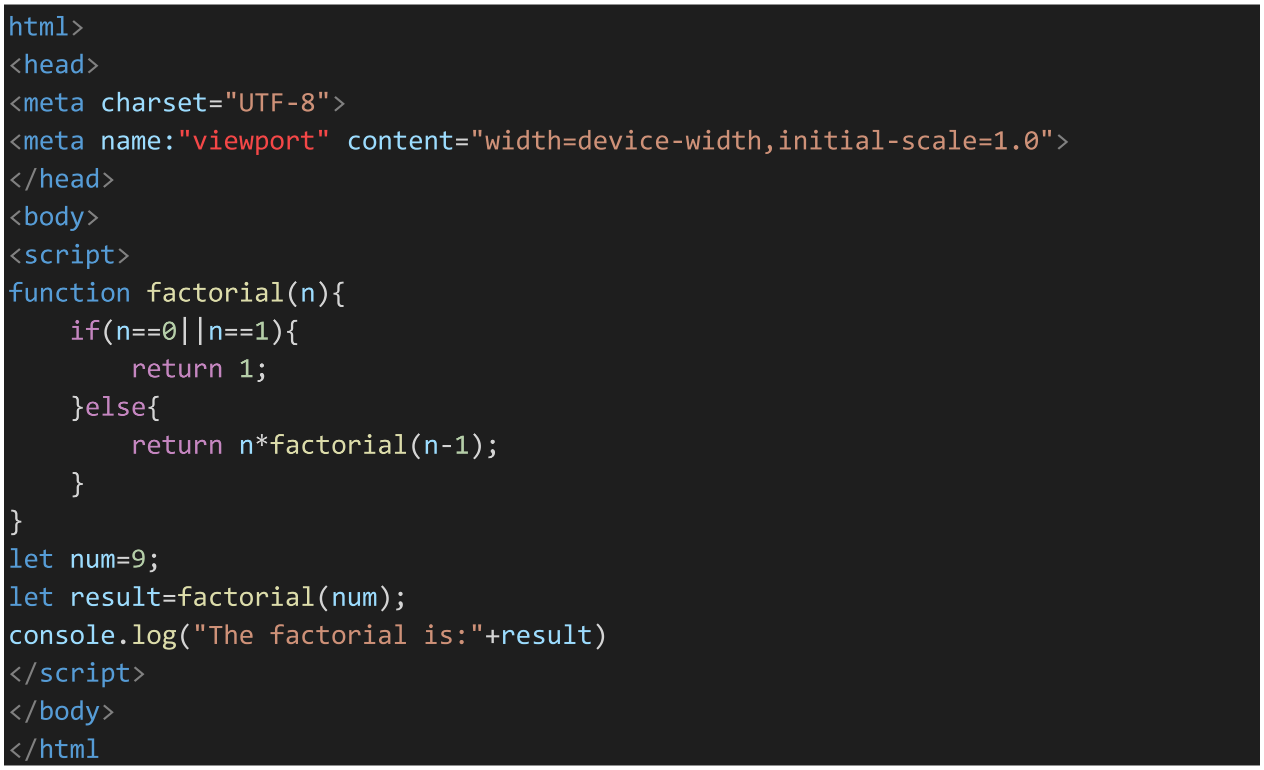
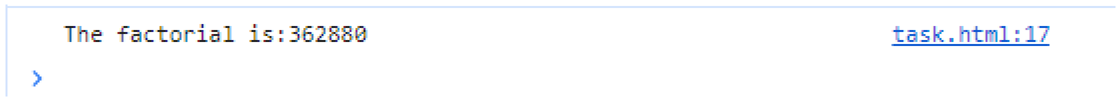
TASK 1:



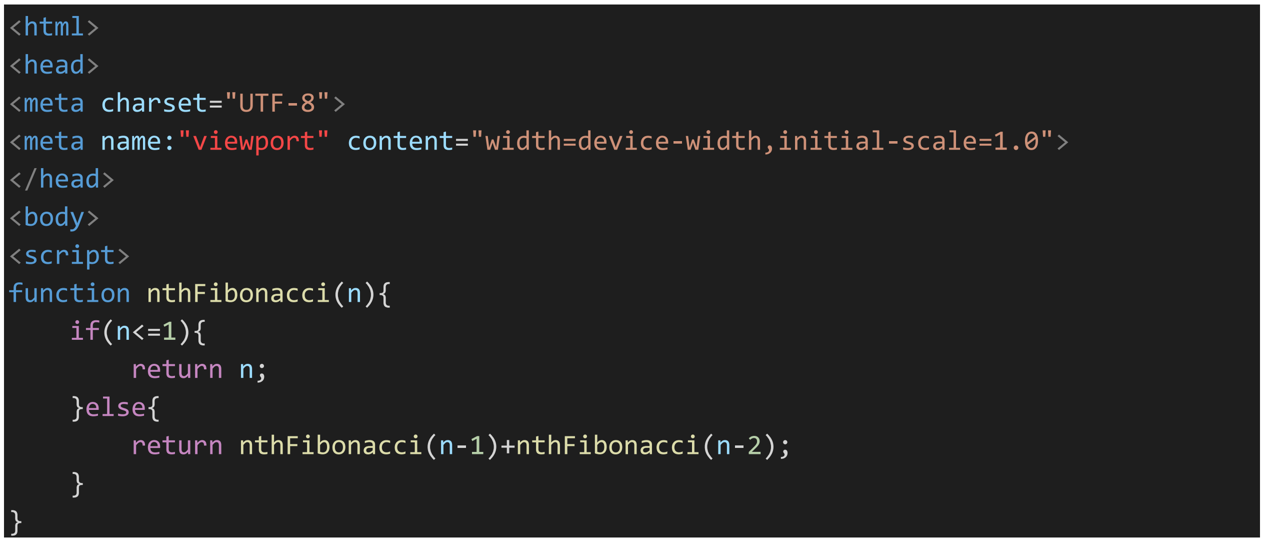
```
html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function factorial(n){
    if(n==0||n==1){
        return 1;
    }else{
        return n*factorial(n-1);
    }
}
let num=9;
let result=factorial(num);
console.log("The factorial is:"+result)
</script>
</body>
</html
```

OUTPUT:



```
The factorial is:362880                                    task.html:17
>
```

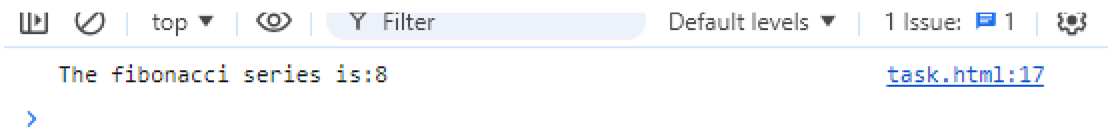
TASK 2:



```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function nthFibonacci(n){
    if(n<=1){
        return n;
    }else{
        return nthFibonacci(n-1)+nthFibonacci(n-2);
    }
}
```

```
let num=6;
let result=nthFibonacci(num);
console.log("The fibonacci series is:"+result);
</script>
</body>
</html
```

OUTPUT:

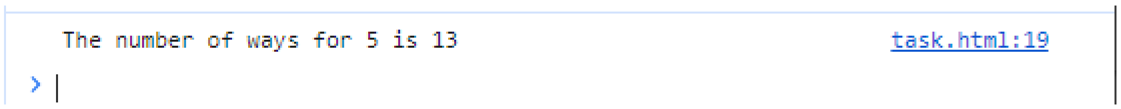


```
The fibonacci series is:8                                  task.html:17
>
```

TASK 3:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function countWays(n){
    if(n==0){
        return 1;
    }else if(n<0){
        return 0;
    }else{
        return countWays(n-1)+countWays(n-2)+countWays(n-3);
    }
}
let num=5;
let result = countWays(num);
console.log("The number of ways for " +num+ " is " +result);
</script>
</body>
</html
```

OUTPUT:

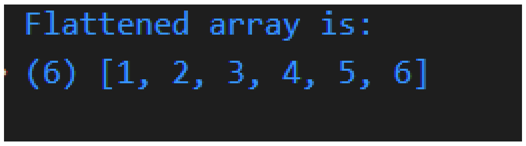


```
The number of ways for 5 is 13                             task.html:19
> |
```

TASK 4:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function customFlat(arr) {
    return arr.reduce((acc, val) =>
    acc.concat(Array.isArray(val) ?
    customFlat(val) : val), []);
}

const nestedArray = [1, [2, 3], [4, [5, 6]]];
console.log("Flattened array is: ")
console.log(customFlat(nestedArray));
</script>
</body>
</html
```

OUTPUT:



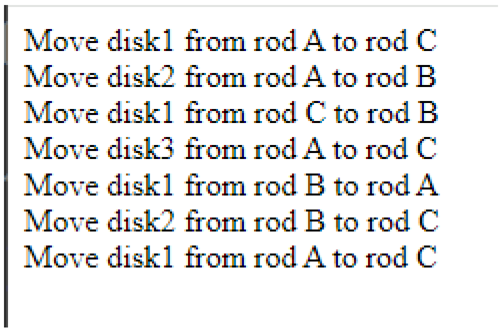
```
Flattened array is:
(6) [1, 2, 3, 4, 5, 6]
```

TASK 5:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function towerofHanoi(n,from_rod,to_rod,aux_rod){
    if(n==0){
        return;
    }
    towerofHanoi(n-1,from_rod,aux_rod,to_rod);
    document.writeln("Move disk"+ n +" from rod "+ from_rod + " to rod "
+  to_rod + "<br>" );
```

```
    towerofHanoi(n-1,aux_rod,to_rod,from_rod);
}
let N=3;
towerofHanoi(N,"A","C","B");
</script>
</body>
</html
```

OUTPUT:

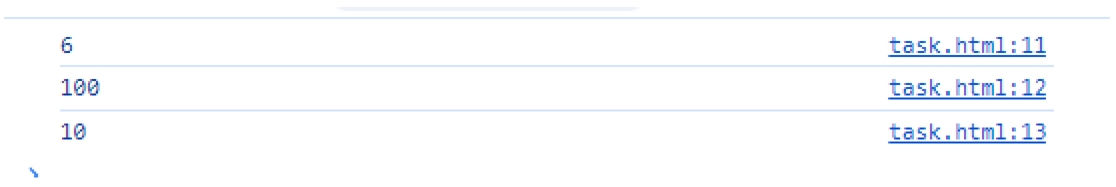


```
Move disk1 from rod A to rod C
Move disk2 from rod A to rod B
Move disk1 from rod C to rod B
Move disk3 from rod A to rod C
Move disk1 from rod B to rod A
Move disk2 from rod B to rod C
Move disk1 from rod A to rod C
```

TASK 6:<html>

```
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function sum(...args){
    return args.reduce((total,currentvalue)=>total+currentvalue,0);
}
console.log(sum(1,2,3));
console.log(sum(20,50,30));
console.log(sum(8,-2,9,-5));
</script>
</body>
</html
```

OUTPUT:

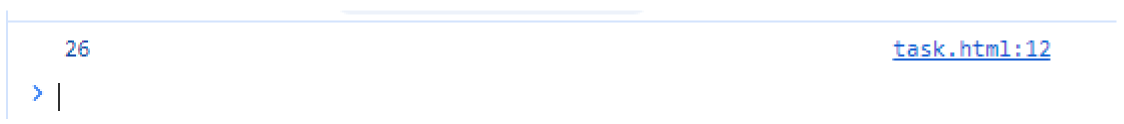


```
6                                                          task.html:11
100                                                        task.html:12
10                                                         task.html:13
```

TASK 7:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function sum(...nums){
    return nums.reduce((total,currentvalue)=>total+currentvalue,0);
}
let num=[2,5,8,2,9];
console.log(sum(...num));
</script>
</body>
</html
```

OUTPUT:



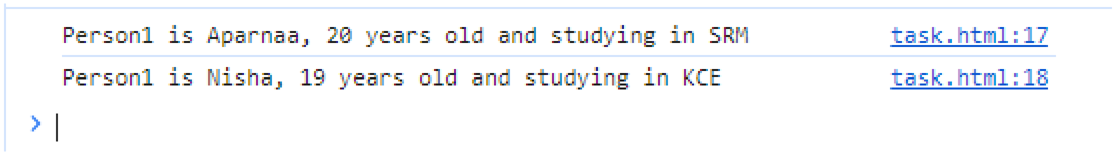
```
26                                                task.html:12
> |
```

TASK 8:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
let person1={
    name:"Nisha",
    age:"19",
    College:"KCE"
}
let person2 = {...person1};
person1.name="Aparnaa"
person1.age="20"
person1.College="SRM"
console.log("Person1 is "+ person1.name + ", "+ person1.age + " years old and
studying in " + person1.College );
```

```
console.log("Person1 is "+ person2.name + ", "+ person2.age + " years old and
studying in " + person2.College );
</script>
</body>
</html
```

OUTPUT:

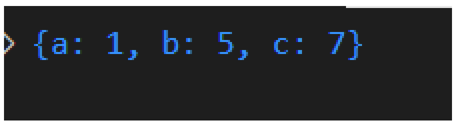


```
Person1 is Aparnaa, 20 years old and studying in SRM          task.html:17
Person1 is Nisha, 19 years old and studying in KCE            task.html:18
>
```

TASK 9:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function merge(obj1,obj2){
    return {...obj1,...obj2};
}
let obj1={a:1,b:2};
let obj2={b:5,c:6,c:7};
let currentobj=merge(obj1,obj2);
console.log(currentobj);
</script>
</body>
</html
```

OUTPUT:



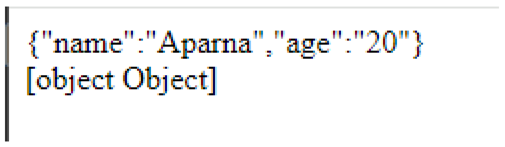
```
> {a: 1, b: 5, c: 7}
```

TASK 10:

```
<html>
<head>
<meta charset="UTF-8">
```

```
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
let person1={
    name:"Nisha",
    age:"20",
};
let person2={
    name:"Aparna",
    age:"20",
};
let obj={...person1,...person2};
document.writeln(JSON.stringify(obj));
document.writeln("<br>");
document.writeln(obj);
</script>
</body>
</html
```

OUTPUT:



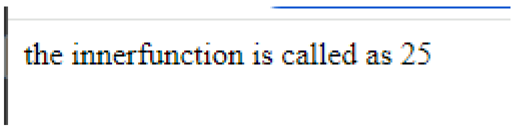
```
{"name":"Aparna","age":"20"}
[object Object]
```

TASK 11:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head>
<body>
<script>
function outerFunction(x){
    function innerFunction(){
        document.writeln("the innerfunction is called as ");
    }
       innerFunction();
return x*5;}
const func=outerFunction(5);
document.writeln(func);
```

```
</script>
</body>
</html
```

OUTPUT:

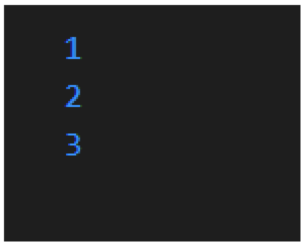


```
the innerfunction is called as 25
```

TASK 12:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function createCounter(){
    let count=0;
    return{
        increment: function(){
            count++;},
        getCount: function(){
            return count;} };}
const counter=createCounter();
counter.increment()
    console.log(counter.getCount());
    counter.increment()
    console.log(counter.getCount());
    counter.increment()
    console.log(counter.getCount());
</script>
</body>
</html
```

OUTPUT:

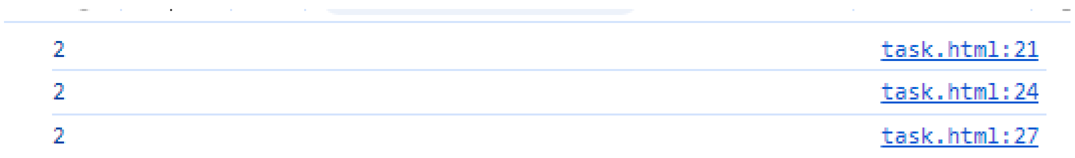


```
1
2
3
```

TASK 13:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function createCounter(){
    let count=0;
    return{
        increment: function(){
            count+=1;},
        getCount: function(){
            return count;} };}

            const counter1=createCounter();
            const counter2=createCounter();
            const counter3=createCounter();
    counter1.increment();
    counter1.increment();

console.log(counter1.getCount());
counter2.increment();
counter2.increment();
console.log(counter2.getCount());
counter3.increment();
counter3.increment();
console.log(counter3.getCount());

</script>
</body>
</html
```

OUTPUT:



```
2                                                      task.html:21
2                                                      task.html:24
2                                                      task.html:27
```

TASK 14:

```
<html>
<head>
<meta charset="UTF-8">
```

```
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function counterwithPrivate(){
    let count=0;
    return{
        increment: function(){
            count+=1;},
        getCount: function(){
            return count;} };}
            const Privatecounter=counterwithPrivate();
    Privatecounter.increment();
    Privatecounter.increment();
console.log(Privatecounter.getCount());
</script>
</body>
</html
```

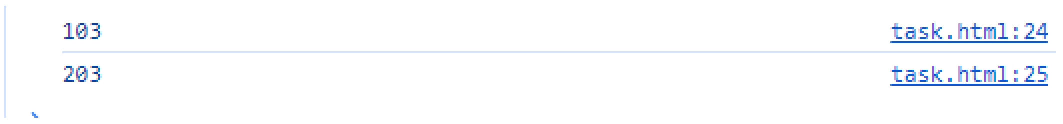
OUTPUT:

```
   2                                                    task.html:20
```

TASK 15:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function Createcounter(start_value){
    let count=start_value;
    return{
        increment: function(){
            count+=1;},
        getCount: function(){
            return count;} };}
            const counter1=Createcounter(100);
            const  counter2=Createcounter(200)
    counter1.increment();
    counter1.increment();
    counter1.increment();
    counter2.increment();
    counter2.increment();
    counter2.increment();
console.log(counter1.getCount());
console.log(counter2.getCount());
</script>
</body>
</html
```

OUTPUT:

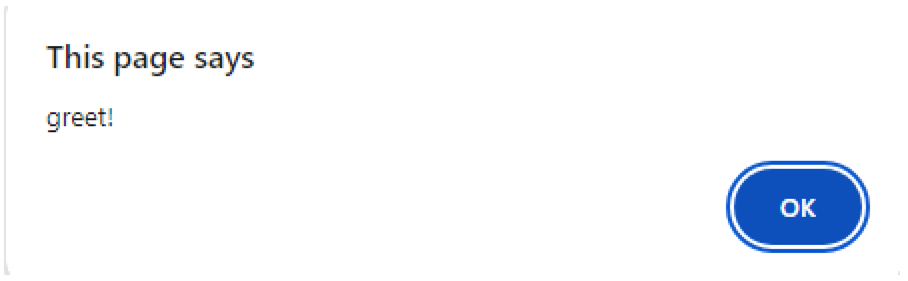


```
103                                                   task.html:24
203                                                   task.html:25
```

TASK 16:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
let promise=new Promise(function(resolve,reject){
    setTimeout(()=>resolve("greet"),3000);
});
promise.then(
    result=>alert(result),
    error=>alert(error)
);
</script>
</body>
</html
```

OUTPUT:



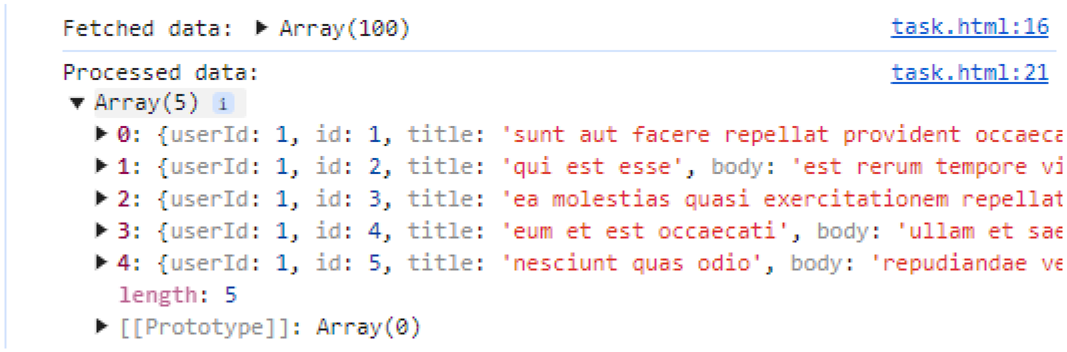
This page says

greet!

TASK 17:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
fetch('https://jsonplaceholder.typicode.com/posts')
  .then(response => {

    if (!response.ok) {
      throw new Error('Network response was not ok');
    }
    return response.json();
  })
  .then(data => {
    console.log('Fetched data:', data);

    return data.slice(0, 5);
  })
  .then(processedData => {
    console.log('Processed data:', processedData);
  })
  .catch(error => {
    console.error('Error:', error);
  });
</script>
</body>
</html
```

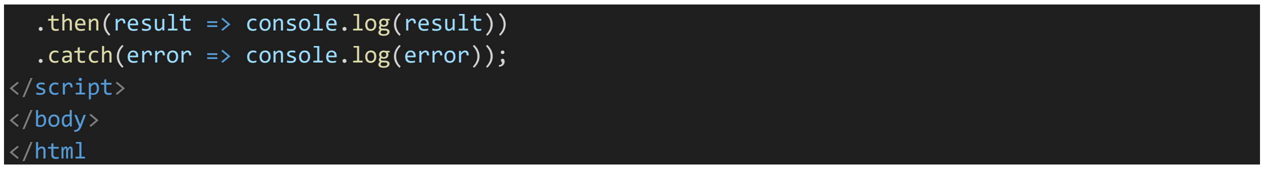
OUTPUT:



```
Fetched data:  ▶ Array(100)                                      task.html:16
Processed data:                                                  task.html:21
 ▼ Array(5) i
   ▶ 0: {userId: 1, id: 1, title: 'sunt aut facere repellat provident occaeca
   ▶ 1: {userId: 1, id: 2, title: 'qui est esse', body: 'est rerum tempore vi
   ▶ 2: {userId: 1, id: 3, title: 'ea molestias quasi exercitationem repellat
   ▶ 3: {userId: 1, id: 4, title: 'eum et est occaecati', body: 'ullam et sae
   ▶ 4: {userId: 1, id: 5, title: 'nesciunt quas odio', body: 'repudiandae ve
     length: 5
   ▶ [[Prototype]]: Array(0)
```

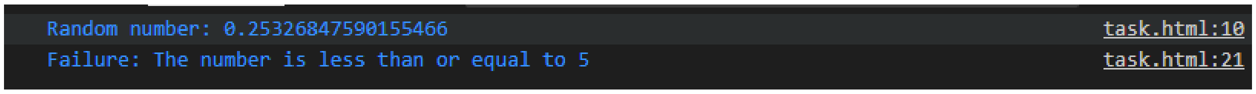
TASK 18:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function randomPromise() {
  return new Promise((resolve, reject) => {
    const randomNumber = Math.random();
    console.log('Random number:', randomNumber);

    if (randomNumber > 5) {
      resolve('Success: The number is greater than 5');
    } else {
      reject('Failure: The number is less than or equal to 5');
    }
  });
}
randomPromise()
```



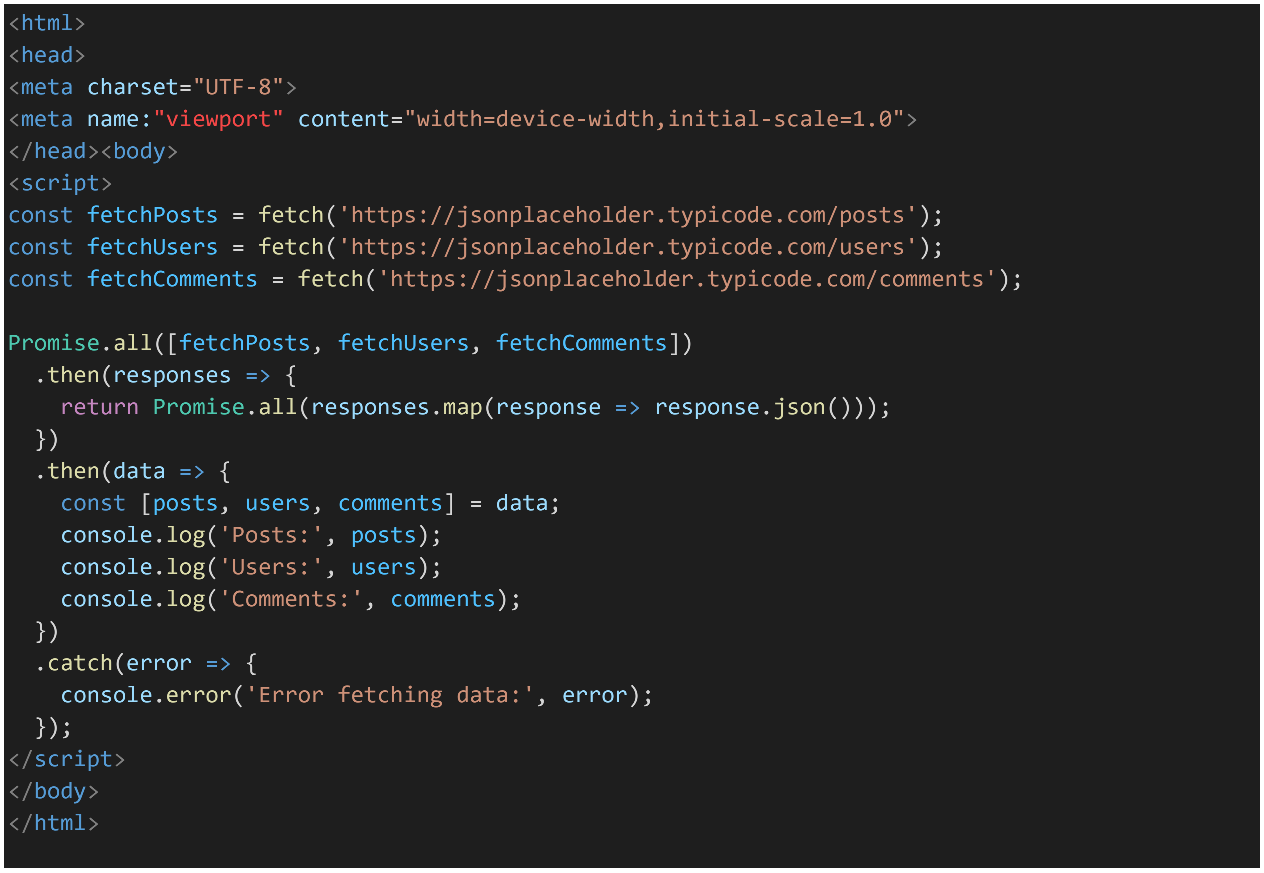
```
  .then(result => console.log(result))
  .catch(error => console.log(error));
</script>
</body>
</html
```

OUTPUT:



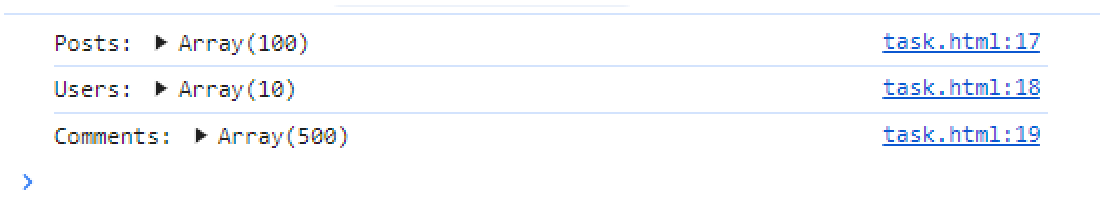
```
   Random number: 0.25326847590155466                                    task.html:10
   Failure: The number is less than or equal to 5                        task.html:21
```

TASK 19:



```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
const fetchPosts = fetch('https://jsonplaceholder.typicode.com/posts');
const fetchUsers = fetch('https://jsonplaceholder.typicode.com/users');
const fetchComments = fetch('https://jsonplaceholder.typicode.com/comments');

Promise.all([fetchPosts, fetchUsers, fetchComments])
  .then(responses => {
    return Promise.all(responses.map(response => response.json()));
  })
  .then(data => {
    const [posts, users, comments] = data;
    console.log('Posts:', posts);
    console.log('Users:', users);
    console.log('Comments:', comments);
  })
  .catch(error => {
    console.error('Error fetching data:', error);
  });
</script>
</body>
</html>
```

OUTPUT:

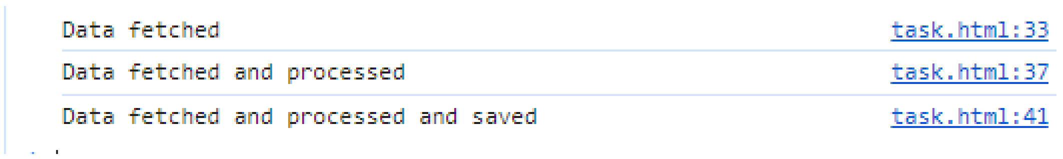


```
    Posts:  ▶ Array(100)                                    task.html:17
    Users:  ▶ Array(10)                                     task.html:18
    Comments:  ▶ Array(500)                                 task.html:19
  ›
```

TASK 20:

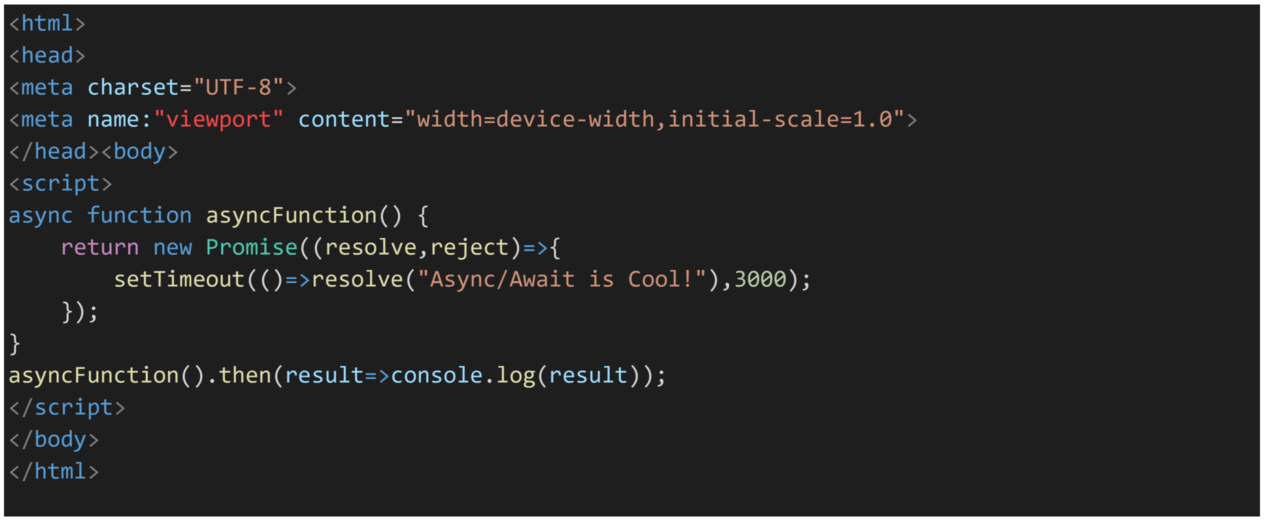
```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
function fetchData() {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve('Data fetched');
    }, 1000);
  });
}

function processData(data) {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve(`${data} and processed`);
    }, 1000);
  });
}

function saveData(data) {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve(`${data} and saved`);
    }, 1000);
  });
}

fetchData()
  .then((data) => {
    console.log(data);
    return processData(data);
  })
  .then((processedData) => {
    console.log(processedData);
    return saveData(processedData);
  })
  .then((savedData) => {
    console.log(savedData);
  })
  .catch((error) => {
    console.error('Error:', error);
  });
</script>
</body>
</html>
```

OUTPUT:



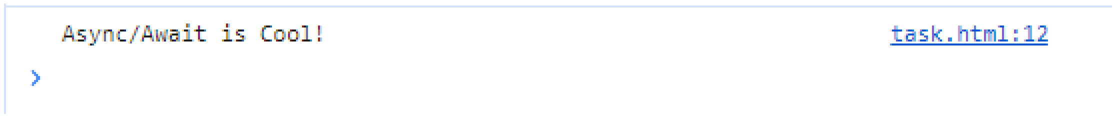
```
Data fetched                                    task.html:33
Data fetched and processed                      task.html:37
Data fetched and processed and saved            task.html:41
```

TASK 21:



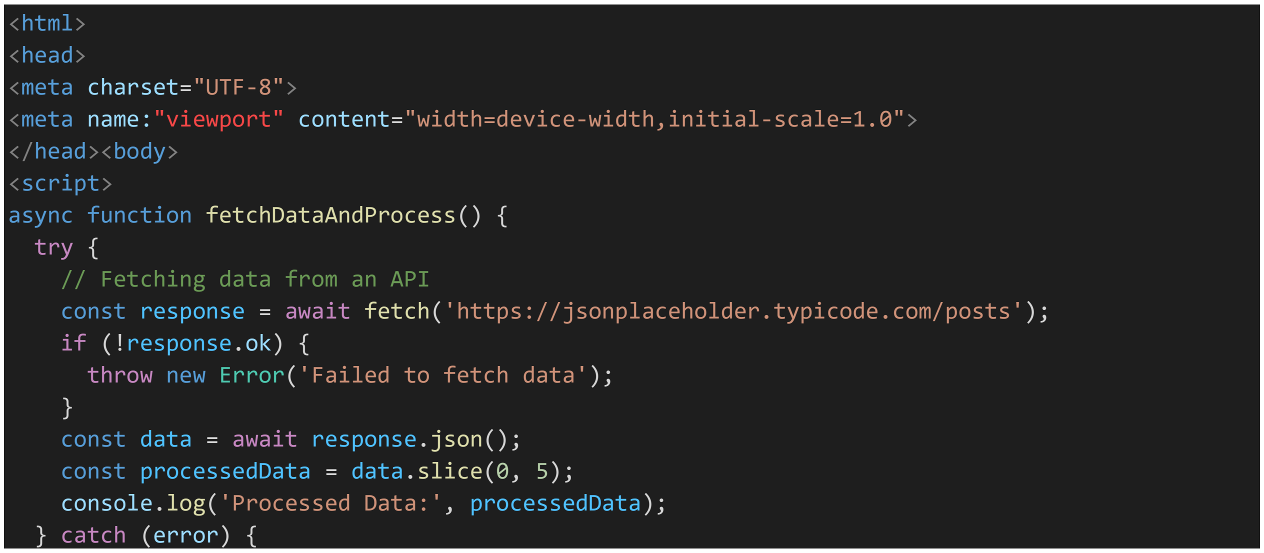
```html
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
async function asyncFunction() {
    return new Promise((resolve,reject)=>{
        setTimeout(()=>resolve("Async/Await is Cool!"),3000);
    });
}
asyncFunction().then(result=>console.log(result));
</script>
</body>
</html>
```

OUTPUT:



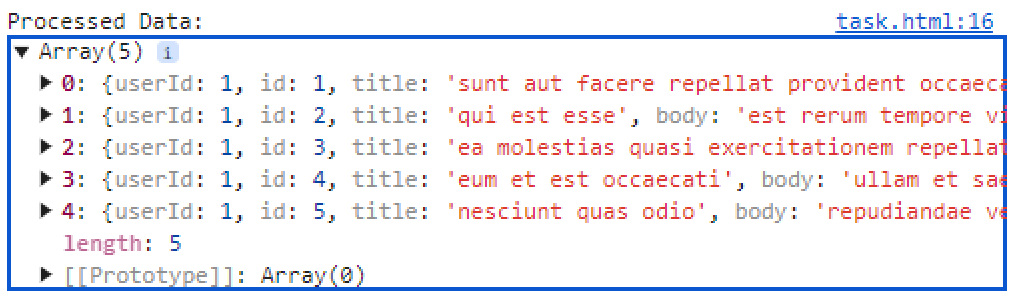
```
Async/Await is Cool!                            task.html:12
>
```

TASK 22:



```html
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
async function fetchDataAndProcess() {
  try {
    // Fetching data from an API
    const response = await fetch('https://jsonplaceholder.typicode.com/posts');
    if (!response.ok) {
      throw new Error('Failed to fetch data');
    }
    const data = await response.json();
    const processedData = data.slice(0, 5);
    console.log('Processed Data:', processedData);
  } catch (error) {
    console.error('Error:', error);
  }
}
fetchDataAndProcess();
</script>
</body>
</html>
```

OUTPUT:

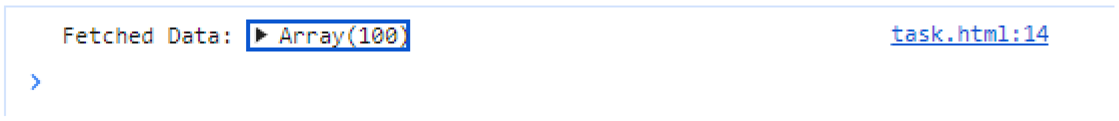


```
Processed Data:                                              task.html:16
▼ Array(5) i
  ▶ 0: {userId: 1, id: 1, title: 'sunt aut facere repellat provident occaeca
  ▶ 1: {userId: 1, id: 2, title: 'qui est esse', body: 'est rerum tempore vi
  ▶ 2: {userId: 1, id: 3, title: 'ea molestias quasi exercitationem repellat
  ▶ 3: {userId: 1, id: 4, title: 'eum et est occaecati', body: 'ullam et sae
  ▶ 4: {userId: 1, id: 5, title: 'nesciunt quas odio', body: 'repudiandae ve
    length: 5
  ▶ [[Prototype]]: Array(0)
```

TASK 23:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
async function fetchDataWithErrorHandling() {
  try {
    const response = await fetch('https://jsonplaceholder.typicode.com/posts');
    if (!response.ok) {
      throw new Error('Failed to fetch data');
    }
    const data = await response.json();
    console.log('Fetched Data:', data);
  } catch (error) {
    console.error('Error occurred:', error);
  }
}
fetchDataWithErrorHandling();
</script>
</body>
</html>
```

OUTPUT:

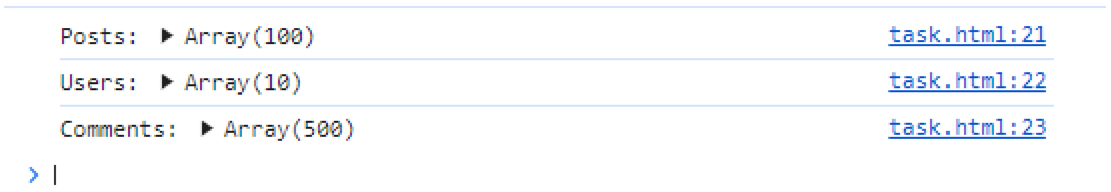


```
  Fetched Data: ▶ Array(100)                    task.html:14
›
```

TASK 24:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
async function fetchMultipleResources() {
  try {
    const [postsResponse, usersResponse, commentsResponse] = await Promise.all([
      fetch('https://jsonplaceholder.typicode.com/posts'),
      fetch('https://jsonplaceholder.typicode.com/users'),
      fetch('https://jsonplaceholder.typicode.com/comments')
    ]);
    if (!postsResponse.ok || !usersResponse.ok || !commentsResponse.ok) {
      throw new Error('One or more requests failed');
    }
    const posts = await postsResponse.json();
    const users = await usersResponse.json();
    const comments = await commentsResponse.json();

    console.log('Posts:', posts);
    console.log('Users:', users);
    console.log('Comments:', comments);
  } catch (error) {
    console.error('Error occurred:', error);
  }
}
fetchMultipleResources();
</script>
</body>
</html>
```

OUTPUT:



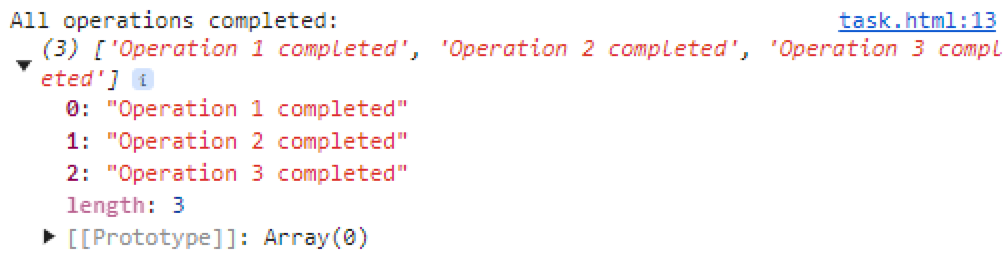
```
Posts: ▶ Array(100)                          task.html:21
Users: ▶ Array(10)                           task.html:22
Comments: ▶ Array(500)                       task.html:23
›
```

TASK 25:

```
<html>
<head>
<meta charset="UTF-8">
<meta name:"viewport" content="width=device-width,initial-scale=1.0">
</head><body>
<script>
```

```
async function waitForMultipleOperations() {
  try {
    const asyncOp1 = new Promise((resolve) => setTimeout(() => resolve('Operation 1
completed'), 1000));
    const asyncOp2 = new Promise((resolve) => setTimeout(() => resolve('Operation 2
completed'), 2000));
    const asyncOp3 = new Promise((resolve) => setTimeout(() => resolve('Operation 3
completed'), 1500));
    const results = await Promise.all([asyncOp1, asyncOp2, asyncOp3]);
    console.log('All operations completed:', results);
  } catch (error) {
    console.error('Error:', error);
  }
}
waitForMultipleOperations();
</script>
</body>
</html>
```

OUTPUT:



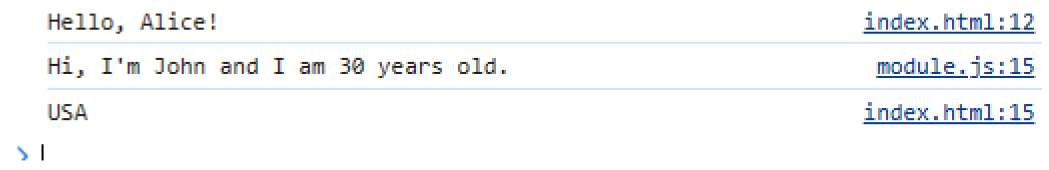
```
All operations completed:                                         task.html:13
▼ (3) ['Operation 1 completed', 'Operation 2 completed', 'Operation 3 compl
  eted'] i
    0: "Operation 1 completed"
    1: "Operation 2 completed"
    2: "Operation 3 completed"
    length: 3
  ▶ [[Prototype]]: Array(0)
```

TASK 26:

```
export function greet(name) {
    return `Hello, ${name}!`;
  }
  export class Person {
    constructor(name, age) {
      this.name = name;
      this.age = age;
    }
    sayHello() {
      console.log(`Hi, I'm ${this.name} and I am ${this.age} years old.`);
    }
  }

  export const country = "USA";
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ES Modules Example</title>
</head>
<body>
    <script type="module">
        import { greet, Person, country } from './module.js';

        console.log(greet("Alice"));
        const john = new Person("John", 30);
        john.sayHello();
        console.log(country);
    </script>
</body>
</html>
```

OUTPUT:



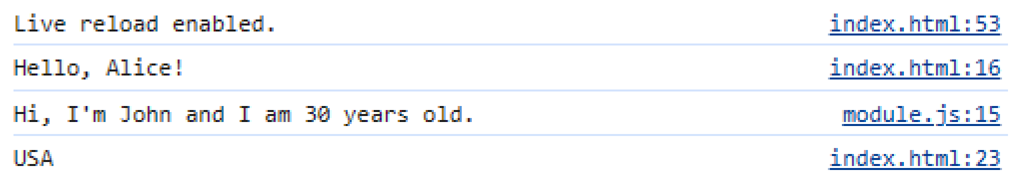
```
Hello, Alice!                                        index.html:12
Hi, I'm John and I am 30 years old.                   module.js:15
USA                                                  index.html:15
```

TASK 27:

```javascript
export function greet(name) {
    return `Hello, ${name}!`;
  }
  export class Person {
    constructor(name, age) {
      this.name = name;
      this.age = age;
    }
    sayHello() {
      console.log(`Hi, I'm ${this.name} and I am ${this.age} years old.`);
    }
  }
  export const country = "USA";
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ES Modules Example</title>
</head>
<body>
    <h1>Check the console for output</h1>

    <script type="module">
        // Importing the function, class, and variable from module.js
        import { greet, Person, country } from './module.js';

        // Using the imported function
        console.log(greet("Alice"));  // Output: Hello, Alice!

        // Using the imported class
        const john = new Person("John", 30);
        john.sayHello();  // Output: Hi, I'm John and I am 30 years old.

        // Using the imported variable
        console.log(country);  // Output: USA
    </script>
</body>
</html>
```

OUTPUT:



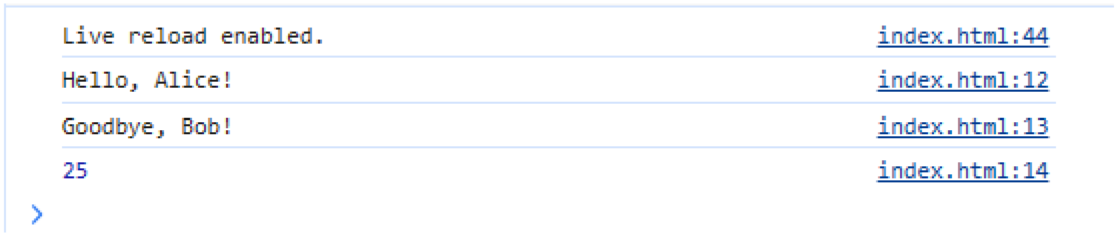
```
Live reload enabled.                                     index.html:53
Hello, Alice!                                            index.html:16
Hi, I'm John and I am 30 years old.                        module.js:15
USA                                                      index.html:23
```

TASK 28:

```
export function greet(name) {
    return `Hello, ${name}!`;
  }
  export function farewell(name) {
    return `Goodbye, ${name}!`;
  }
  export function square(number) {
    return number * number;
  }
```

```
!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Named Exports Example</title>
</head>
<body>
    <h1>Check the console for output</h1>
    <script type="module">
        import { greet, farewell, square } from './module.js';
         console.log(greet("Alice"));
        console.log(farewell("Bob"));
        console.log(square(5));
    </script>
</body>
</html>
```

OUTPUT:



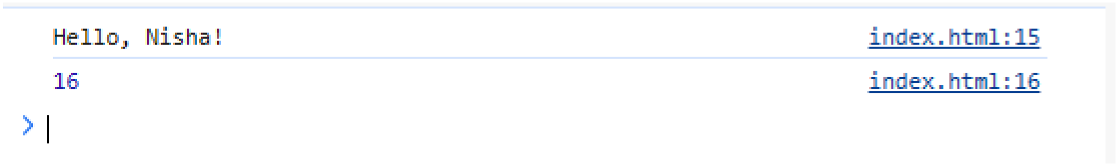
```
Live reload enabled.                                   index.html:44
Hello, Alice!                                          index.html:12
Goodbye, Bob!                                          index.html:13
25                                                     index.html:14
>
```

TASK 29:

```
export function greet(name) {
    return `Hello, ${name}!`;
  }
  export function farewell(name) {
    return `Goodbye, ${name}!`;
  }
  export function square(number) {
    return number * number;}
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Named Imports Example</title>
</head>
<body>
    <h1>Check the console for output</h1>

    <script type="module">

        import { greet, square } from './module.js';

        console.log(greet("Nisha"));
        console.log(square(4));
    </script>
</body>
</html>
```

OUTPUT:

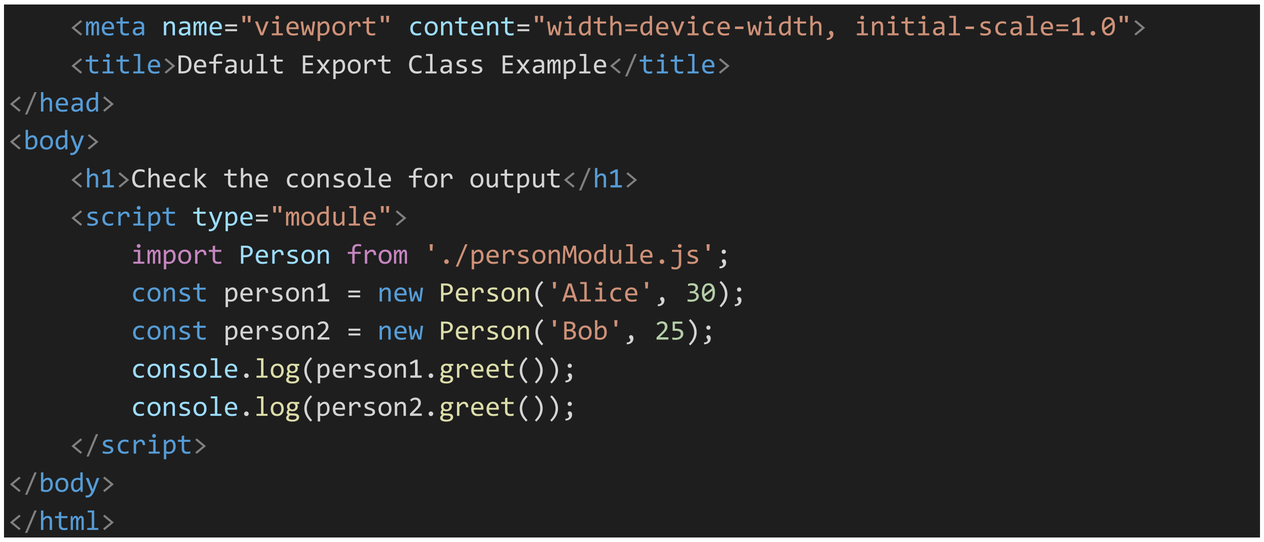


```
Hello, Nisha!                                              index.html:15
16                                                         index.html:16
>
```

TASK 30:

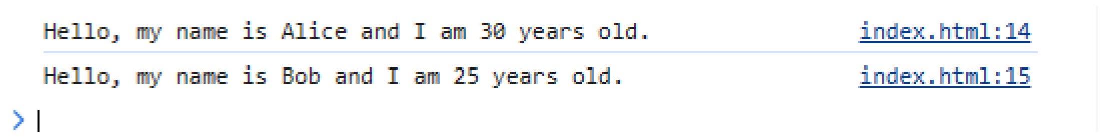
```
export default class Person {
    constructor(name, age) {
      this.name = name;
      this.age = age;
    }
    greet() {
      return `Hello, my name is ${this.name} and I am ${this.age} years old.`;
    }
  }
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```



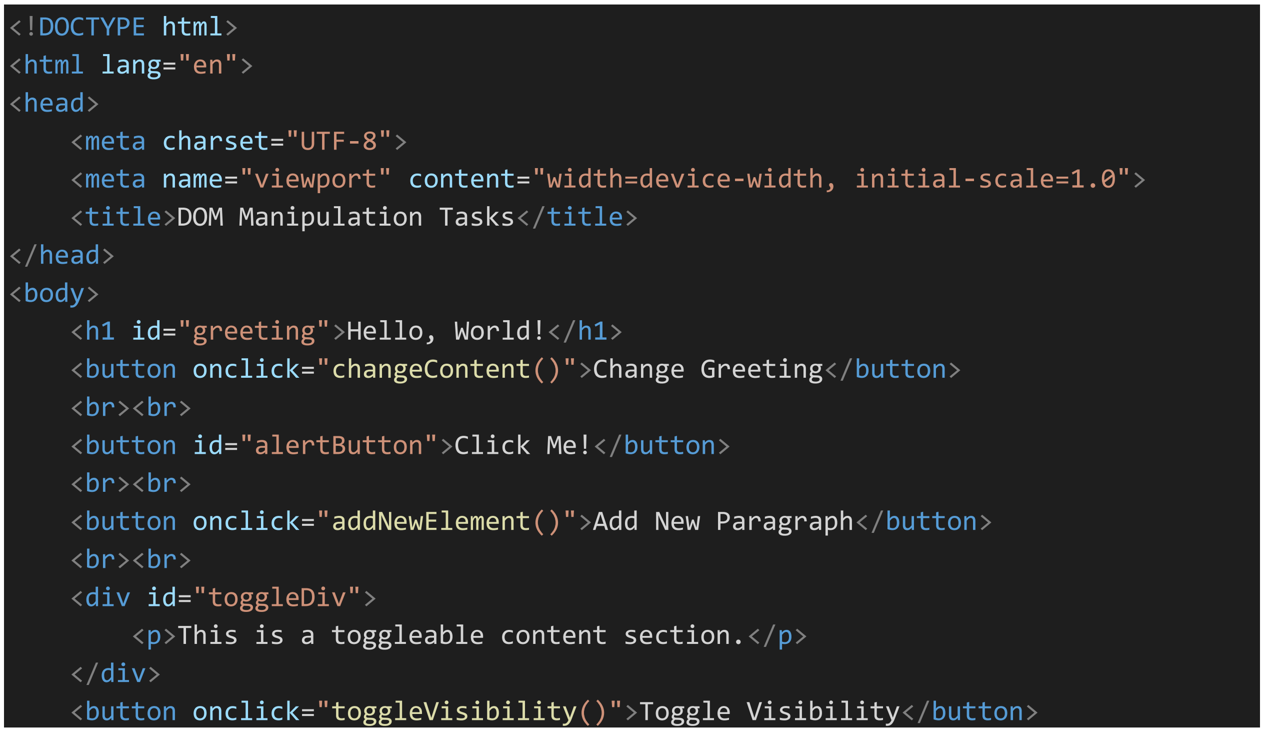
```html
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Default Export Class Example</title>
</head>
<body>
    <h1>Check the console for output</h1>
    <script type="module">
        import Person from './personModule.js';
        const person1 = new Person('Alice', 30);
        const person2 = new Person('Bob', 25);
        console.log(person1.greet());
        console.log(person2.greet());
    </script>
</body>
</html>
```

OUTPUT:



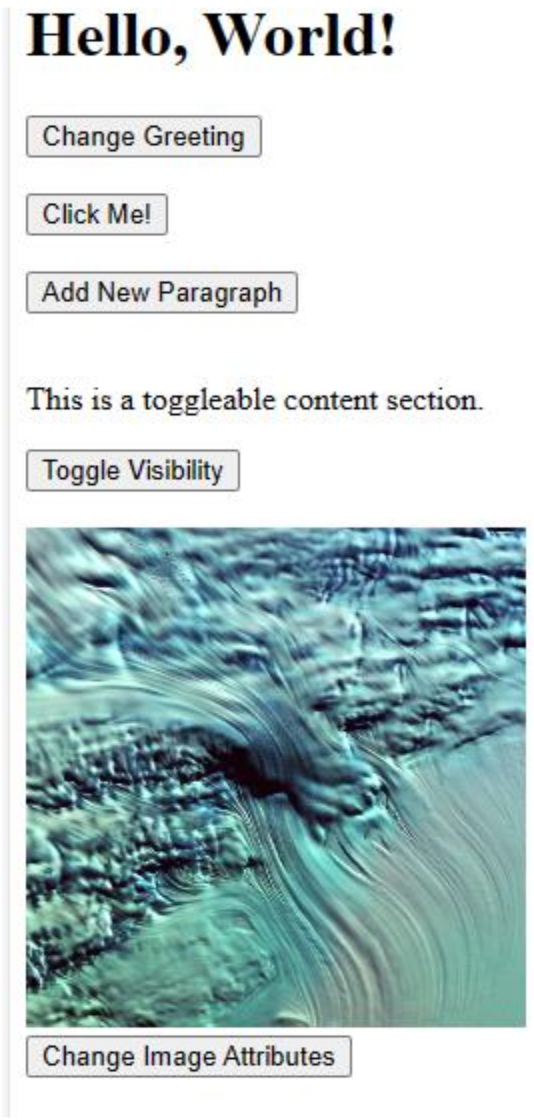
```
Hello, my name is Alice and I am 30 years old.          index.html:14
Hello, my name is Bob and I am 25 years old.            index.html:15
> |
```

TASK 31:



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DOM Manipulation Tasks</title>
</head>
<body>
    <h1 id="greeting">Hello, World!</h1>
    <button onclick="changeContent()">Change Greeting</button>
    <br><br>
    <button id="alertButton">Click Me!</button>
    <br><br>
    <button onclick="addNewElement()">Add New Paragraph</button>
    <br><br>
    <div id="toggleDiv">
        <p>This is a toggleable content section.</p>
    </div>
    <button onclick="toggleVisibility()">Toggle Visibility</button>
    <br><br>
    <img id="myImage" src="usgs-eAGoXRFiysw-unsplash.jpg" alt="Placeholder Image"
width="250p">
    <br>
    <button onclick="changeImageAttributes()">Change Image Attributes</button>
    <script>
        function changeContent() {
            const greeting = document.getElementById("greeting");
            greeting.textContent = "Hello, JavaScript!";
        }
        const alertButton = document.getElementById("alertButton");
        alertButton.addEventListener("click", function() {
            alert("Button was clicked!");
        });
        function addNewElement() {
            const newParagraph = document.createElement("p");
            newParagraph.textContent = "This is a newly added paragraph!";
            document.body.appendChild(newParagraph);
        }
        function toggleVisibility() {
            const div = document.getElementById("toggleDiv");
            if (div.style.display === "none") {
                div.style.display = "block";
            } else {
                div.style.display = "none";
            }
        }
        function changeImageAttributes() {
            const image = document.getElementById("myImage");
            image.src = "usgs-eAGoXRFiysw-unsplash.jpg";
            image.alt = "Updated Image";
        }
    </script>
</body>
</html>
```

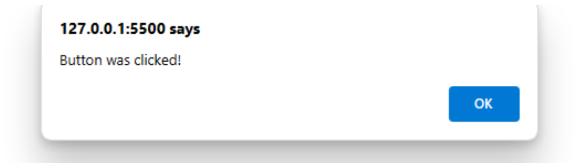
OUTPUT:



TASK 32:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Event Listener Example</title>
</head>
<body>
    <button id="actionButton">Click Me!</button>
    <script>
        const button = document.getElementById("actionButton");
        button.addEventListener("click", function() {
            alert("Button was clicked!");
        });
    </script>
</body>
</html>
```

OUTPUT:

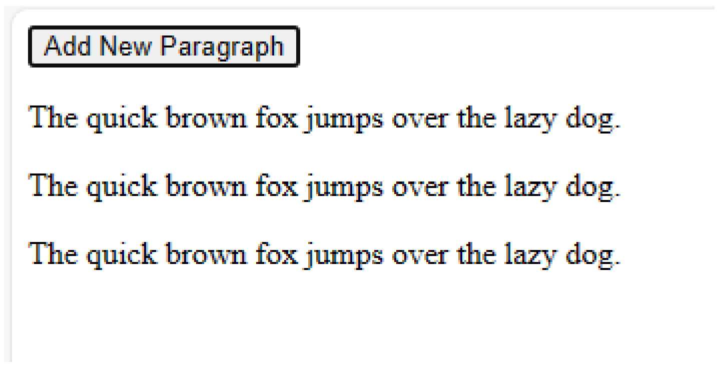

127.0.0.1:5500 says

Button was clicked!

TASK 33:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Create and Append New Element</title>
</head>
<body>
    <button id="addElementButton">Add New Paragraph</button>
    <script>
        function addNewElement() {
            const newParagraph = document.createElement("p");
            newParagraph.textContent =
    "The quick brown fox jumps over the lazy dog. ";
            document.body.appendChild(newParagraph);
        }
        const button = document.getElementById("addElementButton");
        button.addEventListener("click", addNewElement);
    </script>
</body>
</html>
```

OUTPUT:



The quick brown fox jumps over the lazy dog.

The quick brown fox jumps over the lazy dog.

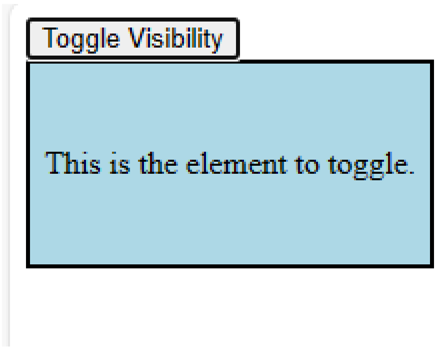
The quick brown fox jumps over the lazy dog.

TASK 34:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Toggle Visibility</title>
    <style>
        #myElement {
            width: 200px;
            height: 100px;
            background-color: lightblue;
            text-align: center;
            line-height: 100px;
            border: 2px solid #000;
        } </style>
</head>
<body>
    <button onclick="toggleVisibility()">Toggle Visibility</button>
    <div id="myElement">
        This is the element to toggle.
    </div>
    <script>
        function toggleVisibility() {
            var element = document.getElementById("myElement");
            if (element.style.display === "none") {
                element.style.display = "block";
            } else {
                element.style.display = "none";    }
        }
    </script>
</body>
```

```
</html>
```

OUTPUT:

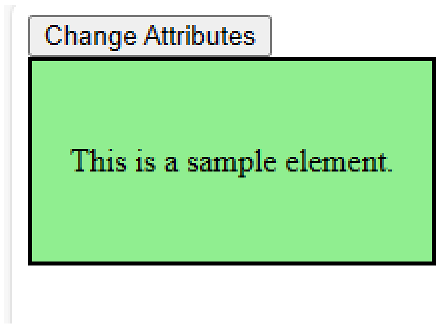


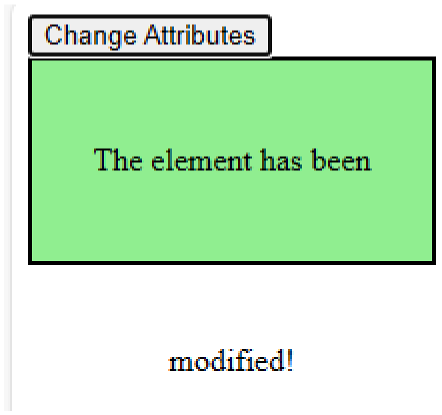
This is the element to toggle.

TASK 35:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modify Element Attributes</title>
    <style>
        #myElement {
            width: 200px;
            height: 100px;
            background-color: lightgreen;
            text-align: center;
            line-height: 100px;
            border: 2px solid #000;
        }
    </style>
</head>
<body>
    <button onclick="changeAttributes()">Change Attributes</button>
    <div id="myElement" class="box" title="Original Title">
        This is a sample element.
    </div>
    <script>
        function changeAttributes() {
            var element = document.getElementById("myElement");
            var currentClass = element.getAttribute("class");
            var currentTitle = element.getAttribute("title");
            console.log("Current class:", currentClass);
            console.log("Current title:", currentTitle);
            element.setAttribute("class", "modified-box");
            element.setAttribute("title", "Modified Title");
            element.textContent = "The element has been modified!";
            console.log("New class:", element.getAttribute("class"));
            console.log("New title:", element.getAttribute("title"));
        }
    </script>
</body>
</html>
```

OUTPUT:


Change Attributes
This is a sample element.


Change Attributes
The element has been
modified!