो देता

फ्रॉन्ट closure

का Concept आता है
मतलब function return
होता है।