Array Method fill()

มี 3 Syntax ดังนี้

1. fill(value) นำค่าที่ป้อนเข้าไปแทนใน อาเรย์ทั้งหมด

วิธีใช้ ตัวแปรอาเรย์ตามด้วย .fill(value); โดยจะรับ พารามิเตอร์เพียง 1 ค่าแล้วนำไปแทนในอาเรย์ทั้งหมด

fill(value) value คือค่าที่จำเป็นต้องใส่เพื่อใช้งาน Syntax สามารถรับข้อมูลได้ทุกชนิด

ตัวอย่าง จะเป็นการใช้งาน arrow function โดยจะรับค่าข้อมูลที่ต่างกัน 3 ชนิด

```
const num = [1,2,3,4];
let x = { num : 1};
let z = [ '10','20'];

const syn = (s) => { num.fill(s); return num;};

console.log(syn(5)); //รับพารามิเตอร์เป็น number
console.log(syn(x)); //รับพารามิเตอร์เป็น object
console.log(syn(z)); //รับพารามิเตอร์เป็น array
```

Output ทั้ง 3 ตัวอย่าง 1.รับข้อมูลชนิดตัวเลข 2.รับข้อมูลชนิดobj 3.รับข้อมูลชนิดarray

โดยข้อมูลที่รับมาจะเข้าไปแทนที่ข้อมูลเดิมในอาเรย์ทั้งหมด

```
D:\Sinkung\INT201\JS-SourceCodes-17082021-master (1)>node 201
[ 5, 5, 5, 5 ]
[ { num: 1 }, { num: 1 }, { num: 1 }, { num: 1 } ]
[ [ '10', '20' ], [ '10', '20' ], [ '10', '20' ], [ '10', '20' ] ]
```

2. fill(value, start)

ความสามารถคล้าย fill(value) แต่จะมีพารามิเตอร์เพิ่มขึ้นมา 1 ตัวเป็นตัวกำหนดelementที่จะแสดง

วิธีใช้ จะมี 2 พารามิเตอร์ 1. Value จะรับข้อมูลทุกชนิดเพื่อไปแทนelementของอาเรย์ทุกช่อง

2. start รับข้อมูลเป็น number ทำหน้าที่ในการแสดงจำนวนelementที่ป้อนเข้าไป

เช่น fill(value,2) ข้อมูลที่จะแสดงคือ [ele1,ele2,value,...]

การใช้งาน ตัวแปรอาเรย์ตามด้วย .fill(value, start); โดยจะรับ พารามิเตอร์ 2 ค่า value รับข้อมูลทุกชนิด

Start รับข้อมูลเป็น number ค่าสูงสุดไม่เกินจำนวน length ของอาเรย์

ตัวอย่าง จะเป็นการใช้งาน arrow function โดยจะรับค่าข้อมูลที่ต่างกัน 3 ชนิด

```
const num = [1,2,3,4];
let x = { num : 1};
let z = [ '10','20'];

const syn = (s,m) => { let o = num.fill(s,m); return o;};

console.log(syn(5,2)); //รับพารามิเตอร์เป็น number
console.log(syn(x,2)); //รับพารามิเตอร์เป็น object
console.log(syn(z,2)); //รับพารามิเตอร์เป็น array
```

Output ทั้ง 3 ตัวอย่าง 1.รับข้อมูลชนิดตัวเลข 2.รับข้อมูลชนิดobj 3.รับข้อมูลชนิดarray

โดยจะแสดง element 2 ช่องแรกก่อน แล้วจะแทนด้วยข้อมูลที่รับมาในelement ที่เหลือ

```
D:\Sinkung\INT201\JS-SourceCodes-17082021-master (1)>node 201
[ 1, 2, 5, 5 ]
[ 1, 2, { num: 1 }, { num: 1 } ]
[ 1, 2, [ '10', '20' ], [ '10', '20' ] ]
```

3. fill(value, start, end)

ความสามารถคล้าย fill(value, start) แต่จะมีพารามิเตอร์เพิ่มขึ้นมา 1 ตัวเป็นตัวกำหนดelementสิ้นสุดที่จะแทนค่าเข้าไป

วิธีใช้ จะมี 2 พารามิเตอร์ 1. Value จะรับข้อมูลทุกชนิดเพื่อไปแทนelementของอาเรย์ทุกช่อง

2. start รับข้อมูลเป็น number ทำหน้าที่ในการแสดงจำนวนelementที่ป้อนเข้าไป

3. end รับข้อมูลเป็น number

ทำหน้าที่ในกำหนดจำนวนelementที่จะเติมค่าเข้าไป

เช่น fill(value,0,3) ข้อมูลที่จะแสดงคือ [value,value,value,ele4,ele...]

การใช้งาน ตัวแปรอาเรย์ตามด้วย .fill(value, start, end); โดยจะรับ พารามิเตอร์ 3 ค่า value รับข้อมูลทุกชนิด

Start รับข้อมูลเป็น number ค่าสูงสุดไม่เกินจำนวน length ของอาเรย์

end รับข้อมูลเป็น number ค่าสูงสุดไม่เกินจำนวน length ของอาเรย์

ตัวอย่าง จะเป็นการใช้งาน arrow function โดยจะรับค่าข้อมูลที่ต่างกัน 3 ชนิด

```
const num = [1,2,3,4];
let x = { num : 1};
let z = [ '10','20'];

const syn = (v,s,e) => { let o = num.fill(v,s,e); return o;};

console.log(syn(5,2,3)); //รับพารามิเตอร์เป็น number
console.log(syn(x,2,3)); //รับพารามิเตอร์เป็น object
console.log(syn(z,2,3)); //รับพารามิเตอร์เป็น array
```

Output ทั้ง 3 ตัวอย่าง 1.รับข้อมูลชนิดตัวเลข 2.รับข้อมูลชนิดobj 3.รับข้อมูลชนิดarray

e จะกำหนดให้มีการเติมค่า v จนถึง element ที่ 3

แต่ element 1 และ 2 ถูกค่า s กำหนดให้แสดงข้อมูลเดิม

ดังนั้น ค่า v จึงถูกเติมใน element ที่ 3 เพียงช่องเดียว

```
D:\Sinkung\INT201\JS-SourceCodes-17082021-master (1)>node 201
[ 1, 2, 5, 4 ]
[ 1, 2, { num: 1 }, 4 ]
[ 1, 2, [ '10', '20' ], 4 ]
```

Array Method find()

มี 3 Syntax ดังนี้

1. find((element) => {...})

วิธีใช้ ตัวแปรอาเรย์ตามด้วย .find(element); โดยจะ return ค่าที่ตรงกับองค์ประกอบแรกจาก array ของฟังก์ชันที่เรากำหนดไว้ใน find

find(element) element คือ ฟังก์ชันที่เรากำหนดขึ้นมา จะเป็นรูปแบบ arrow function ก็ได้

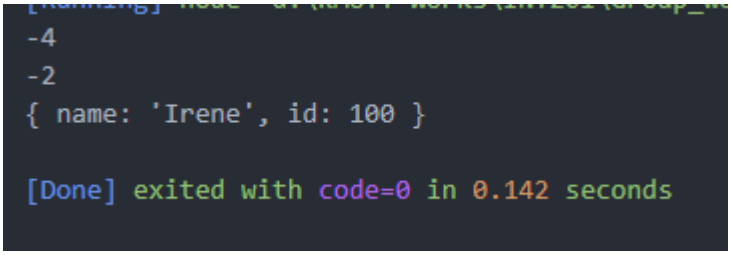
ตัวอย่าง จะเป็นการใช้งาน arrow function

```
let numbers = [-4,4,8,10,25,30];
let values = [-2,2,4,6,8,10,12,19];
const student = [
    {name: "Irene", id: 100},
    {name: "Kai", id: 101},
    {name: "Wendy", id: 102},
    {name: "Lisa", id: 103}];


//1. array.find(element)
    let found = numbers.find((element) => element < 0);
    let found2 = values.find((element) => element % 2 != 0);
    let found3 = student.find((element) => element.name.startsWith('I'));

console.log(found);
console.log(found2);
console.log(found3);
```

Output จะเห็นได้ว่า เมื่อมีการเรียกใช้ method find() แล้ว ฟังก์ชันก็จะทำการตรวจสอบเงื่อนไขของอาร์เรย์นั้นๆ ว่าตรงตามเงื่อนไขหรือเปล่า โดยหากเจอ element ที่ต้องการแล้ว จะทำการ return ออกมาเป็นค่านั้นๆ เช่น



found จะทำการค้นหา element ทั้งหมดใน array นี้ว่า มีตัวใดบ้างที่มีค่าน้อยกว่า 0 จึงกำหนดฟังก์ชั่นให้ element ทำการ return: element < 0

found2 จะทำการค้นหา element ทั้งหมดใน array นี้ว่า มีตัวใดบ้างที่หารสองไม่ลงตัว จึงกำหนดฟังก์ชั่นให้ element ทำการ return: element % 2 != 0

found3 จะทำการค้นหา element ทั้งหมดใน array นี้ว่า มีตัวใดบ้างที่ขึ้นต้นด้วยตัวอักษร ‘I’ จึงกำหนดฟังก์ชั่นให้ element ทำการ return: element.name.startsWith(‘I’)

2. find((element, index) => {...})

วิธีใช้ ตัวแปรอาเรย์ตามด้วย .find(element, index); เราสามารถดึงรายการอาร์เรย์โดยใช้ index เฉพาะของรายการจาก items ทั้งหมดได้ โดยกำหนด index ไว้ต่อจาก element

ตัวอย่าง

```
let values = [-2,2,4,6,8,10,12];
const items = ['phone', 'hammer', 'bolt', 'doll'];
let found = items.find((element, index) => {
  return index > 1 && element.startsWith('d')
});
let found1 = items.find((element, index) => {
  return index > 0 && element.length==6
});
let found2 = values.find((element, index) => {
  return index < 2 && element%2 ===0
});
console.log(found);
console.log(found1);
console.log(found2);
```

Output

```
[Running] node "/Users/giftr
doll
hammer
-2

[Done] exited with code=0 in
```

found จะทำการค้นหาและ return ค่าในarray ที่ index >1 และ element ที่เริ่มต้นด้วย ‘d’

found1 จะทำการค้นหาและ return ค่าในarray ที่ index > 0 และ element มีความยาว 6

found2 จะทำการค้นหาและ return ค่าในarray ที่ index <2 และ element%2 มีค่า 0

3. find((element, index, array) => {...})

วิธีใช้ เมื่อเรียกใช้งาน **method find(function(...))** แบบใส่ 3 parameter ใน function ของ find() จะทำการหาค่าตัวแรกใน array ที่โดน return ออกมากจาก fucntion ที่เป็น parameter ใน find โดยใน function จะสามารถใช้งาน arguments เหล่านี้ได้ เพื่อใช้สร้างเป็นเงื่อนไขในการ return ค่า

ได้แก่ element ใช้ในการแทนค่าของ element ในรอบนั้น ๆ

index ใช้ในการแทนค่าของ index ในรอบนั้น ๆ

array ใช้ในการแทนค่าของ array ที่เรียกใช้งานโดย find(function(...))

ตัวอย่าง ในการใช้ find กับ array ที่เก็บข้อมูล 3 แบบต่างกัน

```
const num = [9, 1, 8, 2, 7, 3, 6, 4, 5];
const obj = [{id: 0, name: 'Alex'},
             {id: 1, name: 'Bob'},
             {id: 2, name: 'Chad'},
             {id: 3, name: 'Dan'},
             {id: 99, name: 'Alex'}];
const arr = [[5, 4],
             [4, 6],
             [2, 4],
             [9, 0],
             [7, 8],
             [1, 3]];

const found1 = num.find((element, index, arr) => {
    return element > arr[arr.length-1] && index > 1;
})

const found2 = obj.find((element, index, arr) => {
    return element.name == arr[0].name && index != 0;
})

const found3 = arr.find((element, index, arr) => {
    return element[0] + element[1] == arr[0][0] + arr[0][1] && index != 0;
})

console.log(found1)
console.log(found2)
console.log(found3)
```

Output

```
[Running] node "c:\Users\benbe\Documents\INT201\63130500083-MyWorks\Assignment\Find\1.js"
8
{ id: 99, name: 'Alex' }
[ 9, 0 ]

[Done] exited with code=0 in 0.097 seconds
```

found1 จะทำการหา หาค่าที่มากกว่า num ที่ index สุดท้าย (5) และต้องมี index ใน num มากกว่า 1 คำตอบคือ 8

found2 จะทำการหา obj ใน array ชื่อ obj ที่มี name เหมือนกับ obj ที่ index แรก (id = 0, name = Alex) และต้องไม่ใช่ obj ที่ index แรก obj คำตอบคือ {id = 99, name = Alex}

found3 จะทำการหา array ใน arr ที่มีผลรวม เท่ากับ array ที่ index แรก([5, 4] 5+4 เท่ากับ 0) และต้องไม่ใช่ array ที่ index แรก คำตอบคือ [9, 0]