

Array methods

- Arrays inherit properties from `Array.prototype`, which defines a rich set of array manipulation methods
 1. `forEach()` & `shift()` & `unshift()`
 2. `map()` & `concat()`
 3. `every()` & `indexOf()`
 4. `some()` & `include()`
 5. `slice()` & `reverse()`
 6. `splice()` (insert & replace & delete)
 7. `fill()` & `find()`
 8. `push()` & `pop()` & `filter()`
 9. `sort()` & `findIndex()`
 10. `join()` & `reduce()`

รายละเอียดการส่งงาน

1. Method Syntax (ทำเฉพาะส่วนที่พารามิเตอร์เป็น primitive types, array, object หรือ arrow function)
2. คำอธิบายความสามารถของ method กระชับและชัดเจน
3. วิธีการใช้ method นั้น ระบุรายการพารามิเตอร์ อธิบายความหมายและชนิดข้อมูล และ output ที่ได้จาก method)
4. ตัวอย่างการใช้งาน ของแต่ละ syntax ให้มีหลาย ๆ กรณีศึกษาที่แตกต่างกัน อย่างน้อย 3 ตัวอย่าง

Github link : <https://github.com/messipeam/INT201-G02-GroupWorks-04.git>

Array.prototype.map()

Syntax

`Array.map(function(currentValue, index, arr), thisValue)`

Arrow function

```
map((element) => {...})  
map((element, index) => {...})  
map((element, index, array) => {...})
```

Ability

- มีการสร้าง array ใหม่พร้อมกับผลลัพธ์ของฟังก์ชันทุกๆ array element
- มีการเรียกใช้ฟังก์ชันที่มีการสร้างเอาไว้ สำหรับแต่ละ element ใน array ตามลำดับ
- ไม่มีการเรียกใช้งานฟังก์ชันถ้าหากมี elements เป็น empty
- ไม่มีการเปลี่ยนค่า original ของ array

How to use**Array.map(function(currentValue, index, arr), thisValue)**

Parameter Values	Description		
function(currentValue, index, arr)	(จำเป็น)Function มีการ run ในแต่ละ element ใน array โดยผ่าน arguments ดังนี้		
	<table><tr><td>arguments</td><td>Description</td></tr></table>	arguments	Description
	arguments	Description	
	<table><tr><td>currentValue</td><td>(จำเป็น) value ของ element ปัจจุบัน</td></tr></table>	currentValue	(จำเป็น) value ของ element ปัจจุบัน
	currentValue	(จำเป็น) value ของ element ปัจจุบัน	
	<table><tr><td>index</td><td>ตำแหน่งปัจจุบัน array index ของ element</td></tr></table>	index	ตำแหน่งปัจจุบัน array index ของ element
index	ตำแหน่งปัจจุบัน array index ของ element		
<table><tr><td>arr</td><td>The array object the current element belongs to</td></tr></table>	arr	The array object the current element belongs to	
arr	The array object the current element belongs to		
thisValue	Value ที่ถูกส่งไปที่ฟังก์ชันจะถูกใช้งานผ่านคำสั่ง “this” value		

Arrow function

```
map((element) => {...})  
map((element, index) => {...})  
map((element, index array) => {...})
```

Parameter Values	Description
<code>map((element, index, array) => {...})</code>	<p><code>element</code> -> ค่า element ใน Arrayที่กำลังถูกประมวลผลใน Array ตัวนั้น</p> <p><code>index</code> -> index หรือ ตำแหน่ง ของ Array ในแต่ละ elementsที่กำลังถูกประมวลผลอยู่</p> <p><code>array</code> -> array object ที่อยู่ใน elements ปัจจุบัน</p>

Return value: Array ที่ได้จะมี results จากการเรียกใช้ฟังก์ชันกับค่า array ที่เป็นค่าเริ่มต้น

Example 3 cases of each syntax

Array.map(function(currentValue, index, arr), thisValue)

case 1 : จะ return new array กับค่ารากที่สองของ original value

Code :

```
const numbers = [4, 9, 16, 25];  
const newArr = numbers.map(Math.sqrt)    // 2,3,4,5  
  
console.log(newArr);
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[ 2, 3, 4, 5 ]

[Done] exited with code=0 in 0.056 seconds
```

case 2 : เป็นการเรียกใช้ฟังก์ชันเพื่อทำการ *10 ให้กับค่าใน array ทุก element

Code :

```
const numbers = [65, 44, 12, 4];
const newArr = numbers.map(myFunction)

function myFunction(num) {
  return num * 10;
}
console.log(newArr);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[ 650, 440, 120, 40 ]

[Done] exited with code=0 in 0.057 seconds
```

case 3 : ใช้ในการรวม first name และ lastname เข้าด้วยกัน กลายเป็น fullname

Code :

```
const persons = [
  {firstname : "Malcom", lastname: "Reynolds"},
  {firstname : "Kaylee", lastname: "Frye"},
  {firstname : "Jayne", lastname: "Cobb"}
];

function getFullName(item) {
  return [item.firstname,item.lastname].join(" ");
}

console.log(persons.map(getFullName));
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[ 'Malcom Reynolds', 'Kaylee Frye', 'Jayne Cobb' ]

[Done] exited with code=0 in 0.058 seconds
```

Arrow function

`map((element) => {...})`

case 1 : หากเรามีรายชื่อหนัง 1 ชุดเก็บเป็น Array แต่รายชื่อหนังที่ถูกเก็บไว้นั้นขึ้นต้นด้วยตัวพิมพ์ใหญ่ และลงท้ายด้วยตัวพิมพ์เล็กทั้งหมด แต่เราอยากให้อาจอ้างรายชื่อหนังนั้นเปลี่ยนเป็นตัวพิมพ์ใหญ่ทั้งหมด ดังนั้น เราสามารถนำ method `map()` มาช่วยในกรณีนี้ได้

Code :

```
let movies = ['Spiderman','Iron man','Batman','Avengers','Star wars','Back to the future','Harry Potter']

let movieToUppercase = movies.map((nameMovie) => {
    return nameMovie.toUpperCase();
})

console.log(movieToUppercase);
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[
  'SPIDERMAN',
  'IRON MAN',
  'BATMAN',
  'AVENGERS',
  'STAR WARS',
  'BACK TO THE FUTURE',
  'HARRY POTTER'
]

[Done] exited with code=0 in 0.059 seconds
```

case 2 : หากเรามีชุดข้อมูลตัวเลขเก็บเป็น Array เลขคู่ตั้งแต่ 2 - 20 เราสามารถหาค่ายกกำลัง 2 หรือค่ายกกำลังอื่นๆ ของชุดข้อมูลนี้ โดยใช้ `map()`

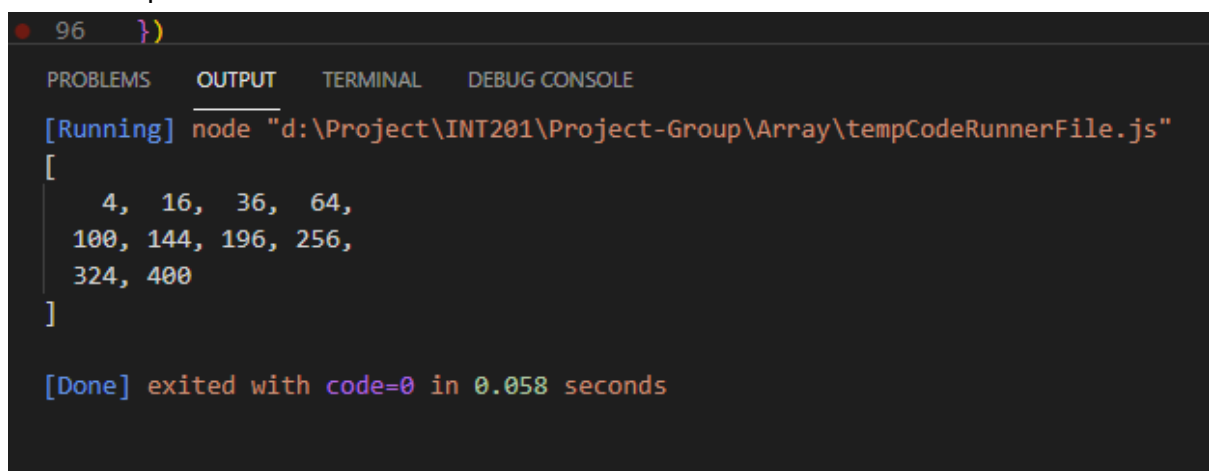
Code :

```
let num = [2,4,6,8,10,12,14,16,18,20]

let result = num.map((x) =>{
    return Math.pow(x,2) // return ค่าโดยใช้คำสั่ง Math.pow(base, exponent)
    โดยการแทน base เป็นค่า x ที่รับค่าแต่ละ element มาจาก attribute num ที่เก็บ ตัวเลขเป็น
    Array อยู่ และใส่ค่า exponent (เลขชี้กำลัง) เป็น 2 เพื่อยกกำลัง 2 ของค่าแต่ละ elements
    ใน Array
})

console.log(result);
```

Output :



96 })

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"

```
[
  4, 16, 36, 64,
  100, 144, 196, 256,
  324, 400
]
```

[Done] exited with code=0 in 0.058 seconds

case 3 : ตรวจสอบความยาวชื่อผลไม้ของแต่ละผลไม้ที่เก็บอยู่ใน Arrays ว่ามีตัวอักษรทั้งหมดกี่ตัว

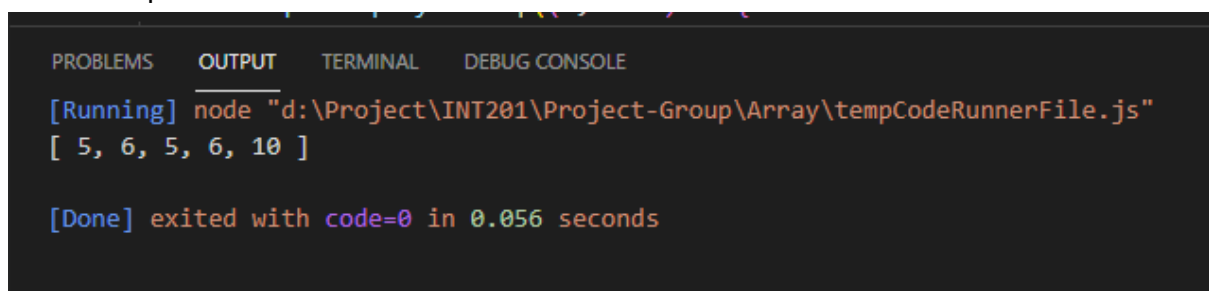
Code :

```
let fruit = ['Apple','Banana','Mango','Orange','Strawberry'];

let countString = fruit.map((x) =>{
    return x.length; //.length เพื่อเช็คความยาวของแต่ละ elements
})

console.log(countString);
```

Output :



PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"

```
[ 5, 6, 5, 6, 10 ]
```

[Done] exited with code=0 in 0.056 seconds

map((element, index) => {...})

case 1 : list รายชื่อพนักงานรวมกัน โดยให้แสดงลำดับที่ของพนักงานคนนั้นว่าเป็นพนักงานคนที่เท่าไร

Code :

```
let employees = ['John', 'Peter', 'Smith', 'Owen'];

let listEmp = employees.map((x, index) => {
    return `Employer number ${index+1} : ${x}` //index+1 เพื่อให้
index เริ่มต้นด้วย 1
})

console.log(listEmp);
```

Output :

```
[Done] exited with code=0 in 0.059 seconds

[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[
  'Employer number 1 : John',
  'Employer number 2 : Peter',
  'Employer number 3 : Smith',
  'Employer number 4 : Owen'
]

[Done] exited with code=0 in 0.069 seconds
```

case 2: นำค่าใน array * ด้วยค่า index ใน Arrays ของแต่ละ elements

Code :

```
let number = [1, 2, 3, 4];

let numbers = number.map((x, index) => {
    return x*index;
})

console.log(numbers);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[ 0, 2, 6, 12 ]  
  
[Done] exited with code=0 in 0.056 seconds
```

case 3: list รายชื่อนักฟุตบอลรวมกัน โดยให้แสดงลำดับที่ของนักฟุตบอลคนนั้นว่าเป็น player คนที่เท่าไร

Code :

```
let players = ['Neymar', 'Messi', 'Ronaldo', 'Kaka'];  
  
let listplayers = players.map((x, index) => {  
    return `Player ${index+1} : ${x}`;  
})  
  
console.log(listplayers);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[  
  'Player 1 : Neymar',  
  'Player 2 : Messi',  
  'Player 3 : Ronaldo',  
  'Player 4 : Kaka'  
]
```


map((element, index, array) => {...})

case 1 : คำนวณคะแนนทั้งหมดของนักเรียนแต่ละคน เก็บเป็น property ใหม่ที่ชื่อ totalScore ไว้ใน object

Code :

```
const students = [
  {name: 'Somchai', score : 30, score2 : 50},
  {name: 'Thanakrit', score : 40, score2 : 80}
]
const newStudents = students.map((student) =>({
  ...student, totalScore : student.score + student.score2
  //ใช้ spread operator ในการกระจายข้อมูลที่อยู่ภายใน object students ออกมา
  //กลายเป็นตัวแปรใหม่ และทำการเพิ่ม property totalScore เข้าไป
}))
console.log(newStudents);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[
  { name: 'Somchai', score: 30, score2: 50, totalScore: 80 },
  { name: 'Thanakrit', score: 40, score2: 80, totalScore: 120 }
]

[Done] exited with code=0 in 0.067 seconds
```

case 2: หาราคารวมของสินค้าทั้งหมดที่อยู่ใน object

Code :

```
const products = [
  {name:'Iphone 13', price : 42000, brand: 'Apple',color : 'Silver'},
  {name:'Laptop' ,price : 25000, brand: 'MSI', color:'Black'},
  {name:'Camera' ,price : 30000, brand: 'Nikon', color: 'Black'},
  {name:'Printer' ,price : 5400, brand: 'Canon', color: 'White'},
]; //สร้าง products เป็น Arrays เก็บ object ของสินค้าในแต่ละ element
let total = 0
const getProductPrice = products.map(products => { //เรียก Array
ของ products เพื่ออ้างถึง Object ภายใน ในการเรียก property price มาคำนวณราคา
  total += products.price;
});

console.log(total);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
102400
```

```
[Done] exited with code=0 in 0.062 seconds
```

case 3: แสดงจำนวนของประชากรครึ่งหนึ่งของแต่ละประเทศ

Code :

```
const population = [
  {country:'Thailand', population : 69800000},
  {country:'India' ,population: 1830000000},
  {country:'Japan' ,population : 125800000},
  {country:'England' ,population : 55980000},
]
let totalPopulation = 0;

const halfpopulation = population.map(population => {
  return {...population, population : population.population / 2
}
  return population;
});
```

```
console.log(halfpopulation);
```

Output :

```
[Running] node d:\Project\INI201\Project-Group\Array\tempco
[
  { country: 'Thailand', population: 34900000 },
  { country: 'India', population: 915000000 },
  { country: 'Japan', population: 62900000 },
  { country: 'England', population: 27990000 }
]

[Done] exited with code=0 in 0.063 seconds
```

Array.prototype.concat()

Syntax

```
concat()  
concat(value0)  
concat(value0, value1)  
concat(value0, value1, ... , valueN)
```

Ability

- สร้างตัวแปร array ใหม่จากการรวมตัวแปร array ตั้งแต่ 2 ตัวขึ้นไป
- ตัวแปร array ที่นำมารวมกัน ยังเหมือนเดิม ไม่มีการเปลี่ยนแปลงใดๆ
- return array ใหม่

How to use

method concat จะทำการสร้าง Array ตัวใหม่ขึ้นมา ซึ่งประกอบด้วย elements ที่ถูกเรียกเข้ามาตามลำดับ

parameter value	description
concat(value1, value2, ... value_n)	สามารถใส่ค่าที่เป็น Arrays หรือ values ที่เป็น primitive type ที่จะต่อกันเป็น Array ใหม่ลงไป ใน parameter ได้

return value -> method concat จะ return ค่าเป็น Array ตัวใหม่ขึ้นมา ซึ่งประกอบด้วย elements ที่ถูกเรียกเข้ามาต่อกันของแต่ละ parameters ตามลำดับ

Example 3 cases of each syntax

syntax -> concat()

case 1 :

Code :

```
let string = ['Hello', 'Hi'];  
let greeting = string.concat();  
console.log(greeting);
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[ 'Hello', 'Hi' ]  
  
[Done] exited with code=0 in 0.062 seconds
```

case 2 :

Code :

```
let books = ['Naruto', 'Attack on titan', 'One-piece', 'Dragonball'];  
let japanese_books = books.concat();  
console.log(japanese_books);
```

Output :

```
[Done] exited with code=0 in 0.062 seconds  
  
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[ 'Naruto', 'Attack on titan', 'One-piece', 'Dragonball' ]  
  
[Done] exited with code=0 in 0.059 seconds
```

case 3 :

Code :

```
let colors = ['Red', 'Green', 'Blue'];  
let rgb = colors.concat();  
console.log(rgb);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[ 'Red', 'Green', 'Blue' ]  
  
[Done] exited with code=0 in 0.057 seconds
```

syntax -> concat(value0)

case 1 : รวมรายชื่อเกมเข้าด้วยกันเป็นรายการเกมทั้งหมด

Code :

```
// สร้าง Attribute ของเกมเป็น Array ที่เก็บรายชื่อเกมขึ้นมา จำนวน 2 Attribute
const game1 = ['Mario', 'Fifa', 'Final Fantasy'];
const game2 = ['ROV', 'Dota2', 'CSGO'];
const games = game1.concat(game2); // หากเราอยากนำรายชื่อเกมจาก Attribute
game1 game2 มารวมกัน สามารถทำได้โดยใช้คำสั่ง concat()
console.log(games);
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[ 'Mario', 'Fifa', 'Final Fantasy', 'ROV', 'Dota2', 'CSGO' ]

[Done] exited with code=0 in 0.056 seconds
```

case 2:

Code :

```
const face = ['face', 'eyes', 'mouse', 'nose'];
const body = ['hand', 'legs', 'neck', 'fingers'];
const num = [1, 2, 3, 4, 5, 6, 7, 8, 9, 0];
const result = face.concat(body, num);

console.log(result);
```

Output :

```
PS E:\ppp\Teamwork\sophomore\semester1\INT201> node lab.js
[
  'face', 'eyes',   'mouse',
  'nose', 'hand',   'legs',
  'neck', 'fingers', 1,
  2,      3,        4,
  5,      6,        7,
  8,      9,        0
]
```

case 3 :

Code :

```
const fighter = ['airi', 'mortos', 'amily', 'dextra', 'riktor'];
const mage = ['lggy', 'zata', 'lorion', 'lauriel', 'nataya'];
const carry = ['thorns', 'joker', 'capheny', 'violet', 'laville'];
const support = ['zip', 'aya', 'ishar', 'annetle', 'sephera']
```

```
const result = fighter.concat(mage, carry, support);

console.log(result);
```

Output :

```
PS E:\pprrreaww\sophomore\semester1\INT201> node lab.js
[
  'airi',    'mortos',  'amily',
  'dextra',  'riktor',  'lggy',
  'zata',    'lorion',  'lauriel',
  'nataya',  'thorns',  'joker',
  'capheny', 'violet',  'laville',
  'zip',     'aya',     'ishar',
  'annetle', 'sephera'
]
```

syntax -> concat(value0,value1)

case 1 :สร้าง attribute letter เก็บค่าแบบ array, สร้าง attribute number ให้ letter เรียกใช้ concat เพื่อรวมค่าของ letter และรวมเพิ่มค่าเข้าไป

code :

```
const letter = ['a', 'b', 'c'];
const number = letter.concat(1, [2, 3]);

console.log(number);
```

output :

```
PS E:\pprrreaww\sophomore\semester1\INT201> node lab.js
[ 'a', 'b', 'c', 1, 2, 3 ]
PS E:\pprrreaww\sophomore\semester1\INT201>
```

case 2:สร้าง attribute Cartoon1 เพื่อเก็บค่าที่เป็น String ใน array, สร้าง attribute Cartoon2 ให้ Cartoon1 เรียกใช้ concat เพื่อรวมค่าของ Cartoon1 และค่าที่เพิ่มเข้าไป

code :

```
const Cartoon1 = ['kitty', 'minion', 'babies'];
const Cartoon2 = Cartoon1.concat('tom&jerry', ['doraemon',
'barebears']);

console.log(Cartoon2);
```

output :

```
PS E:\pprrreaww\sophomore\semester1\INT201> node lab.js
[ 'kitty', 'minion', 'babies', 'tom&jerry', 'doraemon', 'barebears' ]
```

case 3:สร้าง attribute num และ char เก็บค่าใน array และสร้าง result ให้ num เรียก concat เพื่อ

รวมค่า num char และค่าที่เพิ่มไป

code :

```
const num = [1,2,3,4,5,6,7,8,9,0];  
const char = ['A','B',' ','C','D','E','F'];  
const result = num.concat(char,['AAA',444]);  
  
console.log(result);
```

output :

```
PS E:\pprrreaww\sophomore\semester1\IN1201> node lab.js  
[  
  1,    2,    3,    4,  
  5,    6,    7,    8,  
  9,    0,    'A',  'B',  
  'C',  'D',  'E',  'F',  
  'AAA', 444  
]
```


syntax -> concat(value0, value1, ..., valueN)

case 1:

Code :

```
const num1 = [1,2,3,4,5];
const num2 = [6,7,8,9,0];
const num3 = [11,12,13,14,15];
console.log(num1.concat(num2,num3));
```

Output :

```
PS E:\pprrreaww\sophomore\semester1\INT201> node lab.js
[
  1, 2, 3, 4, 5, 6,
  7, 8, 9, 0, 11, 12,
  13, 14, 15
]
```

case 2: รวบรวมรายชื่อนักแสดงหนัง Hollywood

Code :

```
const name1 = ['Jennifer Lawrence','Tom Holland','Benedict Cumberbatch'];
const name2 = ['Emma Stone','Emma Watson'];
const name3 = ['Chris Hemsworth','Chris Evans','Robert Downey Jr.'];
const name4 = ['Scarlett Johansson','Chadwick Boseman'];
const actors = name1.concat(name2,name3,name4);
console.log(actors);
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"
[
  'Jennifer Lawrence',
  'Tom Holland',
  'Benedict Cumberbatch',
  'Emma Stone',
  'Emma Watson',
  'Chris Hemsworth',
  'Chris Evans',
  'Robert Downey Jr.',
  'Scarlett Johansson',
  'Chadwick Boseman'
]

[Done] exited with code=0 in 0.069 seconds
```

case 3: รวบรวมรายชื่อวิชาในสาขาไอทีเข้าด้วยกันเป็นวิชาทั้งหมด โดยรับ value ทั้ง Arrays และ Primitive type มารวมกัน

Code :

```
const courseIt1= ['INT202','INT207','INT205'];
const courseIt2 = ['INT214','INT202'];
const allCourseIt =
courseIt1.concat(courseIt2,'INT201','INT100','INT102',['INT104','INT105',
'INT107'],['INT200','INT114']) //<-- สามารถใส่ค่าเป็น Primitive type หรือใส่
ข้อมูลที่เป็น Arrays ลงไป ใน value ได้ ผลลัพธ์ที่ออกมาจะได้เป็น Arrays ตัวใหม่ที่ถูก
merge กันแล้ว

console.log(allCourseIt);
```

Output :

```
[Running] node "d:\Project\INT201\Project-Group\Array\tempCodeRunnerFile.js"  
[  
  'INT202', 'INT207',  
  'INT205', 'INT214',  
  'INT202', 'INT201',  
  'INT100', 'INT102',  
  'INT104', 'INT105',  
  'INT107', 'INT200',  
  'INT114'  
]  
  
[Done] exited with code=0 in 0.057 seconds
```