Git link: https://github.com/ideastation-x/INT201_A5_S2_Group3

calTax

index.html

```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
        <title>Calculate Tax</title>
    </head>
    <body>
        <!-- เรียกใช้งาน calTax.js มาแสดงผลบน Website ซึ่ง import จะไม่ถูกฝังใส่ script ถ้าไม่เป็น module
-->
        <script src="calTax.js" type="module"></script>
    </body>
</html>
```

getPrice.js

```javascript
// * Export "function getPrice(priceList)" เป็น Module เพื่อสามารถเอาไปใช้ใน script อื่น ๆ ได้
// * getPrice(priceList) เป็น function สำหรับรับค่ารายการสินค้าที่ยังไม่รวมภาษี
export function getPrice(priceList) {

    // * calculateTax(tax) เป็น function คำนวณภาษีแต่ละสินค้าชิ้น
    // * แล้ว return เป็นผลรวมของรายการสินค้าที่รวมภาษีแล้ว
    function calculateTax(tax) {

        // * sumPriceWithTax เป็นการรับค่าก่อนหน้า, และค่าปัจจุบัน แล้ว return ออกมาเป็นผลรวมของทั้งสองค่า
        // * ใช้ reduce โดยรับ parameter เป็น arrow function ที่รับค่าก่อนหน้า, และค่าปัจจุบัน แล้ว
return ออกมาเป็นผลรวมของทั้งสองค่า
        // * กับ priceList เพื่อบวกค่าหาผลรวมทั้งหมด
        return priceList.map((price) => price + (price *
tax)).reduce((previousValue, currentValue) => previousValue + currentValue)
    }

    // * return มาเป็นค่าผลรวมราคาของแต่ละชิ้นทั้งหมดที่รวมภาษีแล้ว
    return calculateTax
}
```

calTax.js

```
// * Import "function getPrice(priceList)" ด้วยชื่อ getPrice จาก ./getPrice.js เพื่อเอามาใช้ภายใน script นี้
import { getPrice } from './getPrice.js'

// * สร้าง TestCase เพื่อทดสอบ
let priceList = {
    testCase1: [10, 20, 30, 40, 50],
    testCase2: [50, 300, 80, 100, 460],
    testCase3: [168, 1890, 1987, 1237, 1984]
}
// * กำหนดภาษี
let tax = 0.07

// * เรียกใช้งาน Method getPrice แล้ว Assign ค่าให้ตัวแปรเพื่อให้สามารถเรียกใช้ Inner Function ผ่านตัวแปรได้
let calTotalPriceWithTax_case1 = getPrice(priceList.testCase1)

// * เรียกใช้งาน calTotalPriceWithTax_case1(tax) เพื่อคำนวณภาษีแล้วหาผลรวมของรายการสินค้าที่รวมภาษีแล้ว
console.log(`Price List : ${priceList.testCase1}\nTax : ${(tax *
100).toFixed(0)}%\nTotal Price : ${calTotalPriceWithTax_case1(tax)}`)

let calTotalPriceWithTax_case2 = getPrice(priceList.testCase2)
console.log(`Price List : ${priceList.testCase2}\nTax : ${(tax *
100).toFixed(0)}%\nTotal Price : ${calTotalPriceWithTax_case2(tax)}`)

let calTotalPriceWithTax_case3 = getPrice(priceList.testCase3)
console.log(`Price List : ${priceList.testCase3}\nTax : ${(tax *
100).toFixed(0)}%\nTotal Price : ${calTotalPriceWithTax_case3(tax)}`)
```

output

```
[Running] node "c:\Users\User\Desktop\INT201_A5_S2_Group3\calTax\calTax.js"
Price List : 10,20,30,40,50
Tax : 7%
Total Price : 160.5
Price List : 50,300,80,100,460
Tax : 7%
Total Price : 1059.3
Price List : 168,1890,1987,1237,1984
Tax : 7%
Total Price : 7774.62
```

Distance

formula.js

```
// * Export "function howLong(p1, p2)" เป็น Module เพื่อสามารถเอาไปใช้ใน script อื่น ๆ ได้
// * howLong(p1, p2) เป็น function สำหรับคำนวณระยะห่างระหว่างจุด โดยรับ parameter เป็นคู่อันดับของจุด 2 จุด
// * return เป็นผลลัพธ์ของระยะห่างระหว่างจุด 2 จุด
export function howLong(p1, p2) {
    return Math.sqrt(Math.pow((p1[0]-p2[0]), 2) + Math.pow((p1[1]-p2[1]), 2),
2)
}

// * Export "function sortLength(...lines)" เป็น Module เพื่อสามารถเอาไปใช้ใน script อื่น ๆ ได้
// * sortLength(...lines) เป็น function สำหรับเรียงค่าจากน้อยไปมาก
// * return array ของ lines ที่ถูกเรียงแล้ว
export function sortLength(...lines) {
    return lines.sort((a, b) => a - b);
}
```

calDistance.js

```
// * Import "function howLong(p1, p2) กับ sortLength(...lines)" ด้วยชื่อ howLong กับ
sortLength จาก formula.js เพื่อเอามาใช้ภายใน script นี้
import { howLong, sortLength } from './formula.js';

// * สร้าง TestCase เพื่อทดสอบ
// * โดยใส่ argument เป็น array ของคู่อันดับ
let l1 = howLong([3, 2], [7, 2]);
let l2 = howLong([5, 10], [5, 27]);
let l3 = howLong([36, 40], [2, 34]);

console.log(`The length of l1 is ${l1}`);
console.log(`The length of l2 is ${l2}`);
console.log(`The length of l3 is ${l3}`);

// * เรียกใช้ sortLength เพื่อเรียงลำดับความยาว
console.log(sortLength(l1, l2, l3));
```

output

```
[Running] node "c:\Users\User\Desktop\INT201_A5_S2_Group3\distance\tempCodeRunnerFile.js"
The length of l1 is 4
The length of l2 is 17
The length of l3 is 34.52535300326414
[ 4, 17, 34.52535300326414 ]
```

calArea

shape.js

```
// * Function ในการคำนวณพื้นที่ของรูปต่าง ๆ
export function area(fn, n1, n2 ) {
    // * return ผลลัพธ์ของ function กลับไปยัง area โดย Function จะได้รับมาจาก Paremeter -> fn ของ area และส่งค่า n1,n2 ไปยัง fn ที่ได้รับมา
    return fn(n1, n2);
}

// * Function ในการคำนวณพื้นที่ของวงกลม
export function circle(n1, n2 = 0) {
    // * return มาเป็นพื้นที่ของวงกลม
    // * กำหนด defualt parameter เนื่องจากการคำนวณพื้นที่วงกลมใส่ค่าเข้ามาเพียงค่าเดียว เพื่อไม่ให้เกิด error จากการใส่ค่าไม่ครบ
    return Math.PI * n1 * n1;
}

// * Function ในการคำนวณพื้นที่ของสามเหลี่ยม
export function triangle(n1, n2) {
    // * return มาเป็นพื้นที่ของสามเหลี่ยม
    return 0.5 * n1 * n2;
}

// * Function ในการคำนวณพื้นที่ของสี่เหลี่ยม
export function rectangle(n1, n2) {
    // * return มาเป็นพื้นที่ของสี่เหลี่ยม
    return n1 * n2;
}
```

calArea.js

```
// * Import "area, circle, triangle, rectangle" จาก ./shapes.js เพื่อเอามาใช้ภายใน script นี้
import { area, circle, triangle, rectangle } from "./shapes.js";

// * เรียกใช้งาน Method area แล้วส่ง Function การทำงานที่ได้สร้างไว้ผ่าน Parameter ของ area
let shape1 = area(circle, 10);
let shape2 = area(triangle, 3, 4);
let shape3 = area(rectangle, 25, 25);

// * แสดงผลลัพธ์ที่ return มาจาก area
console.log(`shape1: ${shape1}`);
console.log(`shape2: ${shape2}`);
console.log(`shape3: ${shape3}`);
```

output

```
[Running] node "c:\Users\User\Desktop\INT201_A5_S2_Group3\calArea\calArea.js"
shape1: 314.1592653589793
shape2: 6
shape3: 625
```