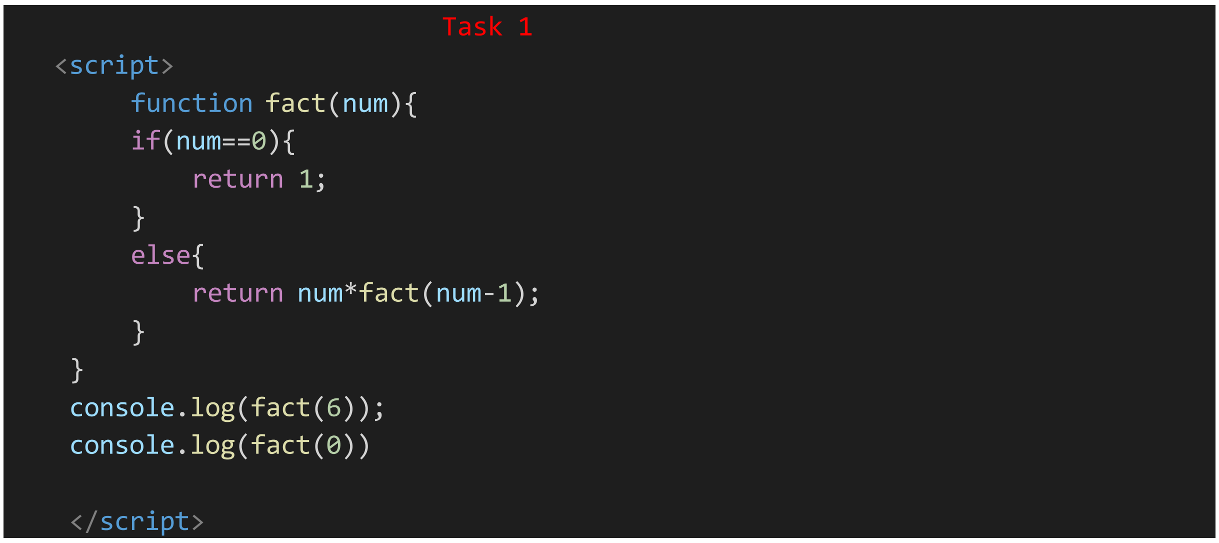
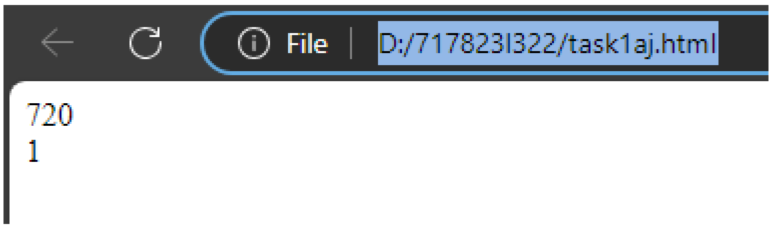
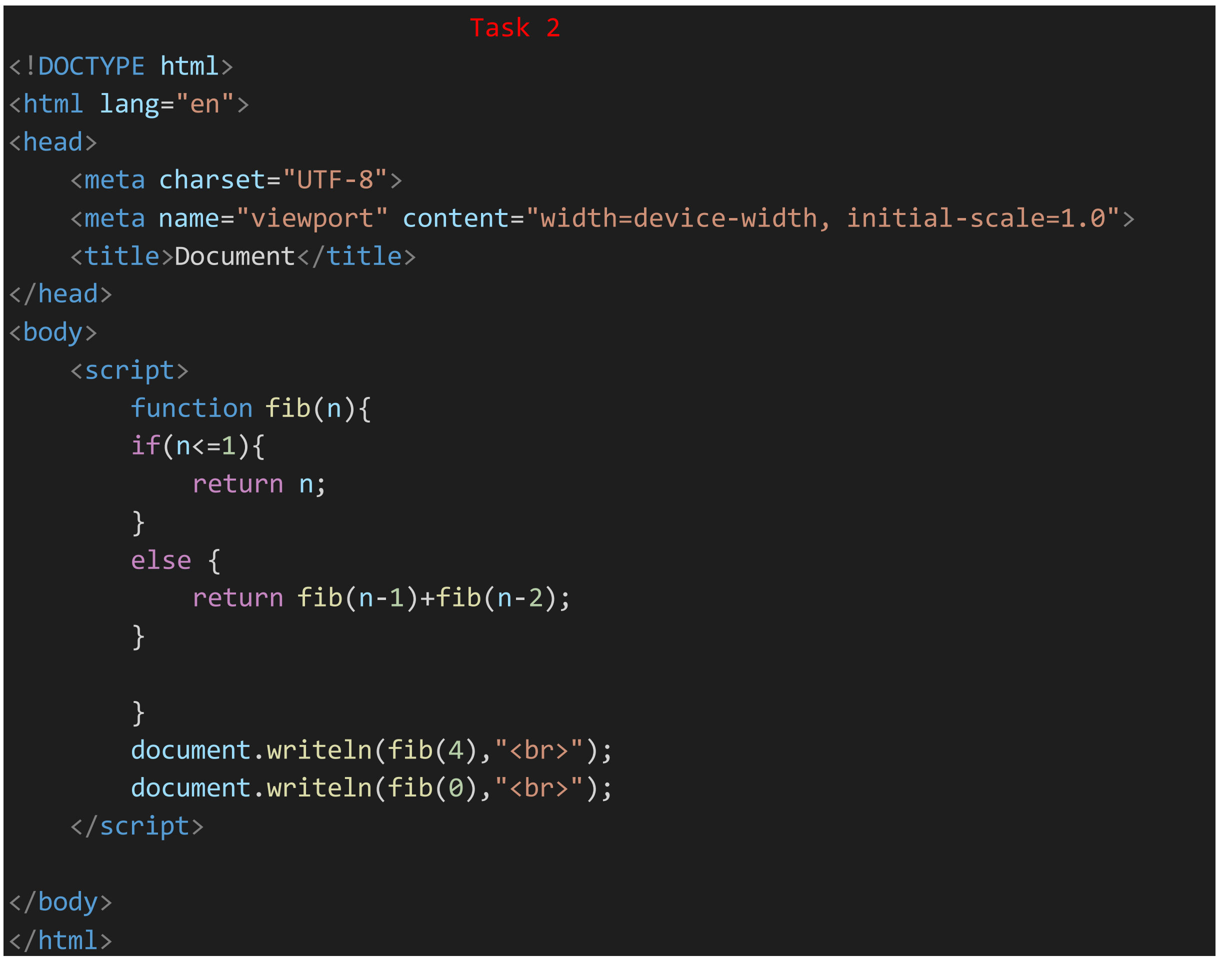


Task 1

```
<script>
     function fact(num){
     if(num==0){
         return 1;
     }
     else{
         return num*fact(num-1);
     }
 }
 console.log(fact(6));
 console.log(fact(0))

 </script>
```

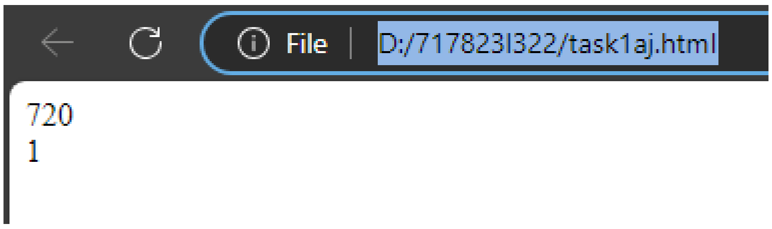


```
720
1
```

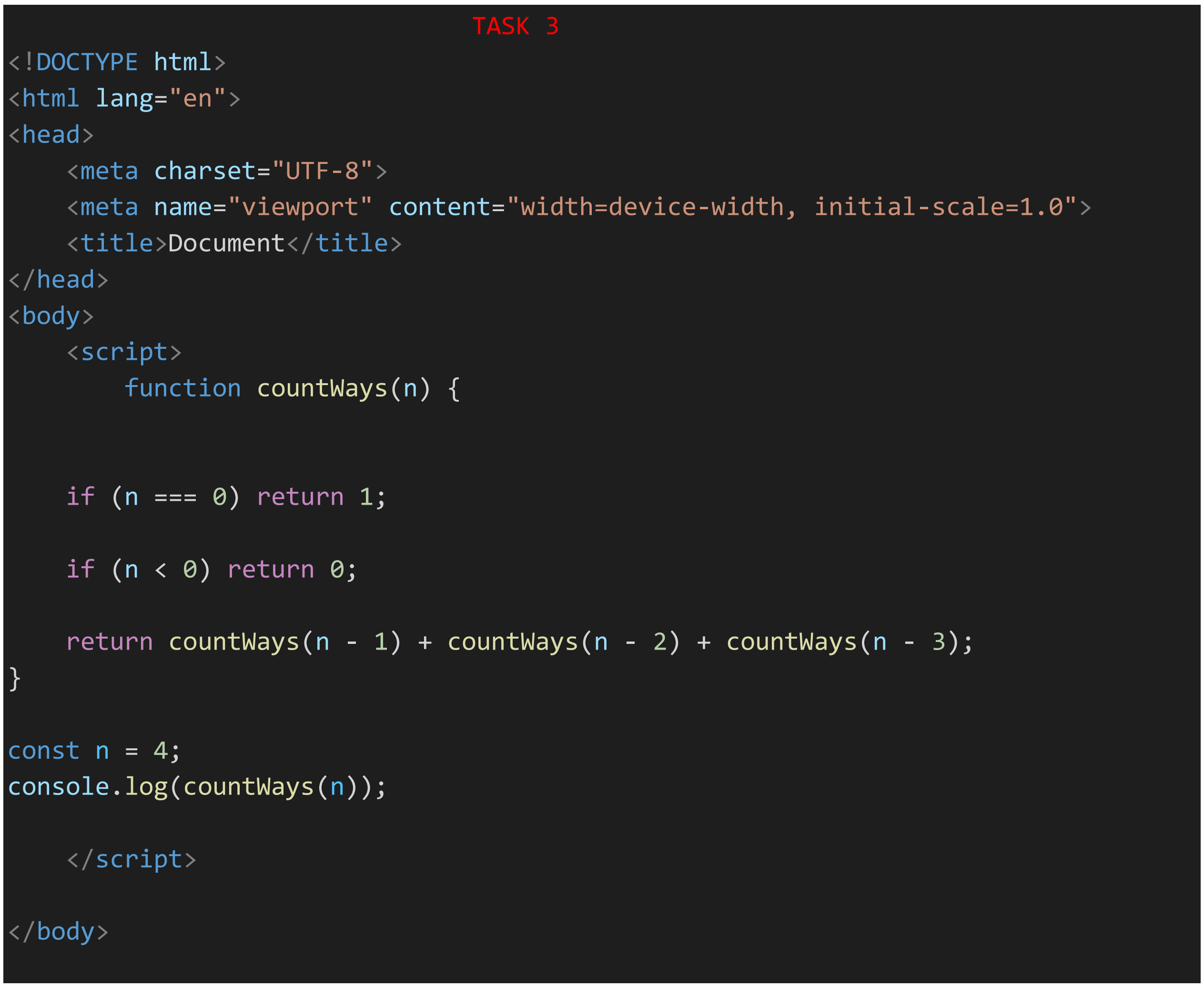


Task 2

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

        }
        document.writeln(fib(4),"<br>");
        document.writeln(fib(0),"<br>");
    </script>

</body>
</html>
```

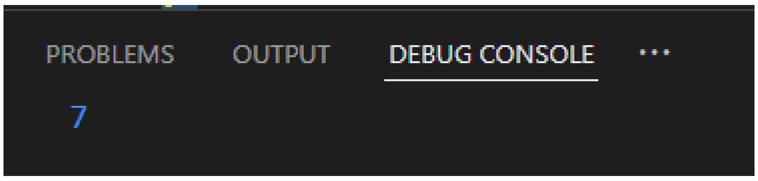


```
720
1
```

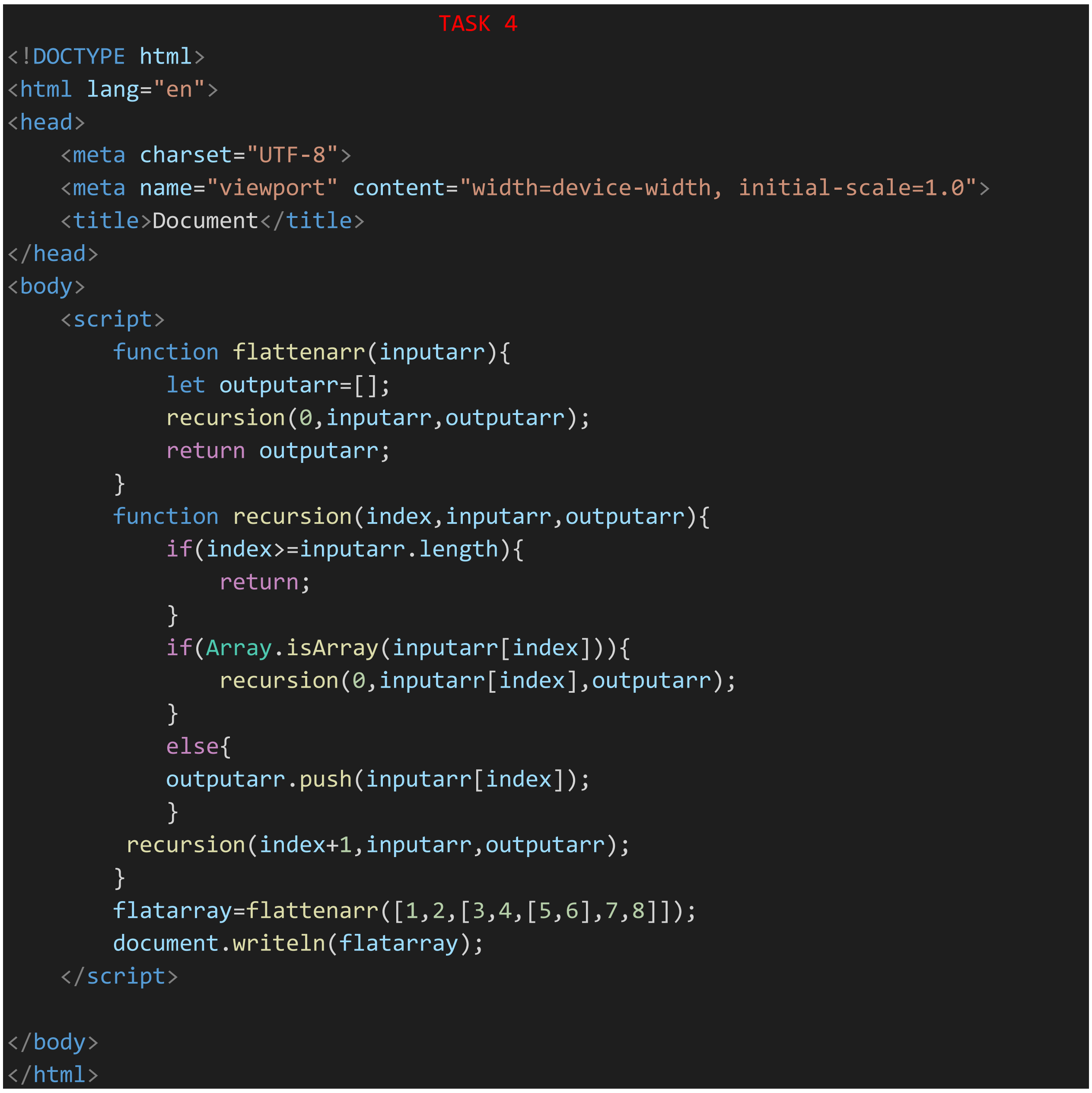


TASK 3

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function countWays(n) {


    if (n === 0) return 1;

    if (n < 0) return 0;

    return countWays(n - 1) + countWays(n - 2) + countWays(n - 3);
}

const n = 4;
console.log(countWays(n));

    </script>

</body>
```

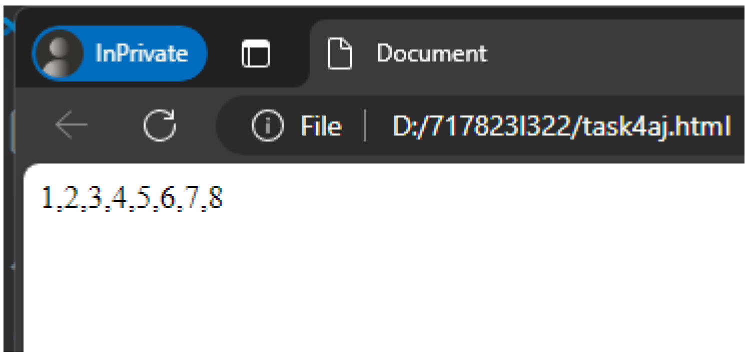


```
7
```



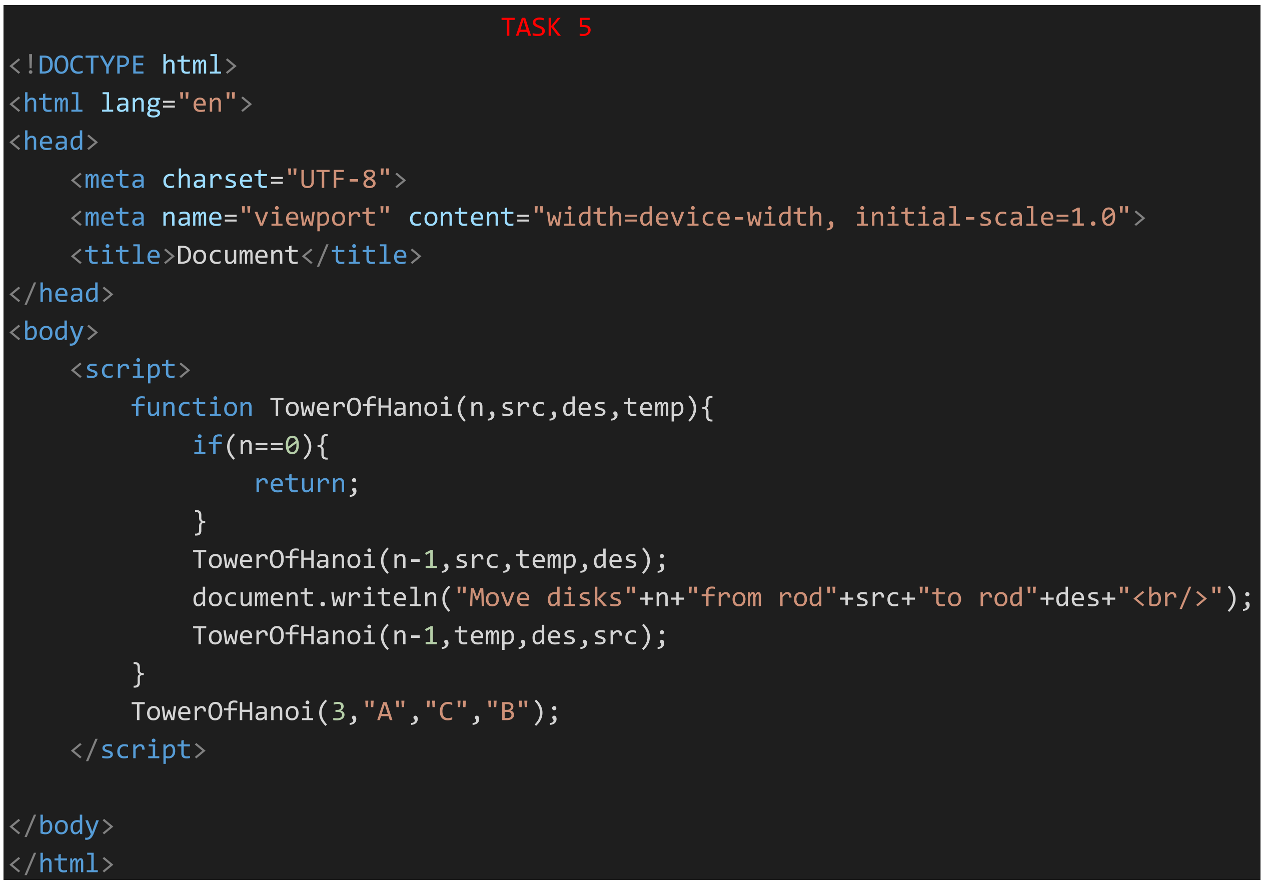
TASK 4

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function flattenarr(inputarr){
            let outputarr=[];
            recursion(0,inputarr,outputarr);
            return outputarr;
        }
        function recursion(index,inputarr,outputarr){
            if(index>=inputarr.length){
                return;
            }
            if(Array.isArray(inputarr[index])){
                recursion(0,inputarr[index],outputarr);
            }
            else{
            outputarr.push(inputarr[index]);
            }
         recursion(index+1,inputarr,outputarr);
        }
        flatarray=flattenarr([1,2,[3,4,[5,6],7,8]]);
        document.writeln(flatarray);
    </script>

</body>
</html>
```



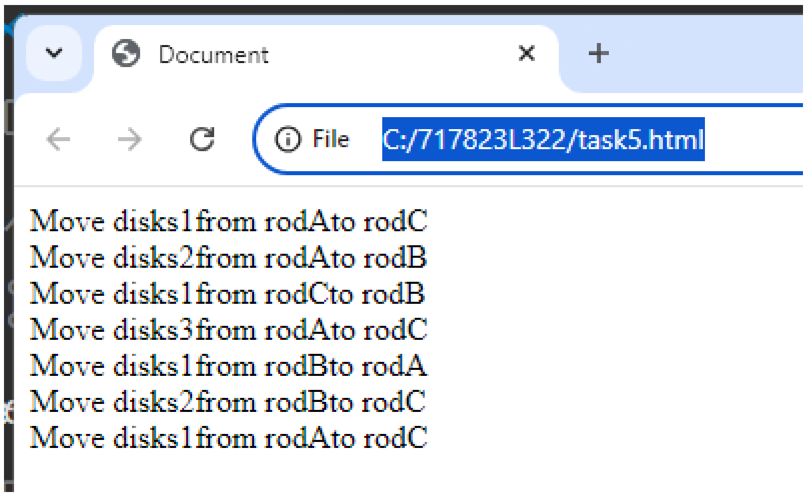
D:/717823l322/task4aj.html

```
1,2,3,4,5,6,7,8
```



TASK 5

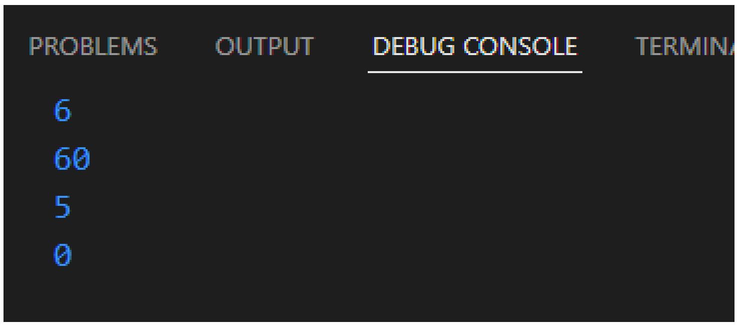
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function TowerOfHanoi(n,src,des,temp){
            if(n==0){
                return;
            }
            TowerOfHanoi(n-1,src,temp,des);
            document.writeln("Move disks"+n+"from rod"+src+"to rod"+des+"<br/>");
            TowerOfHanoi(n-1,temp,des,src);
        }
        TowerOfHanoi(3,"A","C","B");
    </script>

</body>
</html>
```



```
Move disks1from rodAto rodC
Move disks2from rodAto rodB
Move disks1from rodCto rodB
Move disks3from rodAto rodC
Move disks1from rodBto rodA
Move disks2from rodBto rodC
Move disks1from rodAto rodC
```

TASK 6

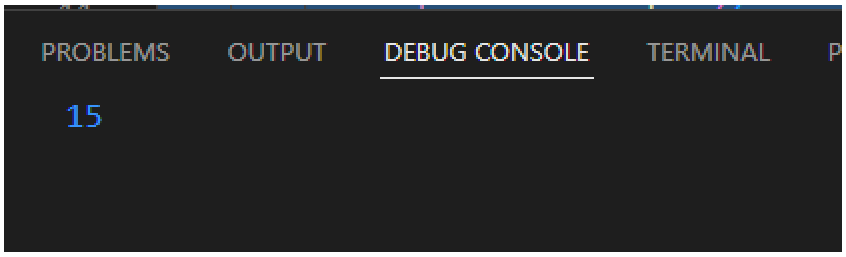
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
    function sum(...args) {
        return args.reduce((accum, num) => accum+ num, 0);
      }

      console.log(sum(1, 2, 3));
      console.log(sum(10, 50));
      console.log(sum(5));
      console.log(sum());

     </script>

</body>
</html>
```



```
6
60
5
0
```

TASK 7

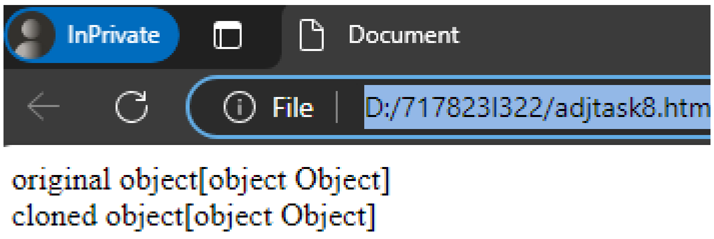
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function sum(...args) {
            return args.reduce((acc, num) => acc + num, 0);
          }
          const arr=[1,2,3,4,5];
          console.log(sum(...arr));
        </script>
</body>
</html>
```



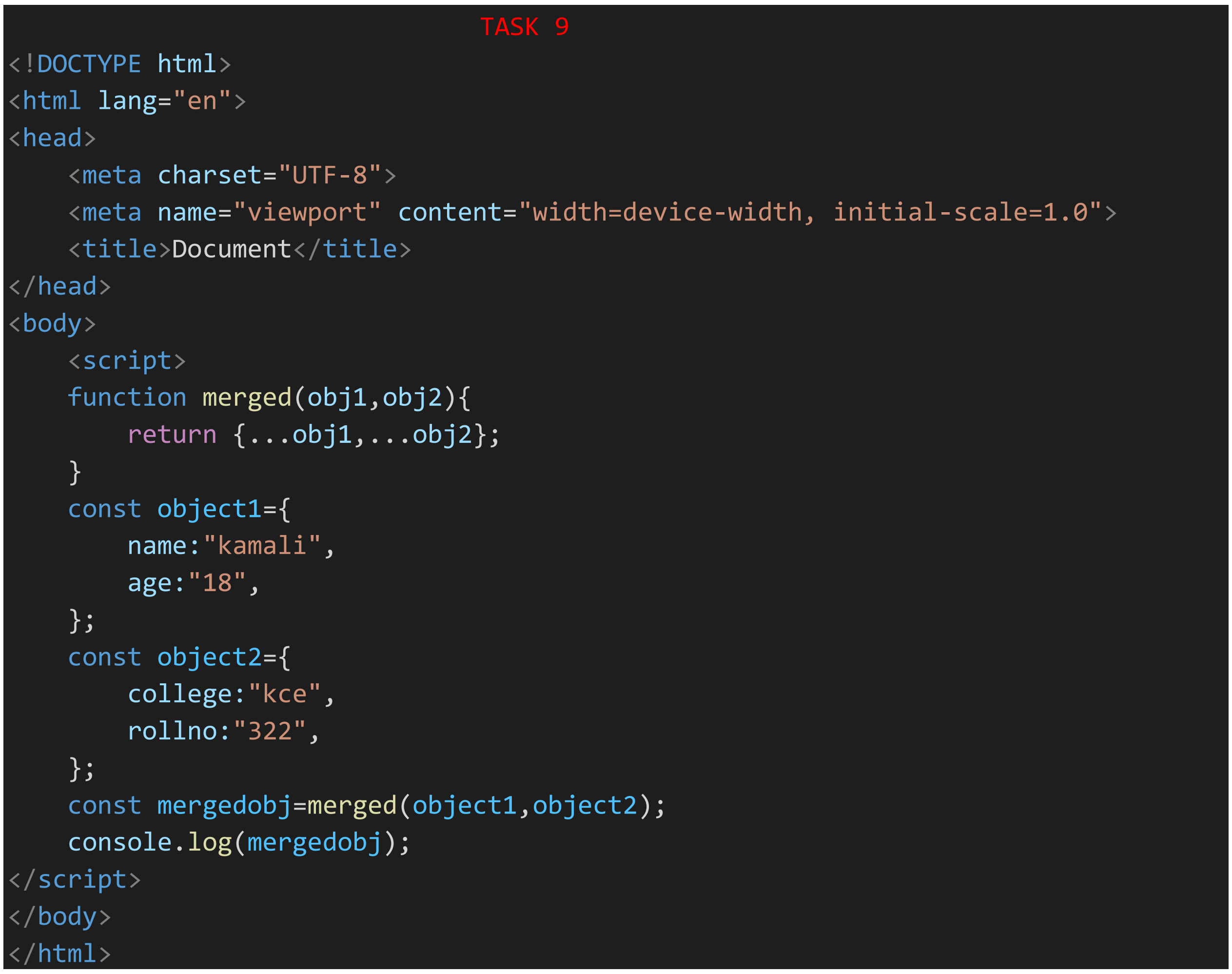
```
15
```

TASK 8

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
    function deepclone(student){
        return JSON.parse(JSON.stringify(student));
    }
    const student={
        name:"kamali",
        college:"kce",
        address:{
            place:"coimbatore",
            pincode:"641032"

        },
        department:"ECE"
    };
     const cloned=deepclone(student);
     cloned.name="Indhirani";
     cloned.college="IIT madras";
     console.log("original object",student);
     console.log("cloned object",cloned);
     </script>
</body>
</html>
```

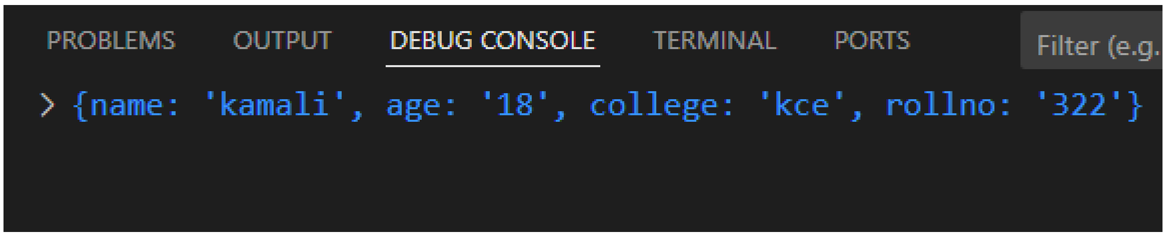


```
original object[object Object]
cloned object[object Object]
```

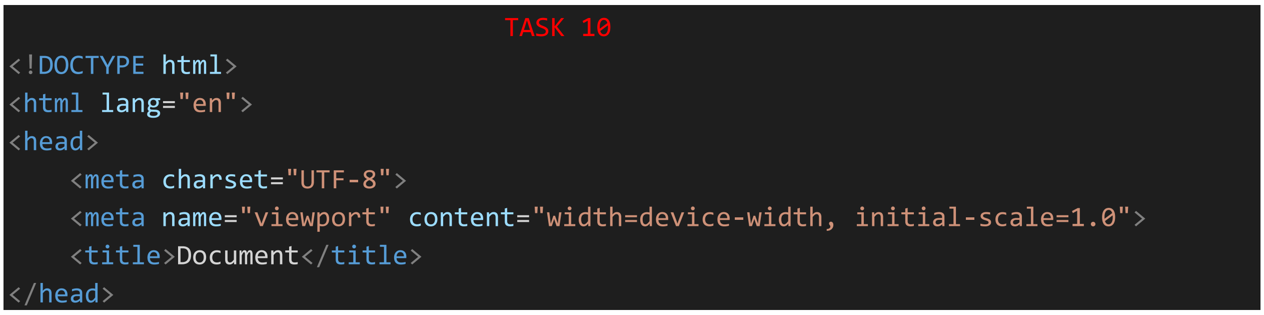


TASK 9

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
    function merged(obj1,obj2){
        return {...obj1,...obj2};
    }
    const object1={
        name:"kamali",
        age:"18",
    };
    const object2={
        college:"kce",
        rollno:"322",
    };
    const mergedobj=merged(object1,object2);
    console.log(mergedobj);
</script>
</body>
</html>
```

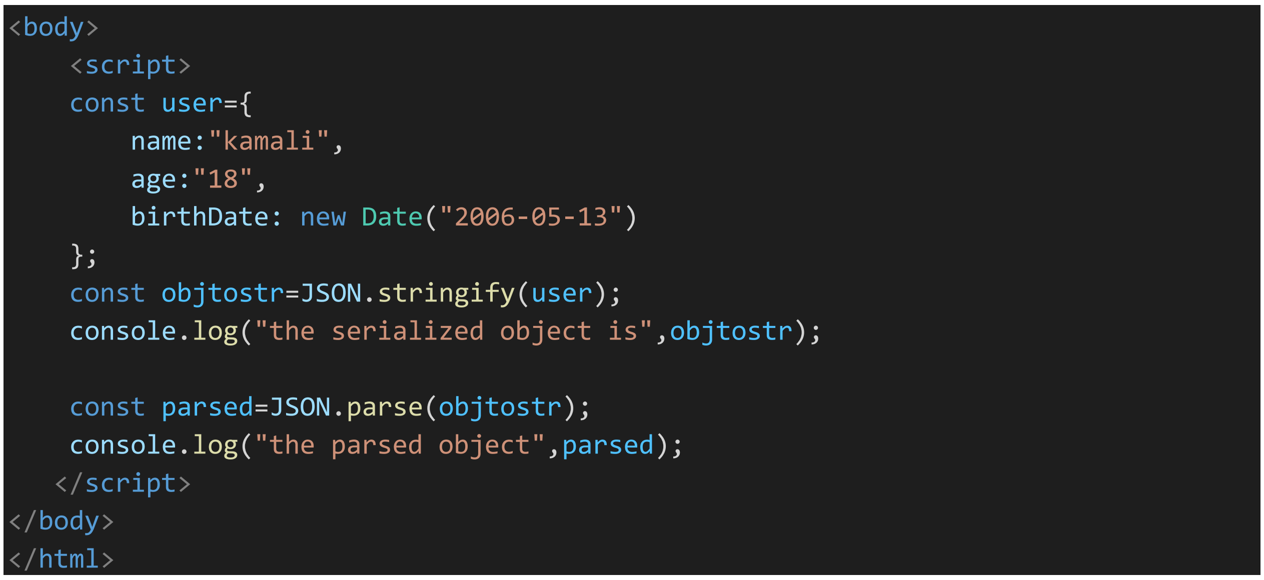


```
> {name: 'kamali', age: '18', college: 'kce', rollno: '322'}
```

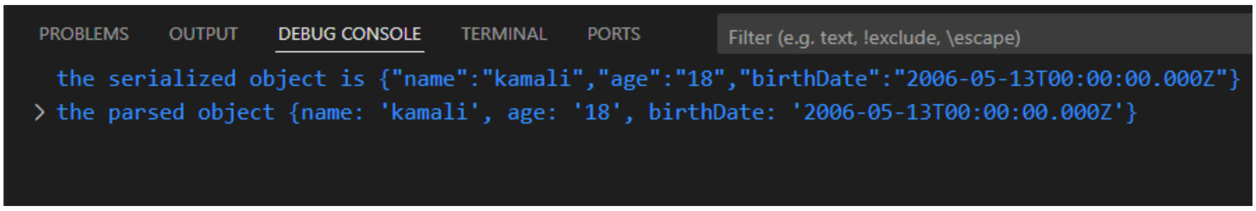


TASK 10

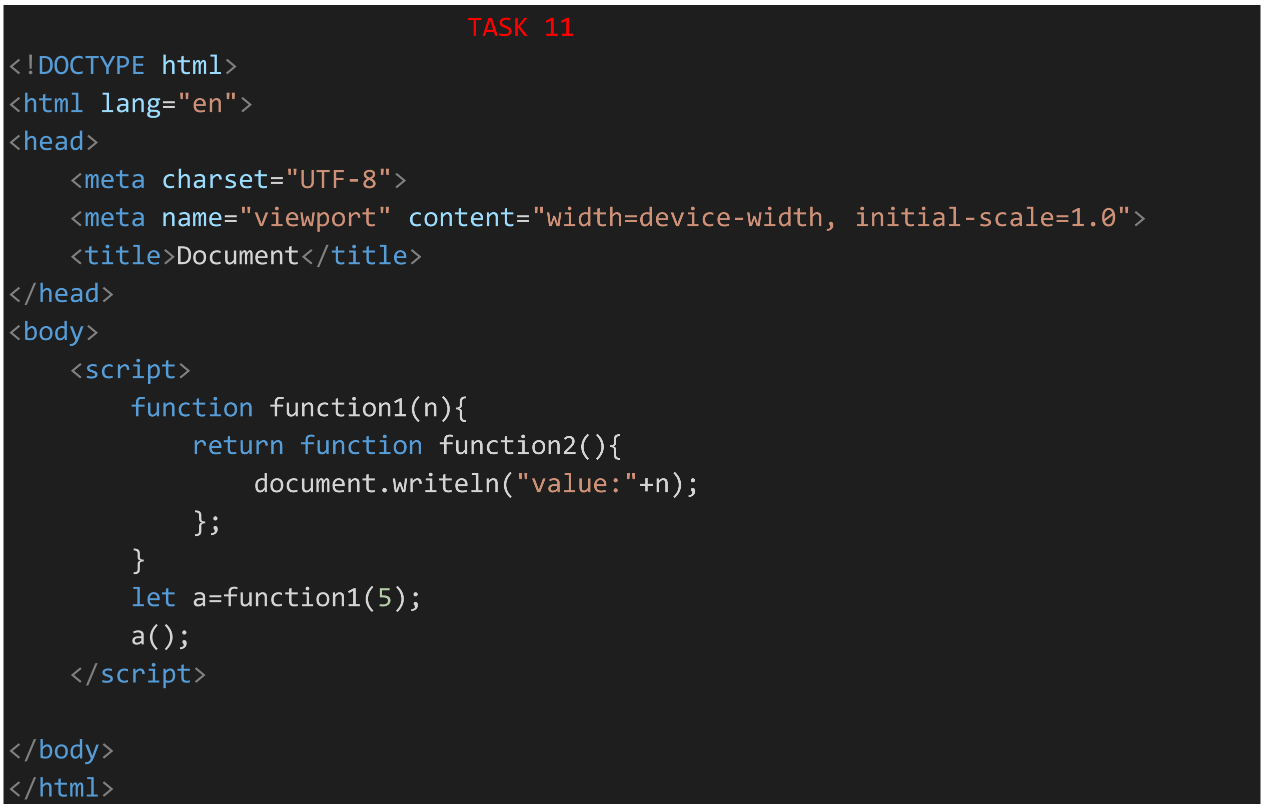
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
```



```
<body>
    <script>
    const user={
        name:"kamali",
        age:"18",
        birthDate: new Date("2006-05-13")
    };
    const objtostr=JSON.stringify(user);
    console.log("the serialized object is",objtostr);

    const parsed=JSON.parse(objtostr);
    console.log("the parsed object",parsed);
   </script>
</body>
</html>
```

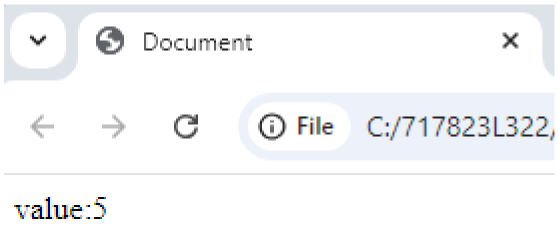


```
  the serialized object is {"name":"kamali","age":"18","birthDate":"2006-05-13T00:00:00.000Z"}
> the parsed object {name: 'kamali', age: '18', birthDate: '2006-05-13T00:00:00.000Z'}
```

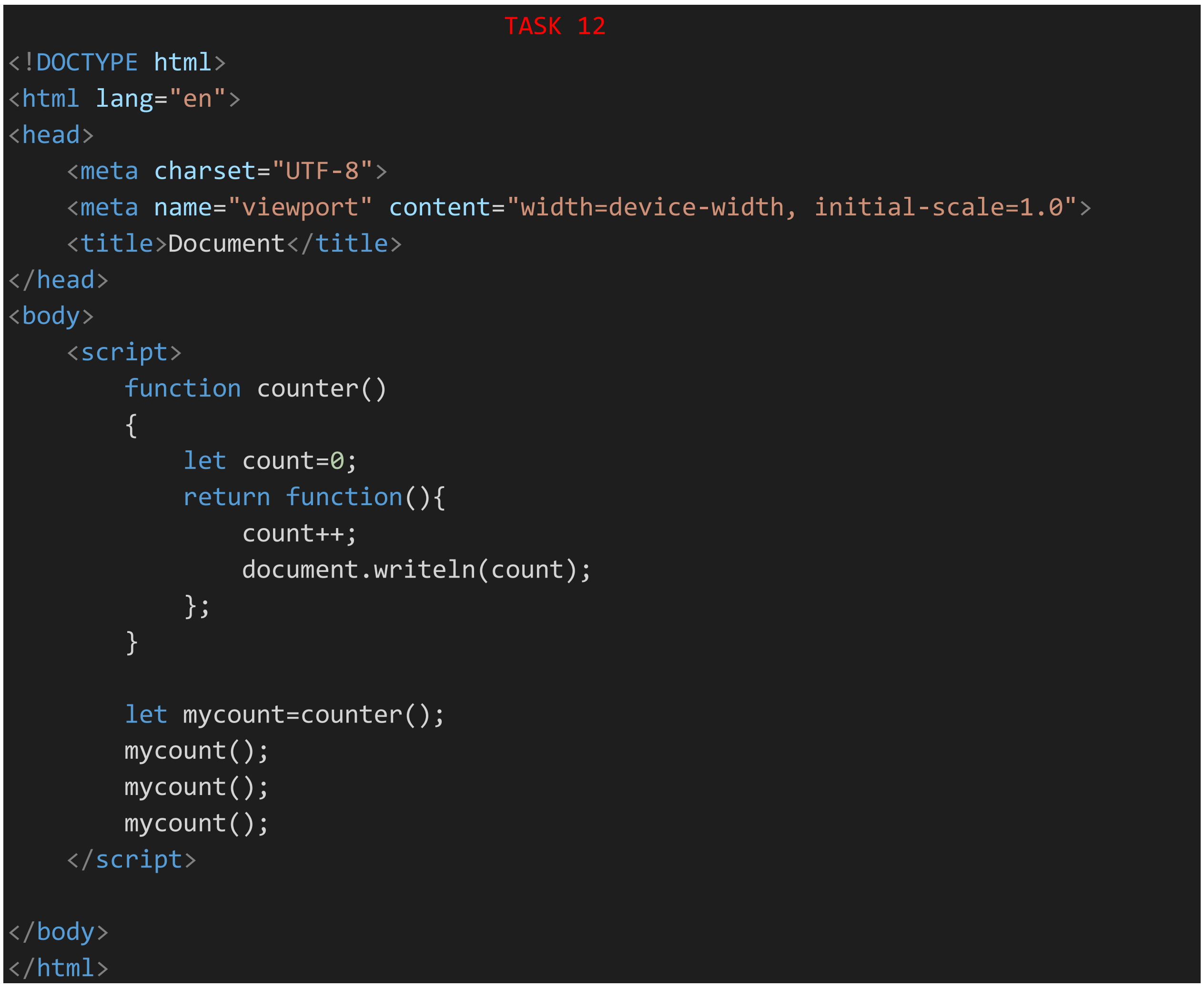


TASK 11

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function function1(n){
            return function function2(){
                document.writeln("value:"+n);
            };
        }
        let a=function1(5);
        a();
    </script>

</body>
</html>
```

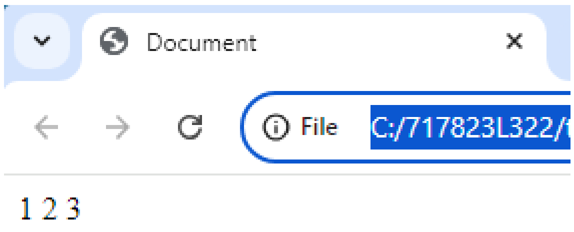


```
value:5
```

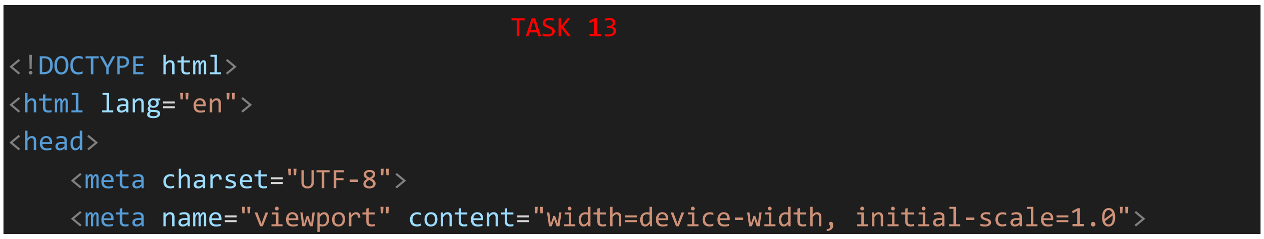


TASK 12

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function counter()
        {
            let count=0;
            return function(){
                count++;
                document.writeln(count);
            };
        }

        let mycount=counter();
        mycount();
        mycount();
        mycount();
    </script>

</body>
</html>
```

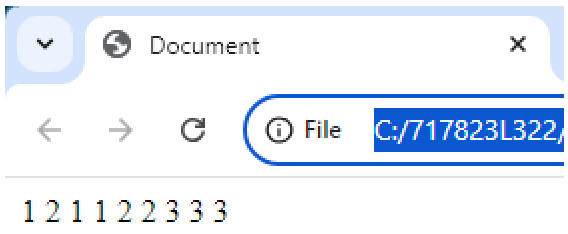


```
1 2 3
```



TASK 13

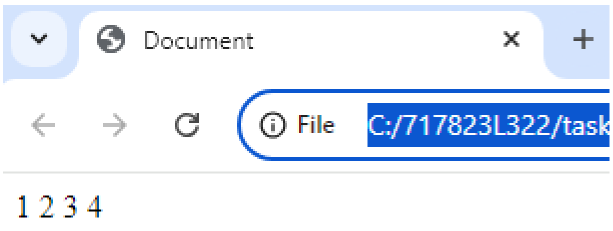
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function counter(){
            let count=0;
            return function(){
                count++;
                document.writeln(count);
            }
        }
        let mycount1=counter();
        let mycount2=counter();
        let mycount3=counter();
        mycount1();
        mycount1();
        mycount2();
        mycount3();
        mycount3();
        mycount2();
        mycount1();
        mycount2();
        mycount3();

    </script>

</body>
</html>
```



```
1 2 1 1 2 2 3 3 3
```

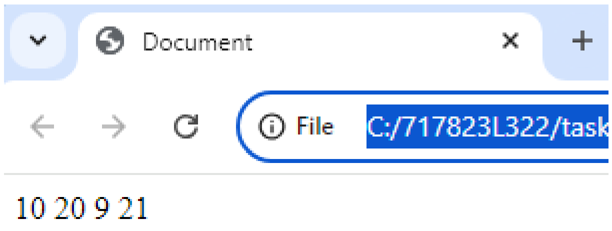
TASK 14

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function counter(){
            let count=0;
            return function(){
                count++;
                return count;
            }
        }
        let mycount=counter();
        document.writeln(mycount());
        document.writeln(mycount());
        document.writeln(mycount());
        document.writeln(mycount());


    </script>

</body>
</html>
```

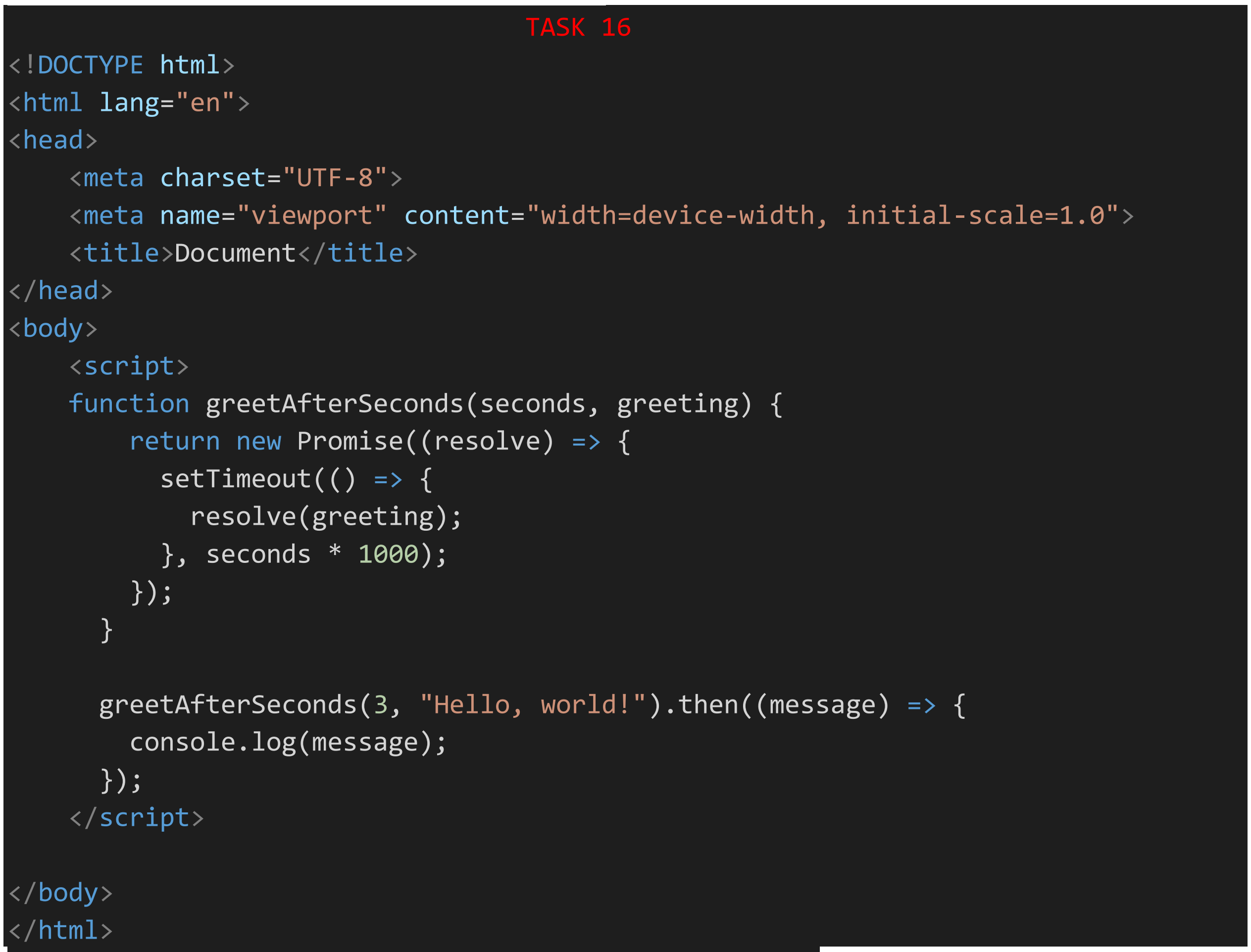


```
1 2 3 4
```

```
                               TASK 15
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function multiplierFactory(multiplier){
            return function(num){
                return num*multiplier;
            }
        }
        let double=multiplierFactory(2);
        let triple=multiplierFactory(3);
        document.writeln(double(5));
        document.writeln(double(10));
        document.writeln(triple(3));
        document.writeln(triple(7));
        </script>


</body>
</html>
```

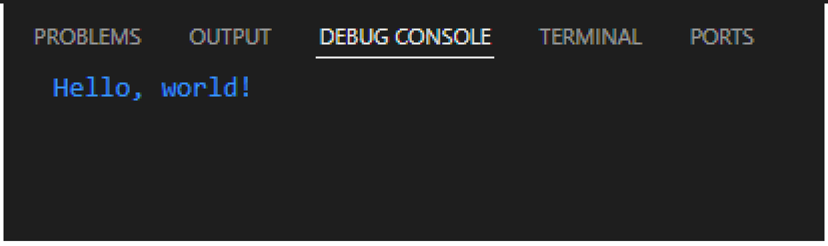


```
10 20 9 21
```

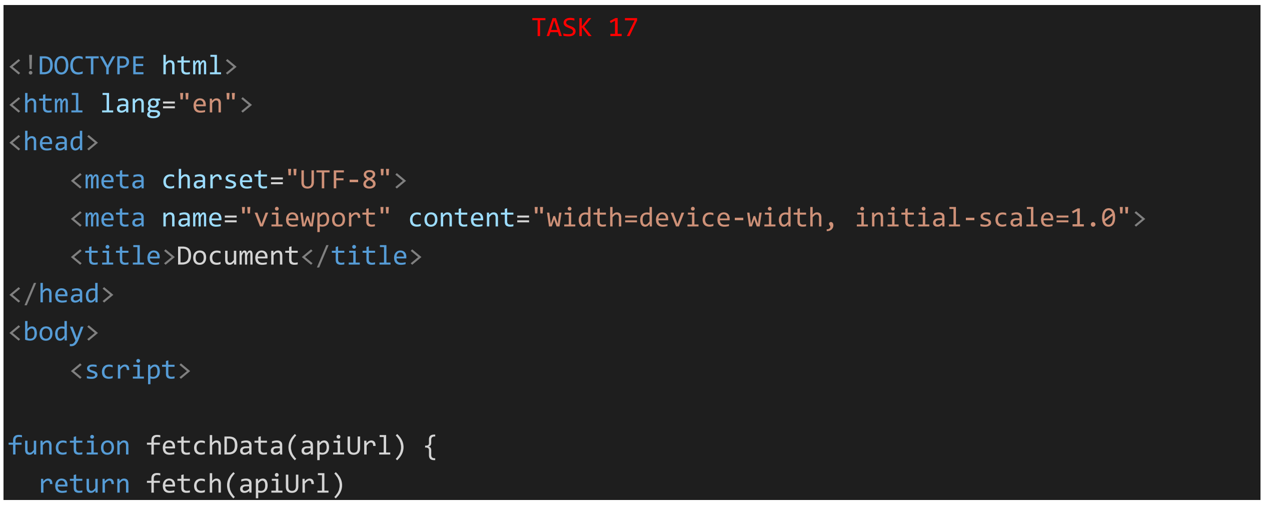


TASK 16

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
    function greetAfterSeconds(seconds, greeting) {
        return new Promise((resolve) => {
          setTimeout(() => {
            resolve(greeting);
          }, seconds * 1000);
        });
      }

      greetAfterSeconds(3, "Hello, world!").then((message) => {
        console.log(message);
      });
    </script>

</body>
</html>
```

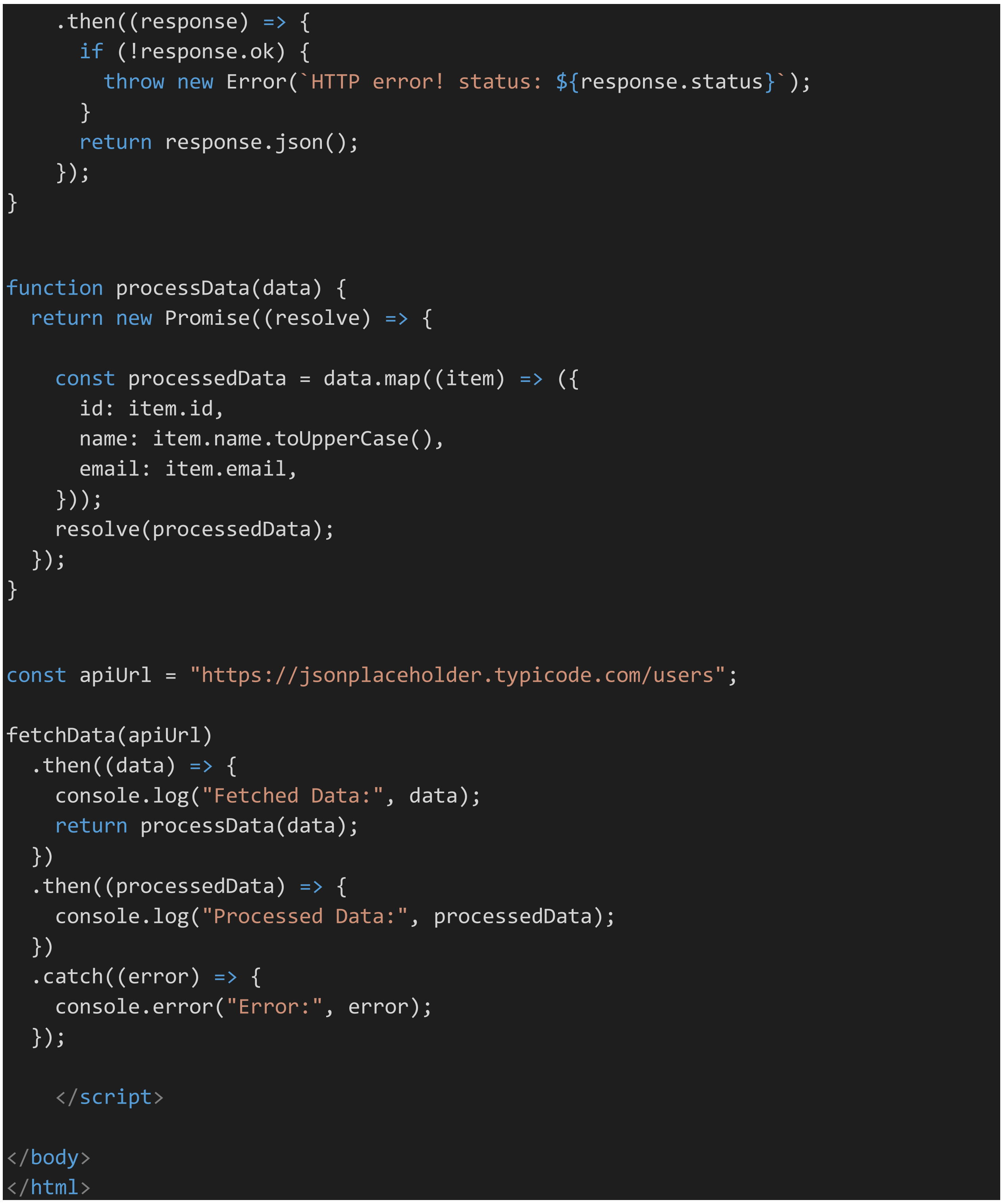


```
Hello, world!
```

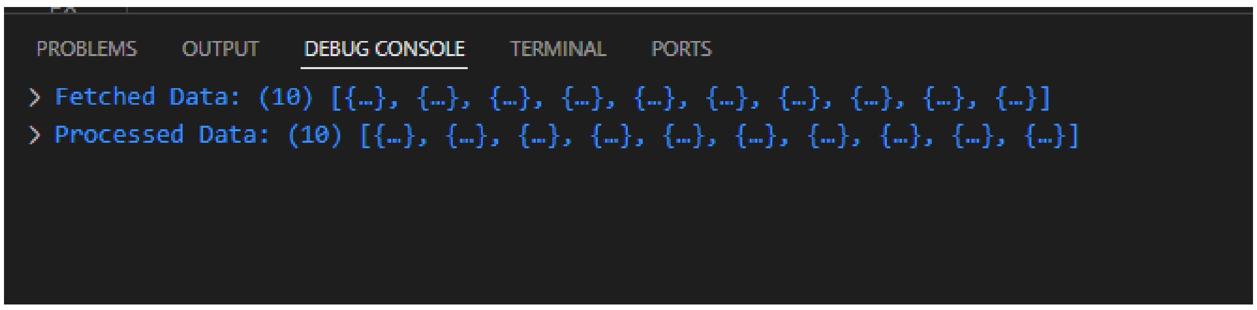


TASK 17

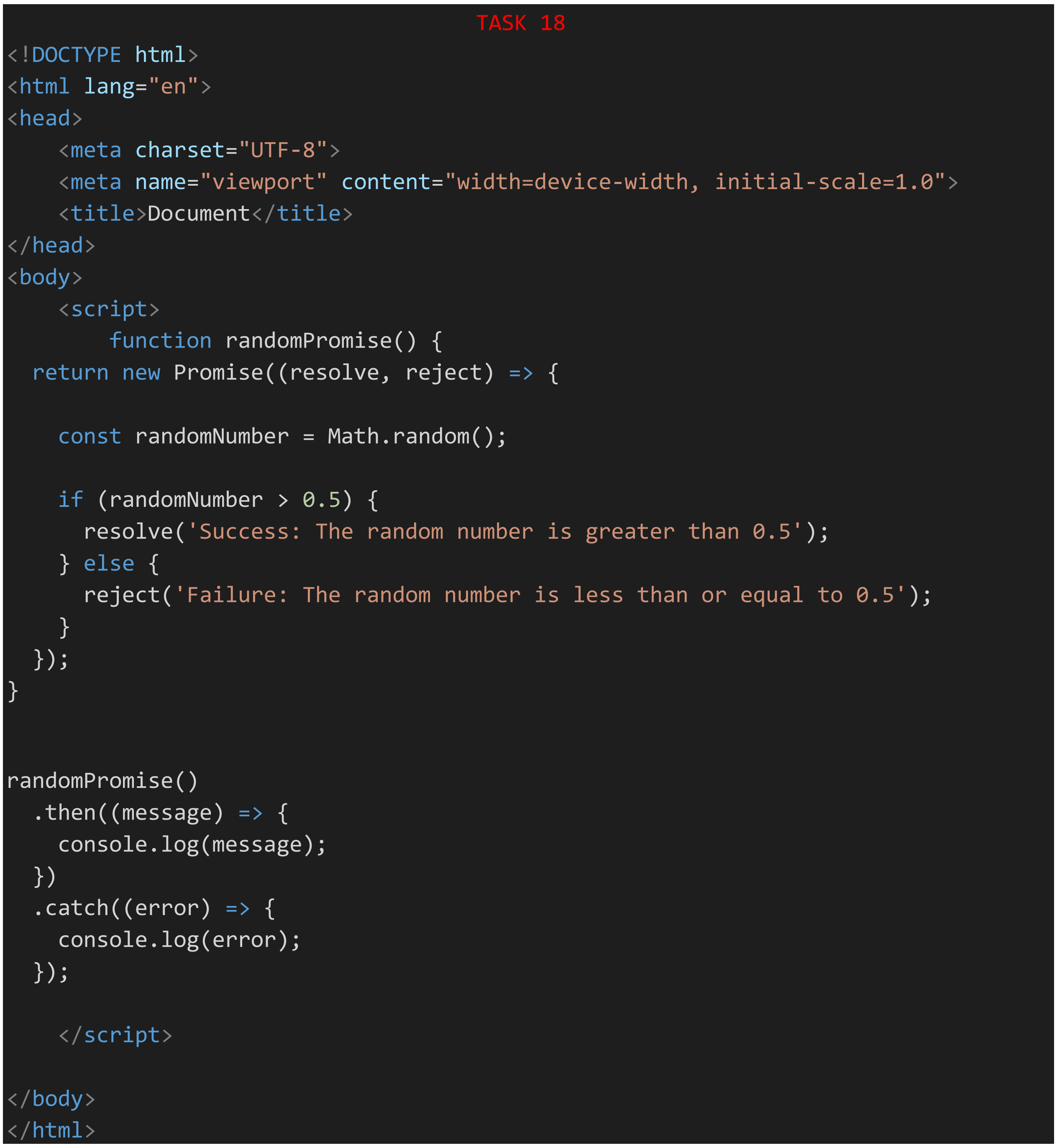
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>

function fetchData(apiUrl) {
  return fetch(apiUrl)
```



```
    .then((response) => {
      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }
      return response.json();
    });
}


function processData(data) {
  return new Promise((resolve) => {

    const processedData = data.map((item) => ({
      id: item.id,
      name: item.name.toUpperCase(),
      email: item.email,
    }));
    resolve(processedData);
  });
}


const apiUrl = "https://jsonplaceholder.typicode.com/users";

fetchData(apiUrl)
  .then((data) => {
    console.log("Fetched Data:", data);
    return processData(data);
  })
  .then((processedData) => {
    console.log("Processed Data:", processedData);
  })
  .catch((error) => {
    console.error("Error:", error);
  });

    </script>

</body>
</html>
```

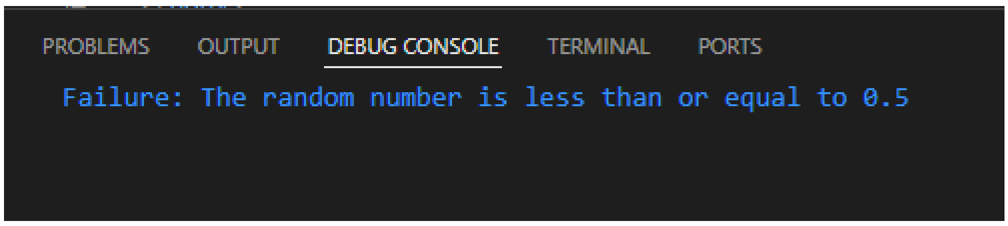


```
> Fetched Data: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
> Processed Data: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
```



TASK 18

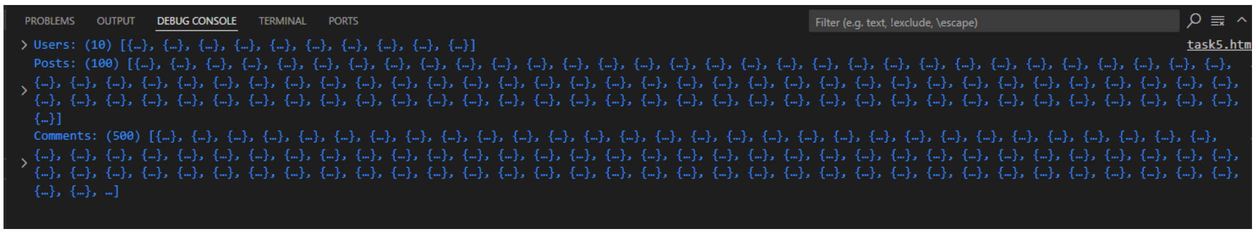
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function randomPromise() {
  return new Promise((resolve, reject) => {

    const randomNumber = Math.random();

    if (randomNumber > 0.5) {
      resolve('Success: The random number is greater than 0.5');
    } else {
      reject('Failure: The random number is less than or equal to 0.5');
    }
  });
}


randomPromise()
  .then((message) => {
    console.log(message);
  })
  .catch((error) => {
    console.log(error);
  });

    </script>

</body>
</html>
```



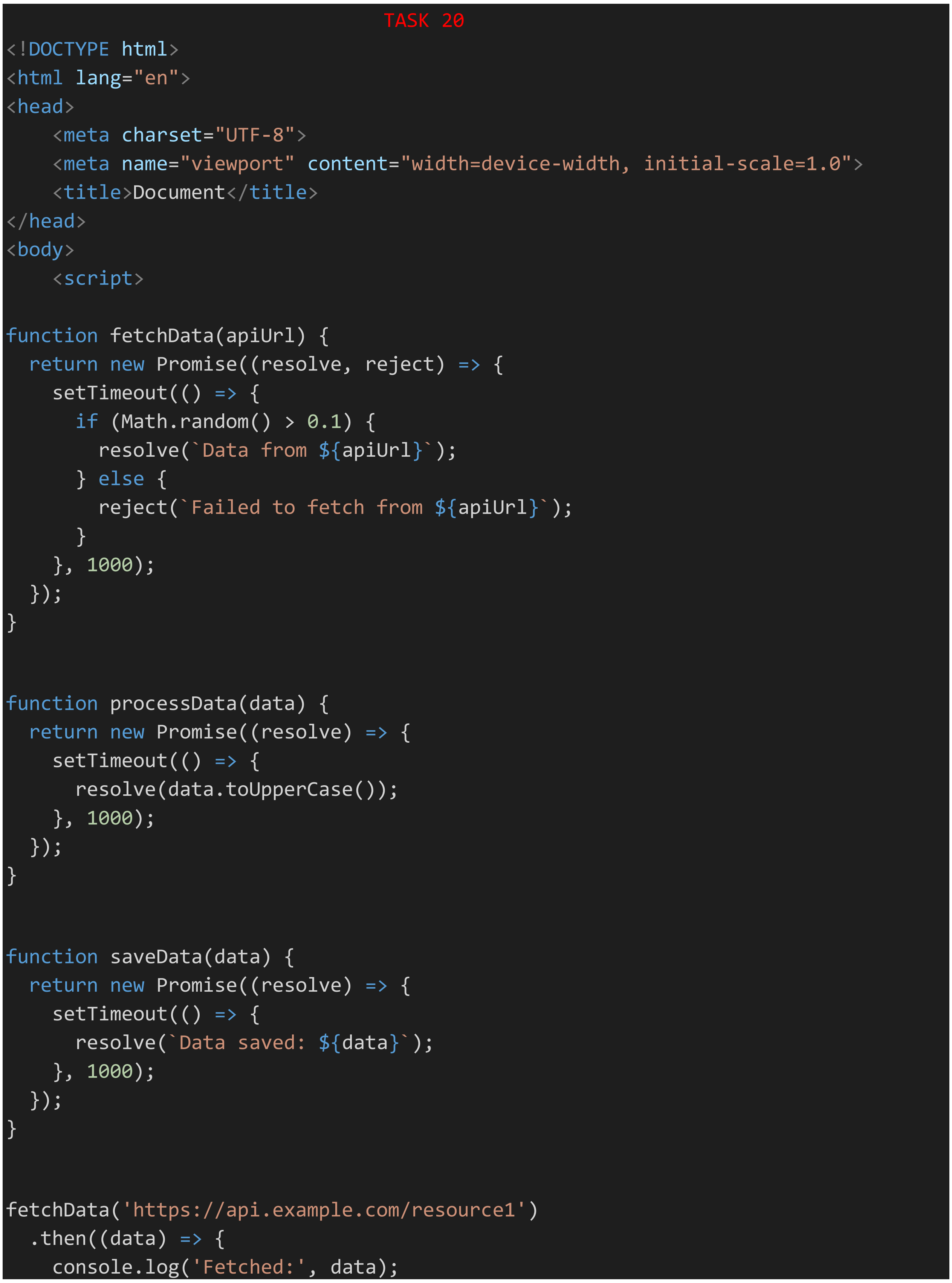
```
Failure: The random number is less than or equal to 0.5
```

TASK 19

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>

function fetchData(apiUrl) {
  return fetch(apiUrl)
    .then((response) => {
      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }
      return response.json();
    });
}


const apiUrls = [
  'https://jsonplaceholder.typicode.com/users',
  'https://jsonplaceholder.typicode.com/posts',
  'https://jsonplaceholder.typicode.com/comments'
];


Promise.all(apiUrls.map(url => fetchData(url)))
  .then((results) => {

    const [users, posts, comments] = results;

    console.log("Users:", users);
    console.log("Posts:", posts);
    console.log("Comments:", comments);
  })
  .catch((error) => {
    console.error("Error:", error);
  });

    </script>

</body>
</html>
```

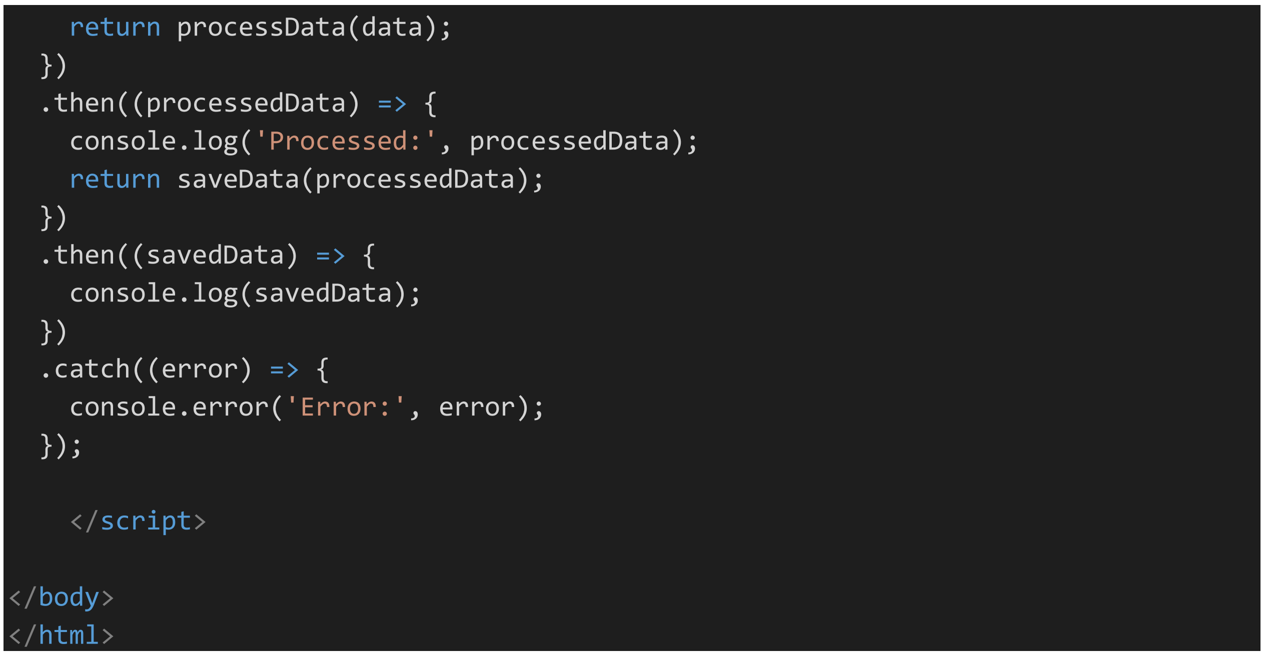


```
> Users: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]                                    task5.htm
  Posts: (100) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
> {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}]
  Comments: (500) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
> {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}, {…}, …]
```

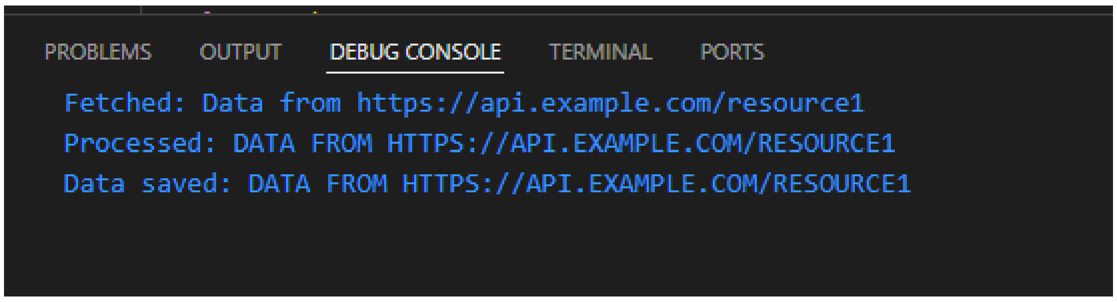


TASK 20

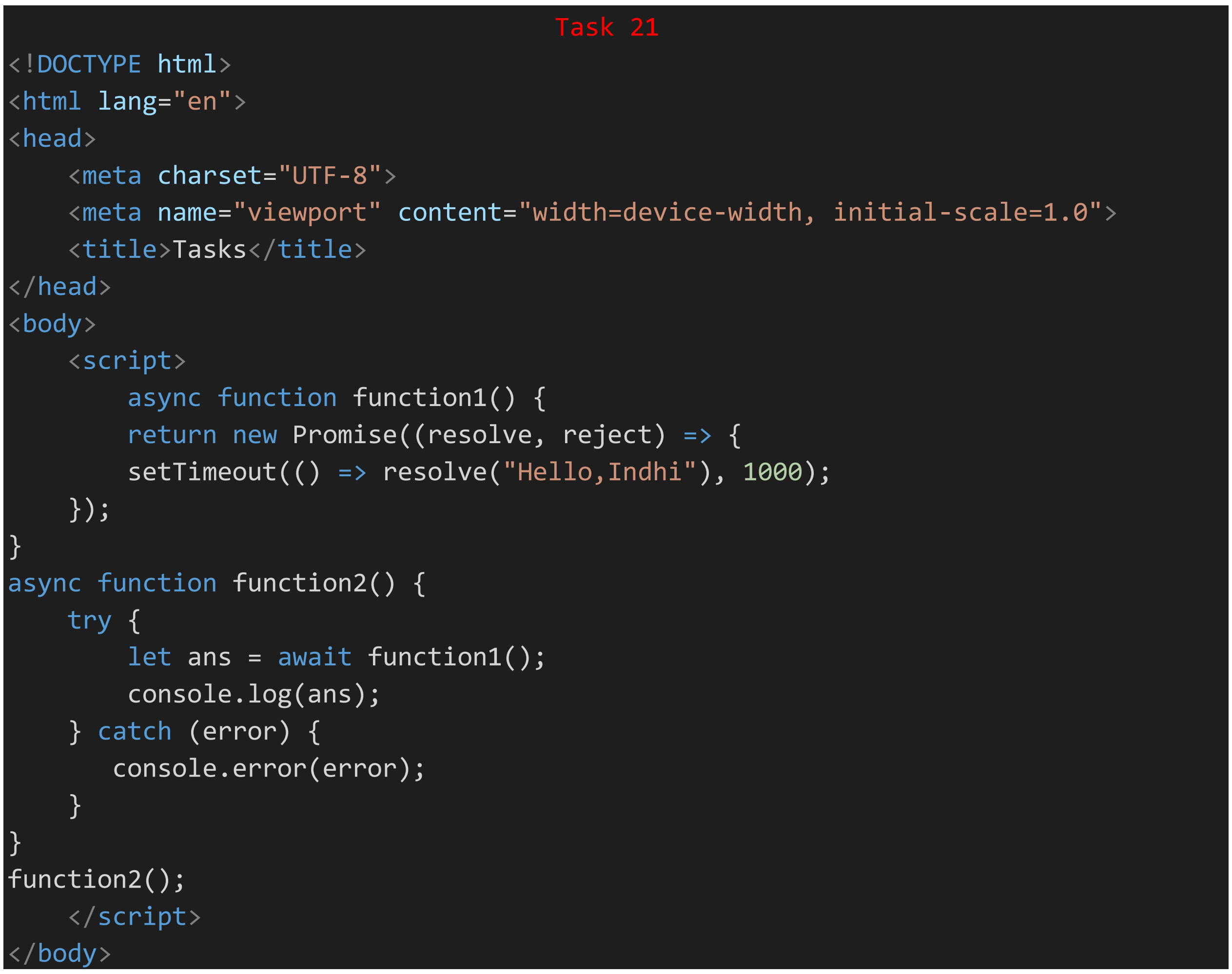
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>

function fetchData(apiUrl) {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (Math.random() > 0.1) {
        resolve(`Data from ${apiUrl}`);
      } else {
        reject(`Failed to fetch from ${apiUrl}`);
      }
    }, 1000);
  });
}


function processData(data) {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve(data.toUpperCase());
    }, 1000);
  });
}


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


fetchData('https://api.example.com/resource1')
  .then((data) => {
    console.log('Fetched:', data);
```



```
    return processData(data);
  })
  .then((processedData) => {
    console.log('Processed:', processedData);
    return saveData(processedData);
  })
  .then((savedData) => {
    console.log(savedData);
  })
  .catch((error) => {
    console.error('Error:', error);
  });

    </script>

</body>
</html>
```



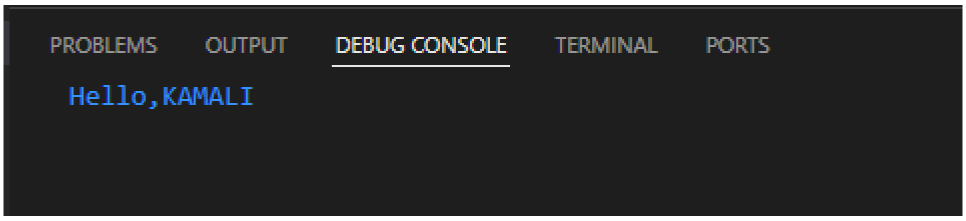
```
Fetched: Data from https://api.example.com/resource1
Processed: DATA FROM HTTPS://API.EXAMPLE.COM/RESOURCE1
Data saved: DATA FROM HTTPS://API.EXAMPLE.COM/RESOURCE1
```



Task 21

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tasks</title>
</head>
<body>
    <script>
        async function function1() {
        return new Promise((resolve, reject) => {
        setTimeout(() => resolve("Hello,Indhi"), 1000);
    });
}
async function function2() {
    try {
        let ans = await function1();
        console.log(ans);
    } catch (error) {
       console.error(error);
    }
}
function2();
    </script>
</body>
```

```html
</html>
```

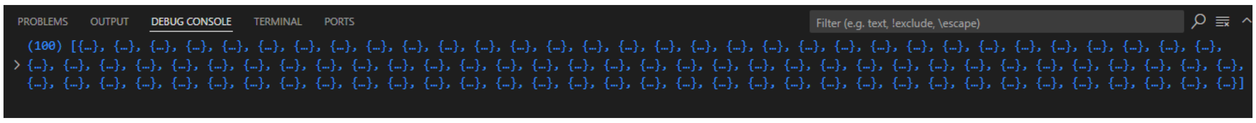


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Hello,KAMALI
```

TASK 22 & 23

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tasks</title>
</head>
<body>
<script>

async function fetchData(apiUrl) {

try {
    const response = await fetch(apiUrl);
    const data = await response.json();
    console.log(data);
} catch (error) {
console.log("Error fetching data:", error);
}
}
fetchData("https://jsonplaceholder.typicode.com/posts");
</script>
</body>
</html>
```



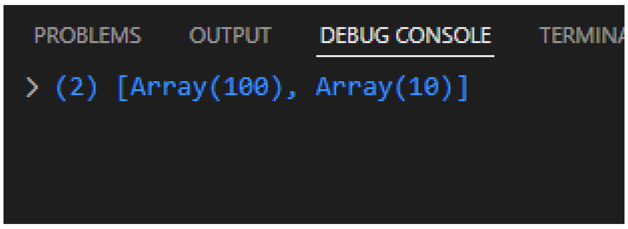
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
  (100) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
> {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
```

TASK 24

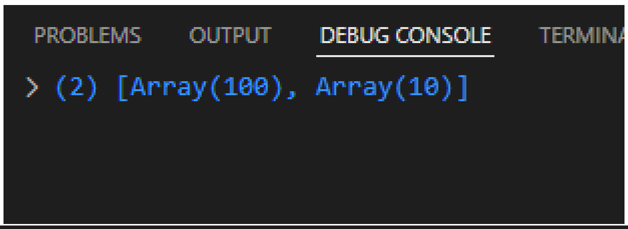
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        async function fetchMultiple(apiUrls) {
        try {
            const responses = await Promise.all(apiUrls.map(url => fetch(url)));
            const data = await Promise.all(responses.map(res => res.json()));
```

```
           console.log(data);
        } catch (error) {
        console.error("Error fetching data:", error.message);
       }
    }
fetchMultiple([
  "https://jsonplaceholder.typicode.com/posts",
  "https://jsonplaceholder.typicode.com/users"

]);
</script>
</body>
</html>
```



```
> (2) [Array(100), Array(10)]
```

```
                                  Task 25
<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8”>
    <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
    <title>Document</title>
</head>
<body>
    <script>
        async function waitForAllOperations() {
        const asyncOperation1 = new Promise(resolve => setTimeout(() =>
resolve(“Operation 1 complete”), 1000));
        const asyncOperation2 = new Promise(resolve => setTimeout(() =>
resolve(“Operation 2 complete”), 2000));
        const asyncOperation3 = new Promise(resolve => setTimeout(() =>
resolve(“Operation 3 complete”), 1500));
        try {
           const results = await Promise.all([asyncOperation1, asyncOperation2,
asyncOperation3]);
           console.log(“All operations completed:”, results);
        } catch (error) {
        console.error(“An error occurred during operations:”, error);
        }
    }
waitForAllOperations();
    </script>
</body>
</html>
```

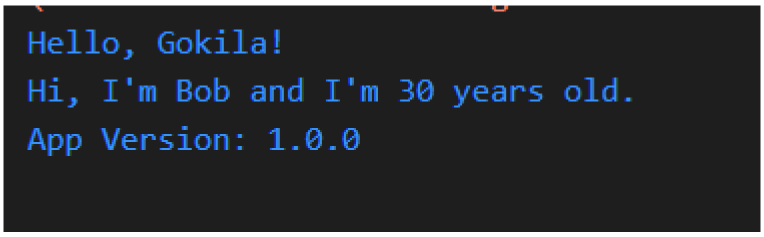


```
> (2) [Array(100), Array(10)]
```

TASK 26 & 27

```
// myModule.js


export function greet(name) {
    return `Hello, ${name}!`;
  }


  export class Person {
    constructor(name, age) {
      this.name = name;
      this.age = age;
    }

    greet() {
      return `Hi, I'm ${this.name} and I'm ${this.age} years old.`;
    }
  }


  export const appVersion = "1.0.0";
```

```
// main.js

import { greet, Person, appVersion } from './myModule.js';

console.log(greet("Gokila"));
const person = new Person("Bob", 30);
console.log(person.greet());
console.log(`App Version: ${appVersion}`);
```

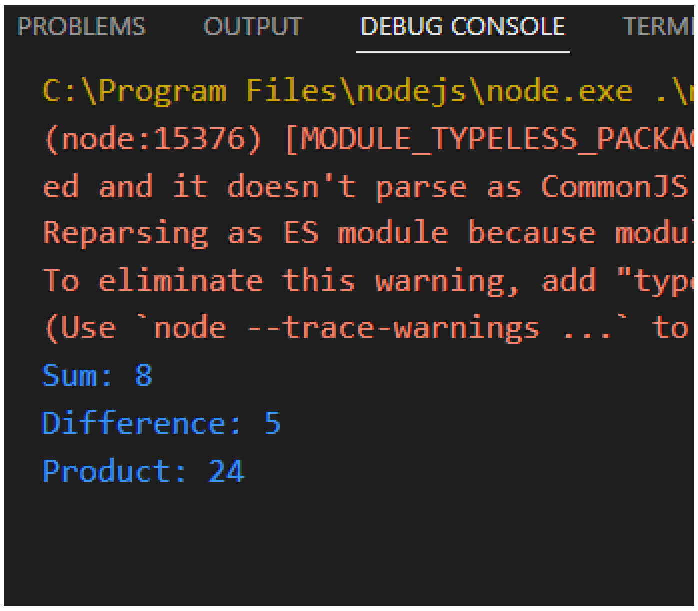


```
Hello, Gokila!
Hi, I'm Bob and I'm 30 years old.
App Version: 1.0.0
```

Task 28

```
// mathFunctions.js

export function add(a, b) {
    return a + b;
  }
  export function subtract(a, b) {
    return a - b;
  }
  export function multiply(a, b) {
    return a * b;
  }
```

```
// main.js

import { add, subtract, multiply } from './mathFunctions.js';

const sum = add(5, 3);
const difference = subtract(9, 4);
const product = multiply(4, 6);

console.log(`Sum: ${sum}`);
console.log(`Difference: ${difference}`);
console.log(`Product: ${product}`);
```

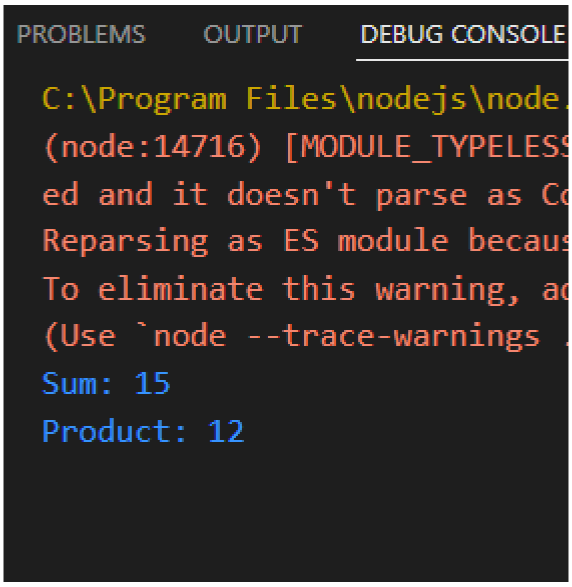


```
C:\Program Files\nodejs\node.exe .\
(node:15376) [MODULE_TYPELESS_PACKAG
ed and it doesn't parse as CommonJS
Reparsing as ES module because modul
To eliminate this warning, add "type
(Use `node --trace-warnings ...` to
Sum: 8
Difference: 5
Product: 24
```

Task 29

```
// mathFunctions.js

export function add(a, b) {
  return a + b;
}
export function subtract(a, b) {
  return a - b;
}
export function multiply(a, b) {
  return a * b;
}
export function divide(a, b) {
  if (b === 0) {
    throw new Error("Cannot divide by zero");
  }
  return a / b;
}
```

```
// main.js

import { add, multiply } from './mathFunctions.js';


const sum = add(10, 5);
const product = multiply(4, 3);

console.log(`Sum: ${sum}`);
console.log(`Product: ${product}`);
```

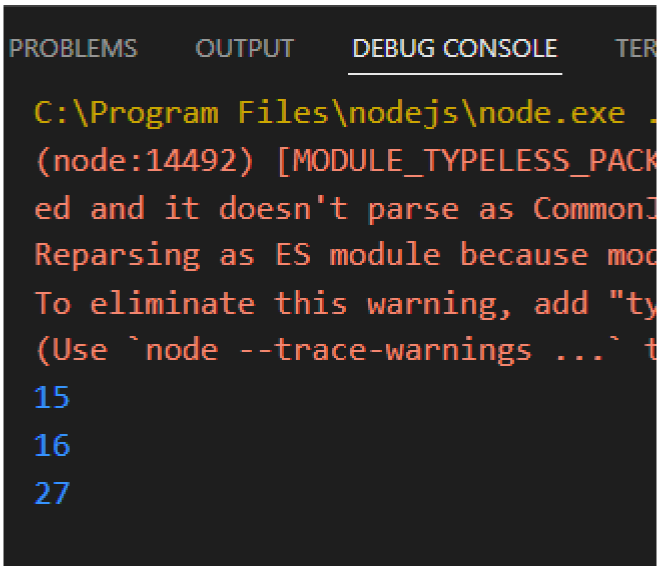


```
C:\Program Files\nodejs\node.
(node:14716) [MODULE_TYPELESS
ed and it doesn't parse as Co
Reparsing as ES module becaus
To eliminate this warning, ad
(Use `node --trace-warnings .
Sum: 15
Product: 12
```

Task 30

```
// mathOperations.js

export default function calculate(a, b, operation) {
    switch (operation) {
      case 'add':
        return a + b;
      case 'subtract':
        return a - b;
      case 'multiply':
        return a * b;
      case 'divide':
        if (b === 0) {
          throw new Error('Cannot divide by zero');
        }
        return a / b;
      default:
        throw new Error('Unknown operation');
    }
  }
  export function square(a) {
    return a * a;
  }

  export function cube(a) {
    return a * a * a;
  }
```

```
// main.js


import calculate from './mathOperations.js';
import { square, cube } from './mathOperations.js';

console.log(calculate(10, 5, 'add'));
console.log(square(4));
console.log(cube(3));
```

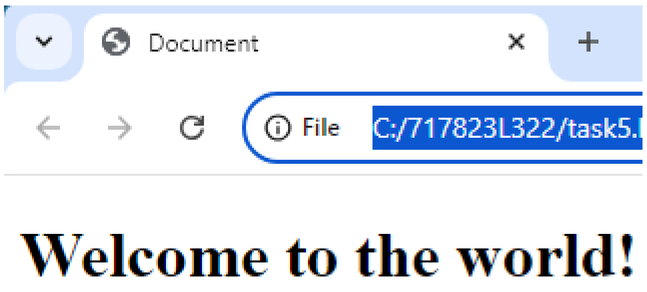


```
C:\Program Files\nodejs\node.exe .
(node:14492) [MODULE_TYPELESS_PACK
ed and it doesn't parse as CommonJ
Reparsing as ES module because mod
To eliminate this warning, add "ty
(Use `node --trace-warnings ...` t
15
16
27
```

Task 31

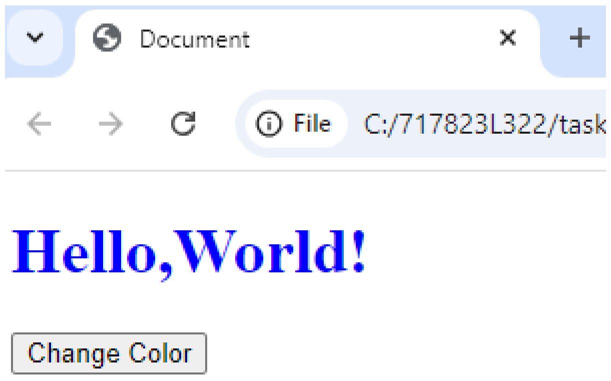
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width,initial-scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <div>
      <h1 id="myElement">Hello,World!</h1>
    </div>

    <script>
        let element = document.getElementById("myElement").innerHTML="Welcome to
the world!";
```

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



Welcome to the world!

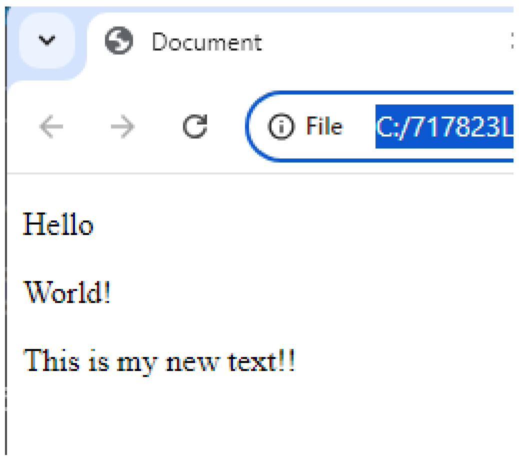
```
                                    Task 32
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width,initial-scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <div>
      <h1 id="myElement">Hello,World!</h1>
      <button onclick="changeColor()">Change Color</button>
    </div>

    <script>
      function changeColor() {
        let ele= document.getElementById("myElement");
        ele.style.color="blue";
      }
    </script>
  </body>
</html>
```



Hello,World!

Task 33

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
   <div id="new">
      <p id="p1">Hello</p>
      <p id="p2">World!</p>
   </div>
   <script>
      var tag = document.createElement("p");
      var text = document.createTextNode("This is my new text!!");
      tag.appendChild(text);
      var element = document.getElementById("new");
      element.appendChild(tag);
   </script>
</body>
</html>
```



Hello

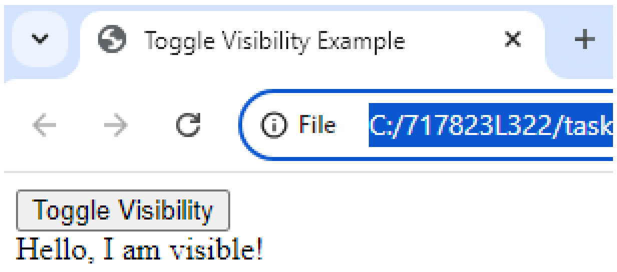
World!

This is my new text!!

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 Example</title>
</head>
<body>
    <button onclick="toggleVisibility('myElement')">Toggle Visibility</button>
    <div id="myElement" style="display: block;">Hello, I am visible!</div>
    <script src="task34.js"></script>
</body>
</html>
```

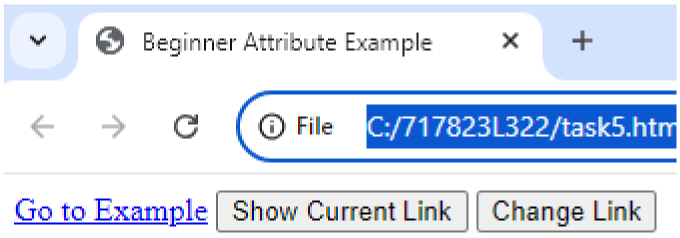
```
//js file
function toggleVisibility(elementId) {
    const element = document.getElementById(elementId);
    if (!element) {
        console.error(`Element with ID "${elementId}" not found.`);
        return;
    }

    if (element.style.display === "none") {
        element.style.display = "block";
    } else {
        element.style.display = "none";
    }
}
```



Toggle Visibility
Hello, I am visible!

```
                                  Task 35
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Beginner Attribute Example</title>
</head>
<body>
    <!-- A simple link -->
    <a id="myLink" href="https://github.com" target="_blank">Go to Example</a>
    <button onclick="handleAttribute('myLink', 'href')">Show Current
Link</button>
    <button onclick="handleAttribute('myLink', 'href',
'https://google.com')">Change Link</button>
</body>
<script>
```

```
function handleAttribute(elementId, attributeName, newValue) {
    const element = document.getElementById(elementId);
    if (!element) {
        console.error(`Element with ID "${elementId}" not found.`);
        return;
    }

    const currentValue = element.getAttribute(attributeName);
    console.log(`Current value of "${attributeName}": ${currentValue}`);

    if (newValue) {
        element.setAttribute(attributeName, newValue);
        console.log(`Updated "${attributeName}" to: ${newValue}`);
    }
}

</script>
</html>
```


Beginner Attribute Example
File
C:/717823L322/task5.htm
Go to Example
Show Current Link
Change Link