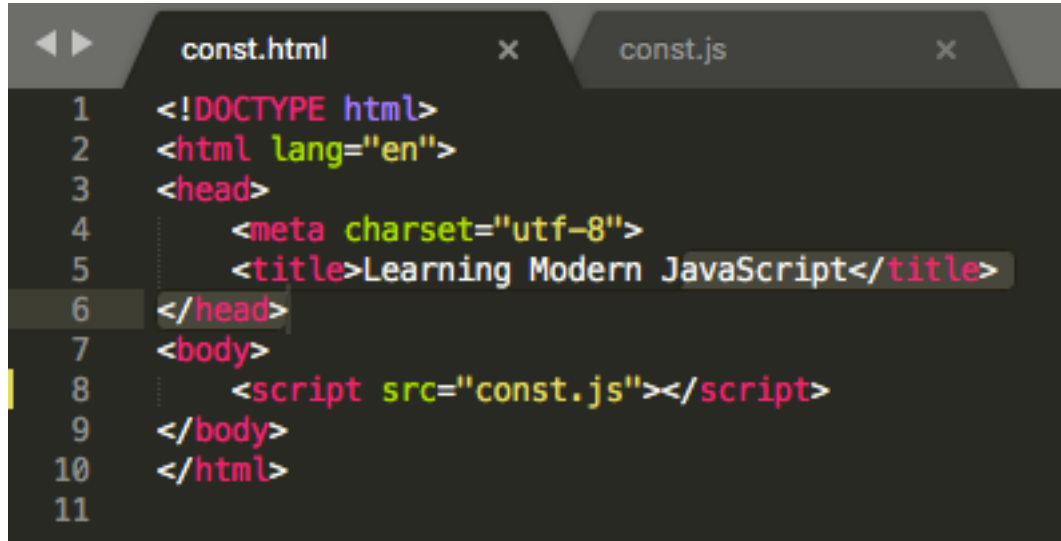
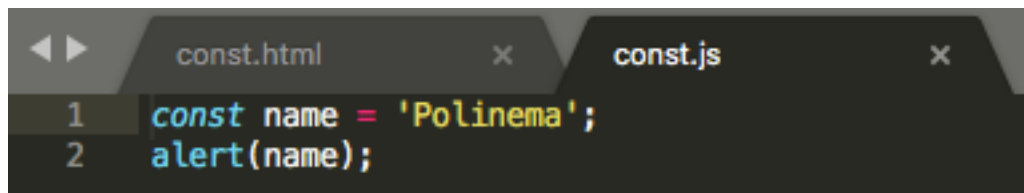


NAMA : IVAN CRISNANDA
KELAS : TI-3C
NO : 11

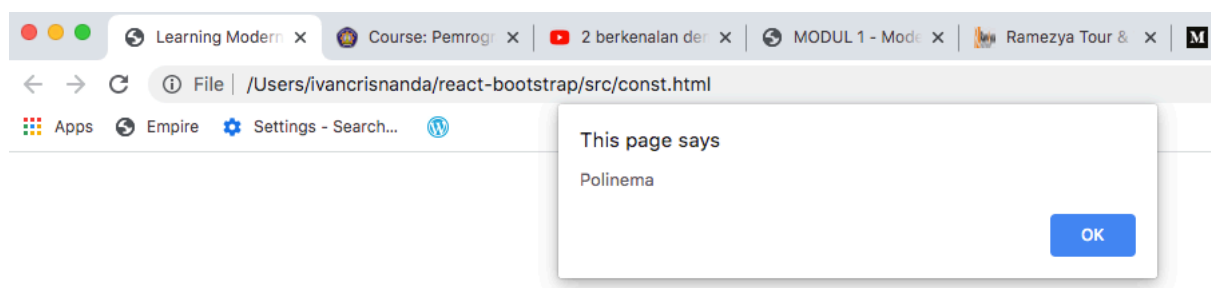
1. CONST



```
const.html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="utf-8">
5      <title>Learning Modern JavaScript</title>
6  </head>
7  <body>
8      <script src="const.js"></script>
9  </body>
10 </html>
11
```



```
const.js
1  const name = 'Polinema';
2  alert(name);
```

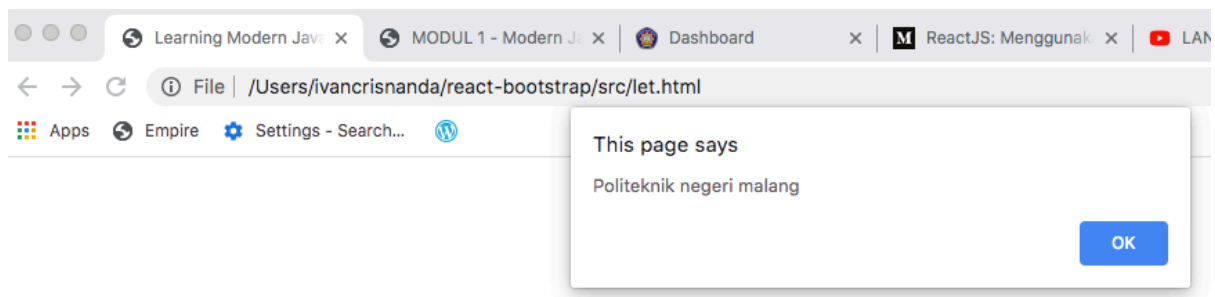


Variabel `const` digunakan untuk mendeklarasikan variable yang nilainya statis atau tidak berubah.

2. LET

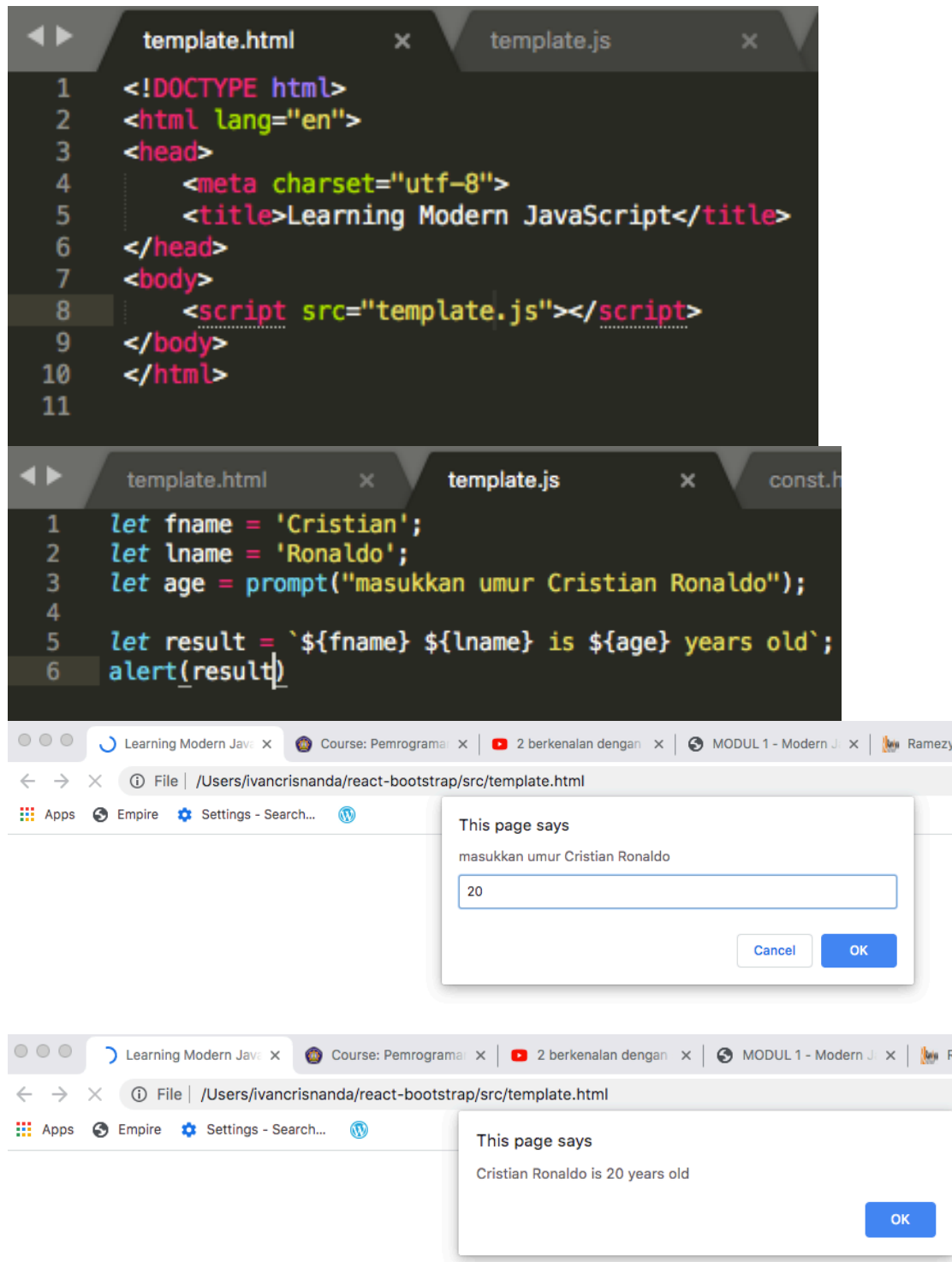
```
let.html x let.js x array.html x array.js
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="let.js"></script>
9 </body>
10 </html>
11
```

```
let.html x let.js x array.html x
1 if (true){
2   let name = 'Polinema';
3   name = 'Politeknik negeri malang';
4   alert(name);
5 }
```



Let merupakan block scope (Block scope dalam javascript di tandai dengan simbol { }). Scope artinya pembagian program, ini sering di temui pada if, for, switch, while dan sebagainya.) Pada code diatas mengapa yang tampil “Politeknik Negeri Malang”, karena variable tanpa deklarasi dianggap var

3. TEMPLATE

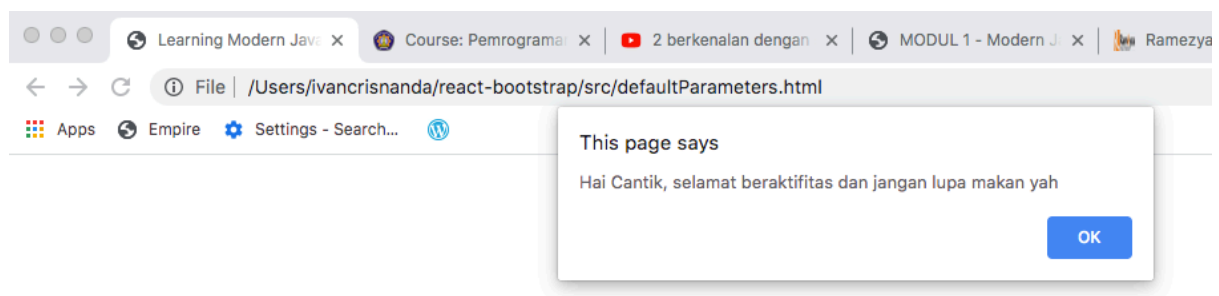


Code diatas menampilkan literal dan variable (kombinasi) tanpa menggunakan pemisan "+". Data non literal dapat diapit dengan menggunakan "" dan kemudian setelah itu letakkan dollar "\$" dan masukkan nama variable diantara dua bracket ({nama variable}). Sesuatu yang berada di dalam apitan tersebut akan diasosiasikan dengan variable yang berada dalam class/file javascript, operasi aritmatika, dan string method.

4. DEFAULT PARAMETERS

```
defaultParameters.html x defaultParameters.js x te
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="utf-8">
5     <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8     <script src="defaultParameters.js"></script>
9 </body>
10 </html>
11
```

```
defaultParameters.html x defaultParameters.js x template.html x template.js x const.html
1 function welcome(user = 'Cantik', message = 'selamat beraktifitas dan jangan lupa makan yah'){
2     alert(`Hai ${user}, ${message}`);
3 }
4
5 welcome();
```

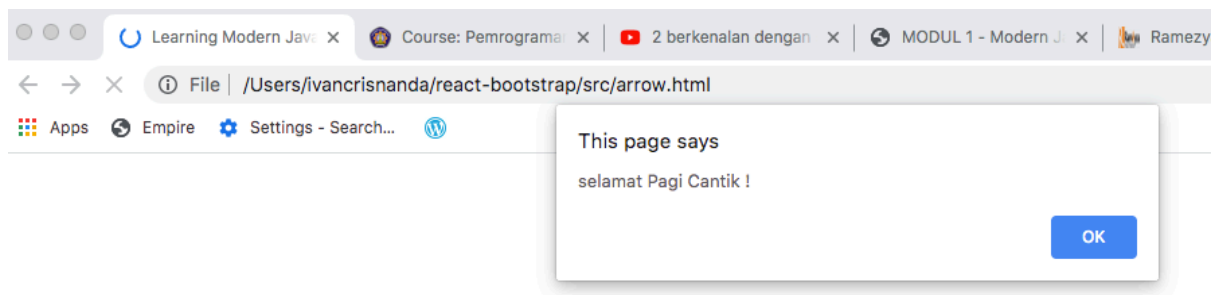


Default parameter memungkinkan parameter bernama diinisialisasi dengan nilai default jika tidak ada nilai atau tidak terdefinisi.

5. ARROW

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="arrow.js"></script>
9 </body>
10 </html>
11
```

```
1 let gretting = message => alert(`${message} Cantik !`);
2
3 gretting ('selamat Pagi');
```

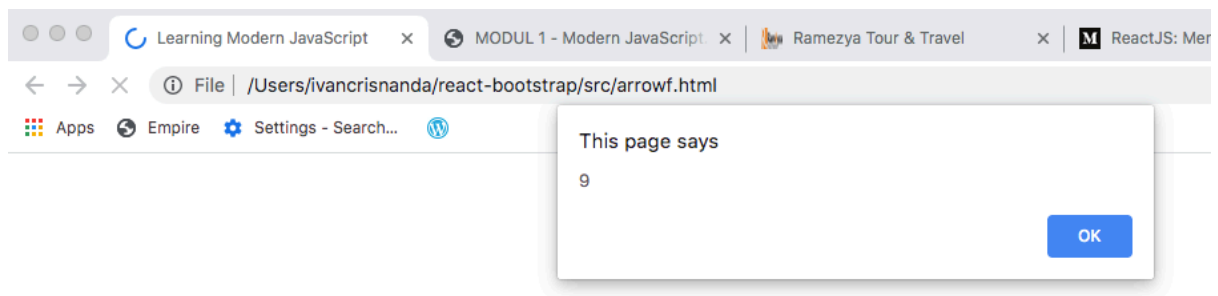


Arrow function Adalah penyederhanaan function, yang di tulis pada ES6 menggunakan tanda “=>”

6. ARROWF

```
1  cont func = (a,b)=> {  
2      return a + b;  
3  };  
4  alert(func(5,4));
```

```
1  const func = (a, b)=> {  
2      return a + b;  
3  };  
4  alert(func(5,4));
```

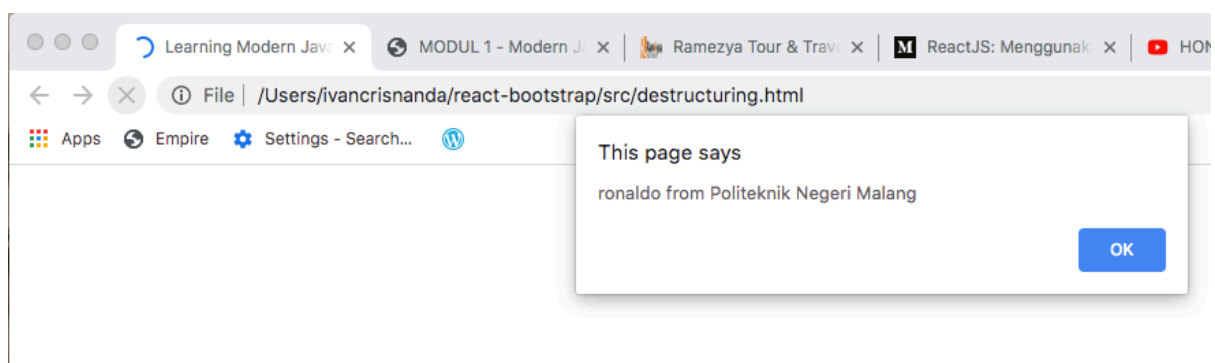


Arrow function Adalah penyederhanaan function, yang di tulis pada ES6 menggunakan tanda “=>”.

7. DESTRUCTURING

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="destructuring.js"></script>
9 </body>
10 </html>
11
```

```
1 let polStudent = ({name, polytechnic}) => {
2   alert(`${name} from ${polytechnic}`);
3 };
4
5 polStudent({
6   name: 'ronaldo',
7   polytechnic: 'Politeknik Negeri Malang'
8 });
```

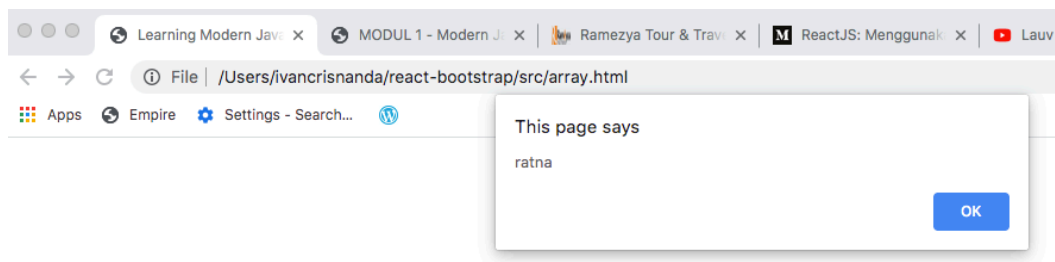


Destructuring adalah ekspresi JavaScript yang memungkinkan untuk mengurai nilai dari object, atau properti dari objek, ke dalam variabel yang berbeda.

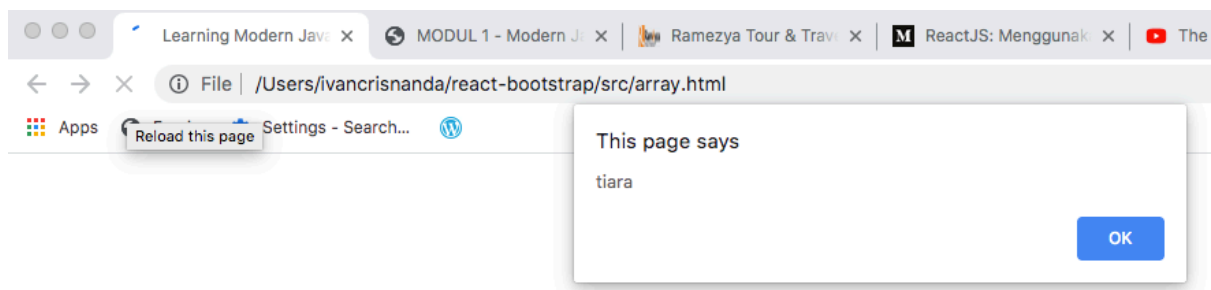
8. ARRAY

```
array.html x array.js x destructuring.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="array.js"></script>
9 </body>
10 </html>
11
```

```
array.html x array.js x destructuring.h
1 let [wife] = ['ratna', 'bunga', 'tiara'];
2 alert(wife);
```



```
array.html x array.js x destructuring.html
1 let [, , wife] = ['ratna', 'bunga', 'tiara'];
2 alert(wife);
```

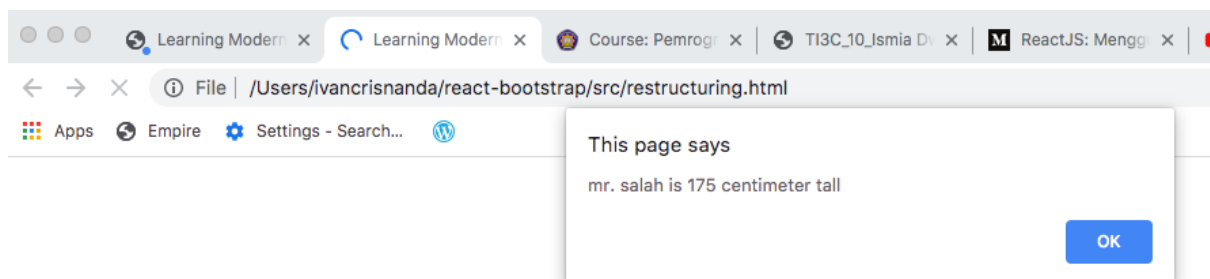


Destucturing adalah ekspresi JavaScript yang memungkinkan untuk mengurai nilai dari array, atau properti dari objek, ke dalam variabel yang berbeda.

9. RESTRUCTURING

```
restructuring.html x restructuring.js x let.ht
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="restructuring.js"></script>
9 </body>
10 </html>
11
```

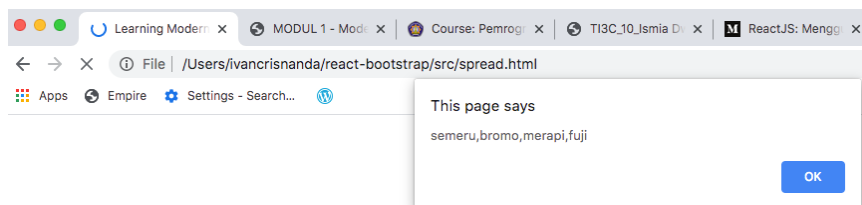
```
restructuring.html x restructuring.js x let.html x let.js x
1 var pemainSepakbola = {
2   name : 'salah',
3   height : '175',
4   output(){
5     alert(`mr. ${this.name} is ${this.height} centimeter tall`);
6   }
7 };
8 pemainSepakbola.output();
```



10. SPREAD

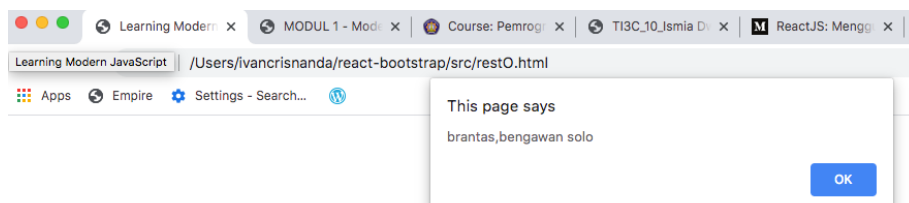
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="spread.js"></script>
9 </body>
10 </html>
```

```
1 var mountains = ['semeru', 'bromo', 'merapi'];
2 var mountainsFromJapan = ['fuji'];
3
4 var allMountains = [...mountains, ...mountainsFromJapan];
5 alert(allMountains);
```



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="rest0.js"></script>
9 </body>
10 </html>
```

```
1 var rivers = ['ciliwung', 'brantas', 'bengawan solo'];
2 var [first, ...rest] = rivers;
3
4 alert(rest);
```

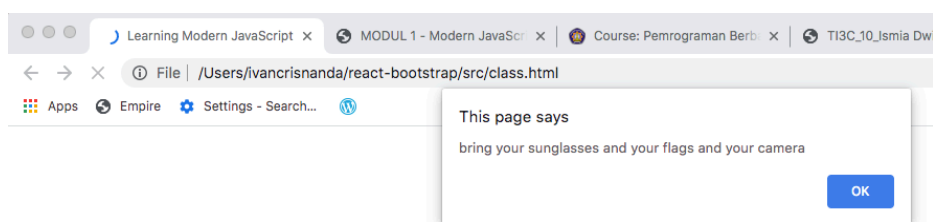
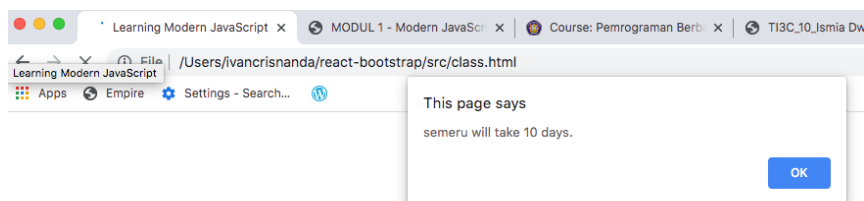


Spread Operator memiliki fungsi mengeluarkan item menjadi sebuah argument atau parameter sehingga dapat digunakan pada kondisi tertentu. Sedangkan rest parameter adalah variable yang dijadikan parameter sehingga jika terdapat beberapa parameter setelahnya itu maka akan masuk ke dalam rest parameter.

11. CLASS

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Learning Modern JavaScript</title>
6 </head>
7 <body>
8   <script src="class.js"></script>
9 </body>
10 </html>
11
```

```
1 class Holiday {
2   constructor(destination,days){
3     this.destination = destination;
4     this.days = days;
5   }
6
7   info(){
8     alert(`${this.destination} will take ${this.days} days.`);
9   }
10 }
11
12 class Expedition extends Holiday {
13   constructor(destination,days,gear){
14     super(destination,days);
15     this.gear = gear;
16   }
17
18   info(){
19     super.info();
20     alert(`bring your ${this.gear.join(' and your ')} `);
21   }
22 }
23 const tripWithGear = new Expedition('semeru', 10, ['sunglasses','flags','camera']);
24 tripWithGear.info();
```



Class constructor adalah sebuah method/function yang di jalankan pertama kali ketika object di buat. Sedangkan kata kunci super adalah untuk memanggil method super pada class super.

