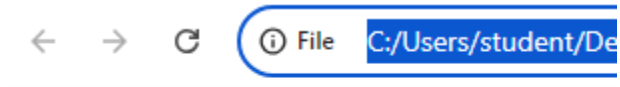
TASK 1:

```
<!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>
        let n = 5;
     function fact(n) {
        if(n==0 || n==1)
       {
        return 1;
       }
       else{
        return n* fact(n -1);
       }
     }
       document.writeln("Factorial of "+ n +":"+ fact(n));
    </script>
</body>
</html>
```

OUTPUT:



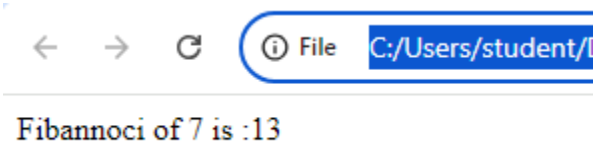
Factorial of 5:120

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>
```

```
    let n=7;
     function fibannoci(n) {
        if(n <=1)
       {
        return n;
       }
       else{
        return fibannoci(n-1) + fibannoci (n-2);
       }
     }
       document.writeln("Fibannoci of "+ n +" is :"+ fibannoci(n));
    </script>
</body>
</html>
```

OUTPUT:

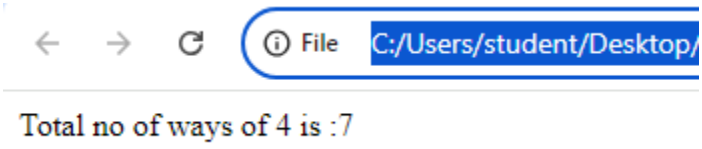


TASK 3:

```
<!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>
    let n= 4;
     function  ways(n) {
        if(n < 0)
       {
        return 0;
       }
       if(n == 0)
       {
        return 1;
       }
       else{
        return ways(n-1) + ways(n-2) + ways(n-3);
       }
```

```
    }
      document.writeln("Total no of ways of "+ n +" is :"+ ways(n));
    </script>
</body>
</html>
```

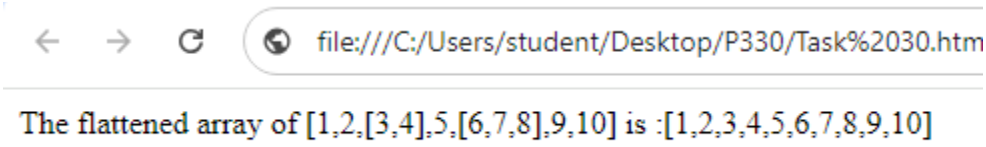
OUTPUT:



TASK 4:

```
<!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  flatten(n)
     {
        let result= [];
     for(let i = 0; i< n.length;i++)
     {
       if(Array.isArray(n[i]))
     {
        result = result.concat(flatten(n[i]));
     }
     else
     {
        result.push(n[i]);
     }
     }
     return result;
     }
       const n = [1,2,[3,4],5,[6,7,8],9,10];
      document.writeln("The flattened array of "+ JSON.stringify(n)+" is :"+
JSON.stringify(flatten(n)));
    </script>
</body>
</html>
```

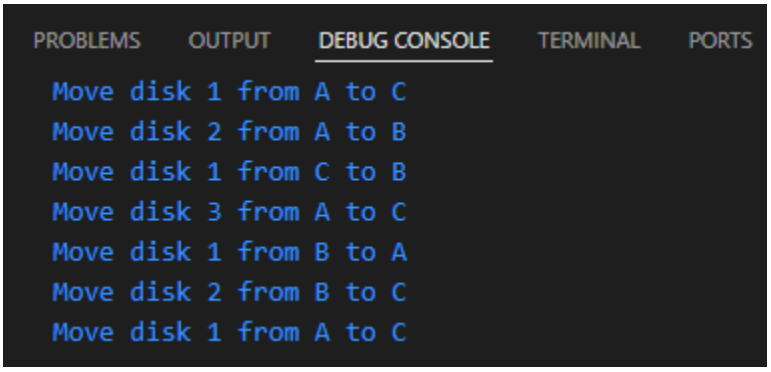
OUTPUT:



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, source, destination, auxiliary) {
        if (n === 1) {
        console.log(`Move disk 1 from ${source} to ${destination}`);
        return;
    }
    towerOfHanoi(n - 1, source, auxiliary, destination);
    console.log(`Move disk ${n} from ${source} to ${destination}`);
    towerOfHanoi(n - 1, auxiliary, destination, source);
}
let numDisks = 3;
towerOfHanoi(numDisks, 'A', 'C', 'B');
    </script>
</body>
</html>
```

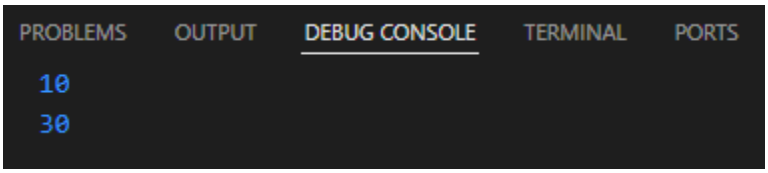
OUTPUT:



TASK 6:

```
<!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, curr) => acc + curr, 0);
}
console.log(sum(1, 2, 3, 4));
console.log(sum(5, 10, 15));
    </script>
</body>
</html>
```

OUTPUT:



TASK 7:

```
<!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 sumNumbers(...numbers) {
    return numbers.reduce((acc, num) => acc + num, 0);
}
const arr = [1, 2, 3, 4, 5];
console.log(sumNumbers(...arr));
    </script>
</body>
</html>
```

OUTPUT:

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(obj) {
    return JSON.parse(JSON.stringify(obj));
}
const original = {
    name: "John",
    address: { street: "123 Main St",
    city: "New York" }
};
const cloned = deepClone(original);
cloned.address.city = "Los Angeles";

console.log(original.address.city);
console.log(cloned.address.city);
    </script>
</body>
</html>
```

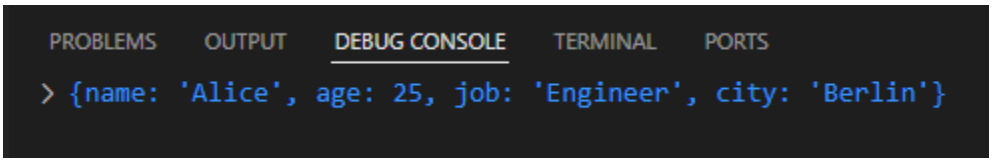
OUTPUT:

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 mergeObjects(obj1, obj2) {
    return { ...obj1, ...obj2 };
}
const obj1 = { name: "Alice", age: 25 };
const obj2 = { job: "Engineer", city: "Berlin" };

const mergedObj = mergeObjects(obj1, obj2);
console.log(mergedObj);
    </script>
</body>
</html>
```

OUTPUT:

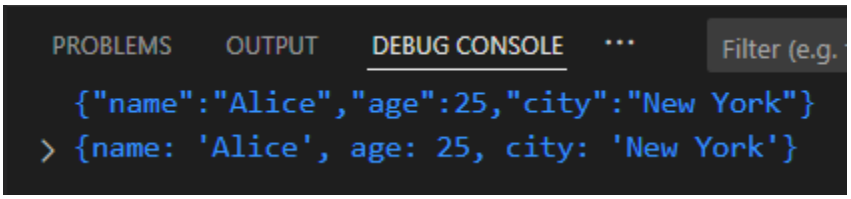


TASK 10:

```
<!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 person = {
  name: "Alice",
  age: 25,
  city: "New York"
};
const jsonString = JSON.stringify(person);
console.log(jsonString);

const parsedObject = JSON.parse(jsonString);
```

```
console.log(parsedObject);
    </script>
</body>
</html>
```

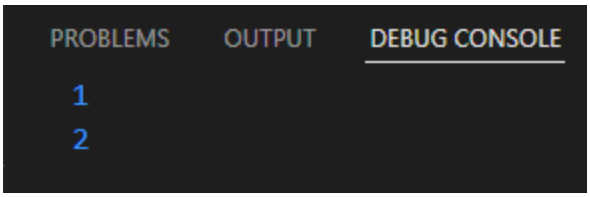
OUTPUT:



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 outerFunction() {
  let counter = 0;
  return function innerFunction() {
    counter++;
    console.log(counter);
  };
}
const increment = outerFunction();
increment();
increment();
    </script>
</body>
</html>
```

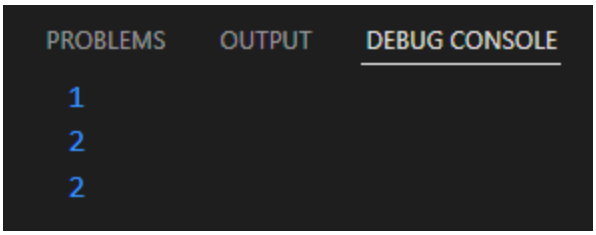
OUTPUT:



TASK 12:

```
<!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 createCounter() {
  let count = 0;
  return {
    increment: function() {
      count++;
      console.log(count);
    },
    getCount: function() {
      return count;
    }
  };
}
const counter = createCounter();
counter.increment();
counter.increment();
console.log(counter.getCount());

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

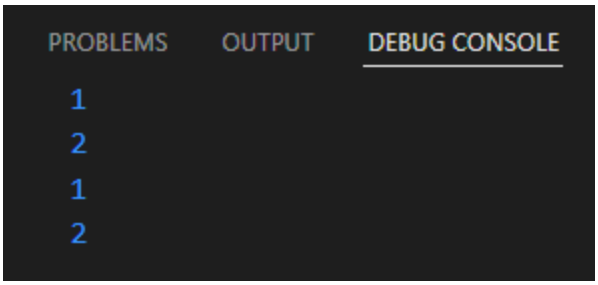
OUTPUT:



TASK 13:

```
<!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 createCounter() {
  let count = 0;
  return function() {
    count++;
    return count;
  };
}
const counter1 = createCounter();
const counter2 = createCounter();

console.log(counter1());
console.log(counter1());
console.log(counter2());
console.log(counter2());
    </script>
</body>
</html>
```

OUTPUT:

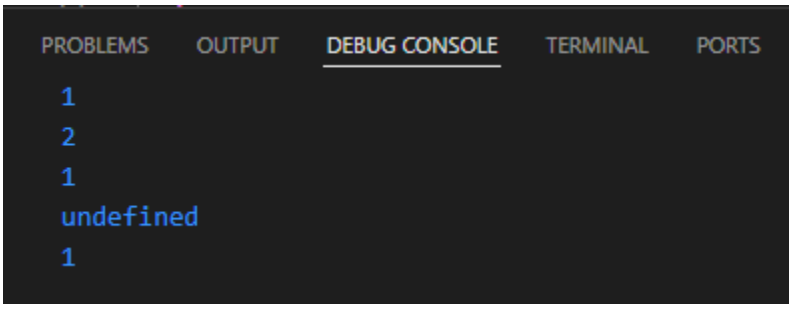


TASK 14:

```
<!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 createCounter() {
  let count = 0;

  return {
    increment: function() {
      count++;
      console.log(count);
    },
    decrement: function() {
      count--;
      console.log(count);
    },
    getCount: function() {
      return count;
    }
  };
}
const counter = createCounter();
counter.increment();
counter.increment();
counter.decrement();

console.log(counter.count);

console.log(counter.getCount());
    </script>
```

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

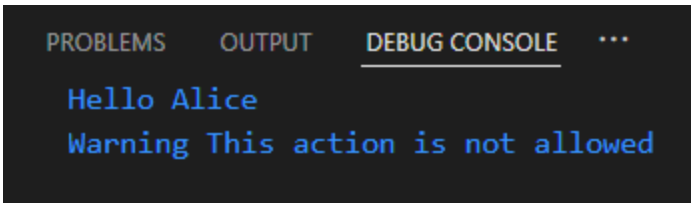
OUTPUT:



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 functionFactory(prefix) {
  return function(name) {
    console.log(`${prefix} ${name}`);
  };
}
const greet = functionFactory("Hello");
greet("Alice");

const warn = functionFactory("Warning");
warn("This action is not allowed");
    </script>
</body>
</html>
```

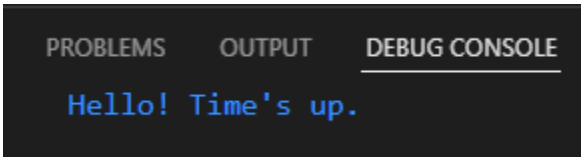
OUTPUT:



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 greetAfter(seconds) {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve("Hello! Time's up.");
    }, seconds * 1000);
  });
}
greetAfter(3).then((message) => {
  console.log(message);
});
    </script>
</body>
</html>
```

OUTPUT:

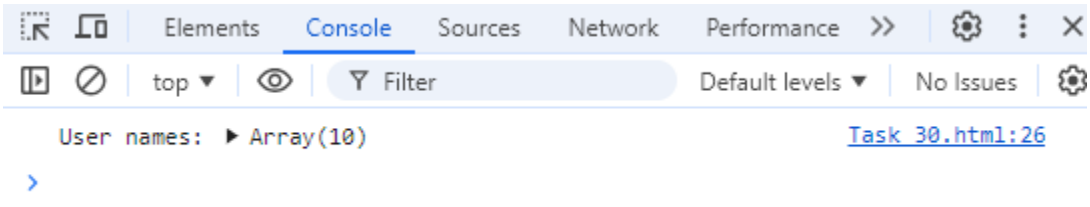


TASK 17:

```
<!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>

fetch('https://jsonplaceholder.typicode.com/users')
  .then(response => {

    if (!response.ok) {
      throw new Error('Failed to fetch data');
    }
    return response.json();
  })
  .then(users => {

    const userNames = users.map(user => user.name);
    return userNames;
  })
  .then(userNames => {

    console.log('User names:', userNames);
  })
  .catch(error => {

    console.error('Error:', error);
  });

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

OUTPUT:

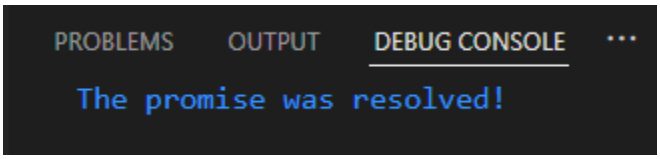


TASK 18:

```
<!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 randomOutcome() {
  return new Promise((resolve, reject) => {
    const randomNum = Math.random();
    if (randomNum > 0.5) {
      resolve("The promise was resolved!");
    } else {
      reject("The promise was rejected!");
    }
  });
}
randomOutcome()
  .then((message) => {
    console.log(message);
  })
  .catch((message) => {
    console.log(message);
  });
    </script>
</body>
</html>
```

OUTPUT:

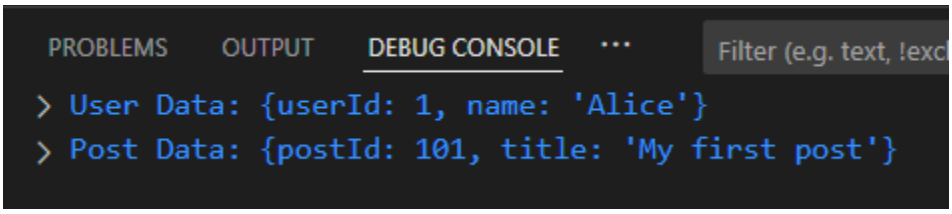


TASK 19:

```
<!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 fetchUserData() {
```

```
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve({ userId: 1, name: "Alice" });
    }, 1000);
  });
}
function fetchPostData() {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve({ postId: 101, title: "My first post" });
    }, 1500);
  });
}
Promise.all([fetchUserData(), fetchPostData()])
  .then((results) => {
    const [userData, postData] = results;
    console.log("User Data:", userData);
    console.log("Post Data:", postData);
  })
  .catch((error) => {
    console.error("Error:", error);
  });
    </script>
</body>
</html>
```

OUTPUT:

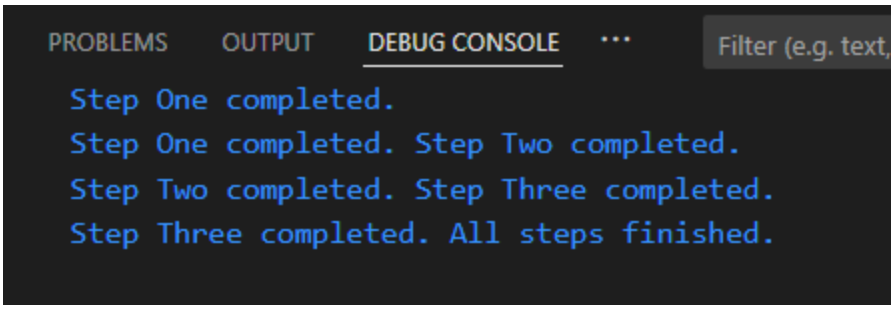


TASK 20:

```
<!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 stepOne() {
  return new Promise((resolve) => {
    setTimeout(() => {
      console.log("Step One completed.");
      resolve("Step One");
    }, 1000);
  });
}
function stepTwo(previousStep) {
  return new Promise((resolve) => {
    setTimeout(() => {
      console.log(`${previousStep} completed. Step Two completed.`);
      resolve("Step Two");
    }, 1000);
  });
}
function stepThree(previousStep) {
  return new Promise((resolve) => {
    setTimeout(() => {
      console.log(`${previousStep} completed. Step Three completed.`);
      resolve("Step Three");
    }, 1000);
  });
}
stepOne()
  .then((result) => {
    return stepTwo(result);
  })
  .then((result) => {
    return stepThree(result);
  })
  .then((finalResult) => {
    console.log(`${finalResult} completed. All steps finished.`);
  })
  .catch((error) => {
    console.error("Error during the process:", error);
  });
    </script>
</body>
</html>
```

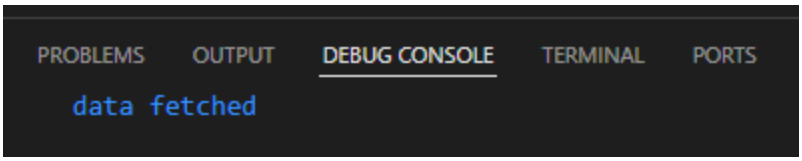
OUTPUT:



TASK 21:

```
<!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 fetchData() {
  return new Promise((resolve,reject) => {
    setTimeout(() => {
      resolve(" data fetched");
    }, 1000);
  });
}
async function getData() {
   try{
     const response = await fetchData();
        console.log(response);
    }
    catch(error) {
        console.error(error);
    }
}
getData();
    </script>
</body>
</html>
```

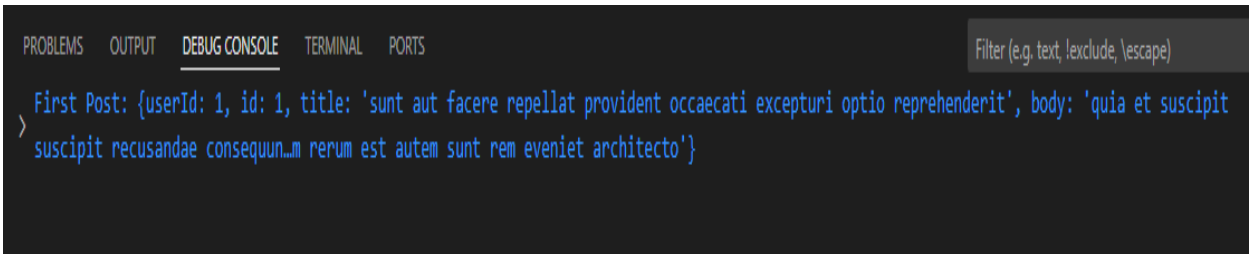
OUTPUT:



TASK 22:

```
<!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 fetchAndProcessData() {
  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('First Post:', data[0]);

  } catch (error) {
    console.error('Error fetching data:', error);
  }
}
fetchAndProcessData();
    </script>
</body>
</html>
```

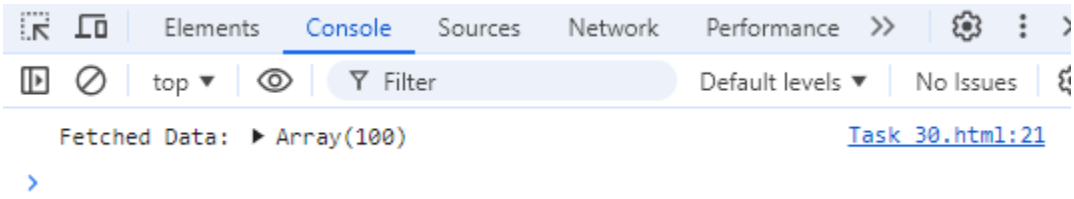
OUTPUT:



TASK 23:

```
<!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 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.message);
  }
}
fetchDataWithErrorHandling();
    </script>
</body>
</html>
```

OUTPUT:

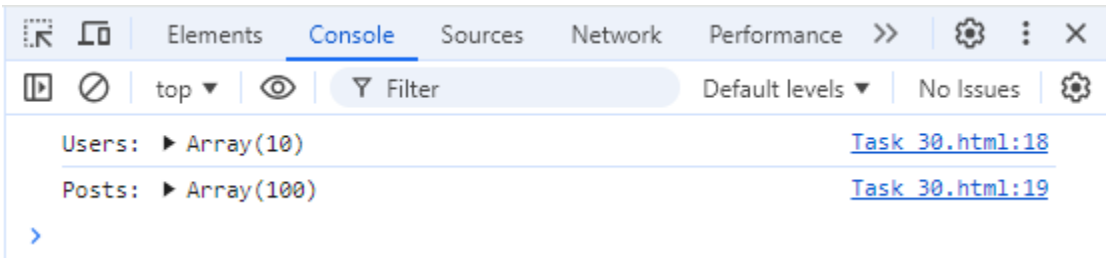


TASK 24:

```
<!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 fetchMultipleData() {
  try {

    const [users, posts] = await Promise.all([
      fetch('https://jsonplaceholder.typicode.com/users').then(response =>
response.json()),
      fetch('https://jsonplaceholder.typicode.com/posts').then(response =>
response.json())
    ]);

    console.log('Users:', users);
    console.log('Posts:', posts);

  } catch (error) {
    console.error('Error fetching multiple data:', error.message);
  }
}
fetchMultipleData();
    </script>
</body>
</html>
```

OUTPUT:

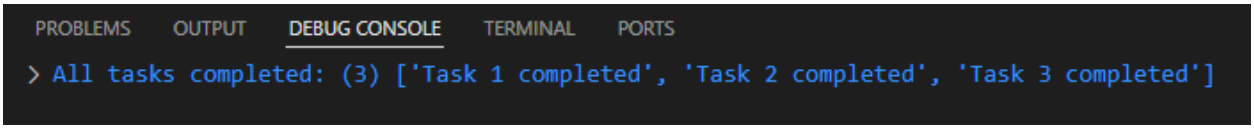


TASK 25:

```
<!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() {
  try {

    const task1 = new Promise(resolve => setTimeout(() => resolve('Task 1
completed'), 1000));
    const task2 = new Promise(resolve => setTimeout(() => resolve('Task 2
completed'), 2000));
    const task3 = new Promise(resolve => setTimeout(() => resolve('Task 3
completed'), 3000));

    const results = await Promise.all([task1, task2, task3]);
    console.log('All tasks completed:', results);

  } catch (error) {

    console.error('Error occurred while waiting for tasks:', error.message);
  }
}
waitForAllOperations();
    </script>
</body>
</html>
```

OUTPUT:



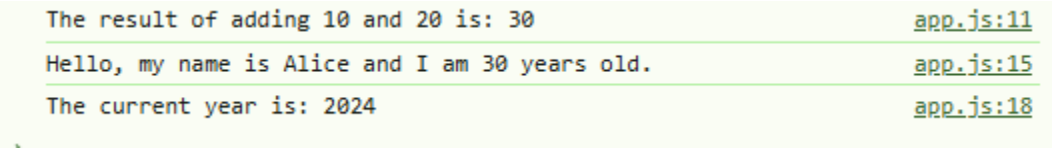
TASK 26:

```
export function add(a, b) {
    return a + b;
  }
  export 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.`;
    }
  }
  export const currentYear = new Date().getFullYear();
```

```
import { add, Person, currentYear } from './myModule.js';

const result = add(10, 20);
console.log(`The result of adding 10 and 20 is: ${result}`);

const person = new Person('Alice', 30);
console.log(person.greet());

console.log(`The current year is: ${currentYear}`);
```

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

  <script type="module" src="app.js"></script>
</body>
</html>
```

OUTPUT:



TASK 27:

```
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;
  }
import { add, subtract, multiply, divide } from './mathUtils.js';


const num1 = 20;
const num2 = 10;

console.log('Addition:', add(num1, num2));
console.log('Subtraction:', subtract(num1, num2));
console.log('Multiplication:', multiply(num1, num2));
console.log('Division:', divide(num1, num2));
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Math Operations</title>
</head>
<body>
  <h1>Math Operations Example</h1>

  <script type="module" src="app.js"></script>
</body>
</html>
```

OUTPUT:

```
Addition: 30                    app.js:10
Subtraction: 10                 app.js:11
Multiplication: 200             app.js:12
Division: 2                     app.js:13
```

TASK 28:

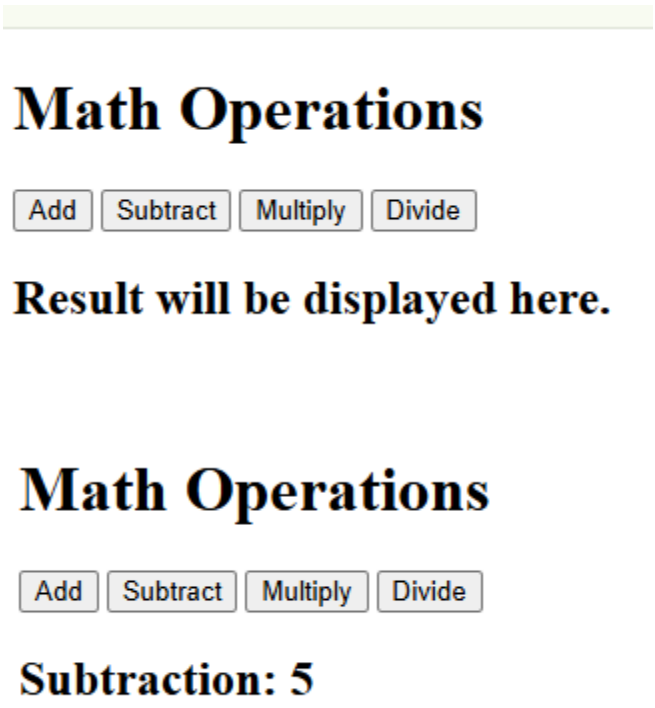
```
export function add(a, b) {
  return a + b;
```

```
}
export function subtract(a, b) {
  return a - b;
}

export function multiply(a, b) {
  return a * b;
}

export function divide(a, b) {
  if (b === 0) {
    return 'Error: Cannot divide by zero';
  }
  return a / b;
}
```

```
<!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>
  <script type="module">
    import { add, subtract, multiply, divide } from './mathUtils.js';

    function displayResult(operation, result) {
      const resultElement = document.getElementById('result');
      resultElement.innerHTML = `${operation}: ${result}`;
    }

    document.getElementById('addButton').addEventListener('click', () => {
      const result = add(5, 3);
      displayResult('Addition', result);
    });

    document.getElementById('subtractButton').addEventListener('click', () => {
      const result = subtract(9, 4);
      displayResult('Subtraction', result);
    });

    document.getElementById('multiplyButton').addEventListener('click', () => {
      const result = multiply(6, 7);
      displayResult('Multiplication', result);
    });
```

```
    document.getElementById('divideButton').addEventListener('click', () => {
      const result = divide(8, 2);
      displayResult('Division', result);
    });
  </script>
</head>
<body>
  <h1>Math Operations</h1>

  <button id="addButton">Add</button>
  <button id="subtractButton">Subtract</button>
  <button id="multiplyButton">Multiply</button>
  <button id="divideButton">Divide</button>

  <h2 id="result">Result will be displayed here.</h2>
</body>
</html>
```

OUTPUT:



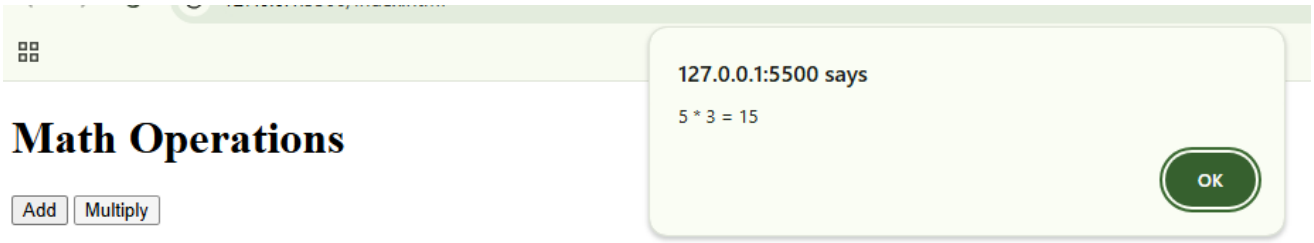
TASK 29:

```
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) {
      return 'Cannot divide by zero';
    }
    return a / b;
  }
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Named Import Example</title>
</head>
<body>
  <h1>Math Operations</h1>
  <button id="addButton">Add</button>
  <button id="multiplyButton">Multiply</button>

  <script type="module">
    import { add, multiply } from './mathFunctions.js';

    const addButton = document.getElementById('addButton');
    const multiplyButton = document.getElementById('multiplyButton');

    addButton.addEventListener('click', () => {
      const result = add(5, 3);
      alert(`5 + 3 = ${result}`);
    });
    multiplyButton.addEventListener('click', () => {
      const result = multiply(5, 3);
      alert(`5 * 3 = ${result}`);
    });
 </script>
</body>
```

```
</html>
```

OUTPUT:

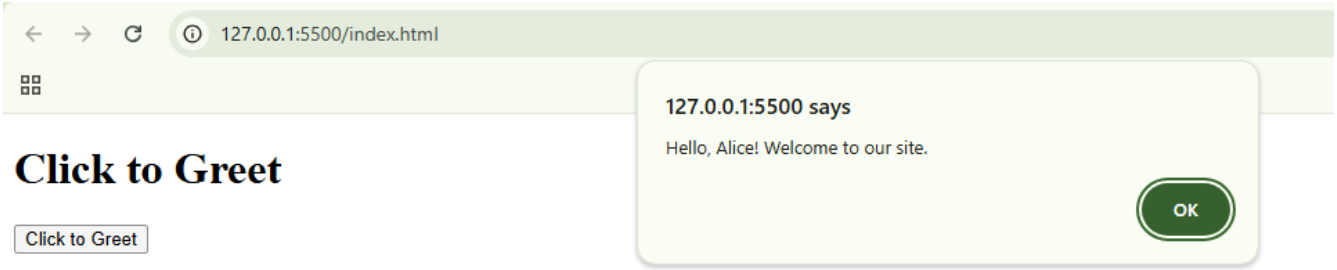


TASK 30:

```
function greet(name) {
    return `Hello, ${name}! Welcome to our site.`;
  }
  export default greet;
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Default Export & Import Example</title>
</head>
<body>
  <h1>Click to Greet</h1>
  <button id="greetButton">Click to Greet</button>

  <script type="module">
    import greet from './greet.js';
    const greetButton = document.getElementById('greetButton');
    greetButton.addEventListener('click', () => {
       const message = greet('Alice');
      alert(message);
    });
  </script>
</body>
</html>
```

OUTPUT:



TASK 31:

```
<!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="christmas.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:

Hello, World!

Change Greeting

Click Me!

Add New Paragraph

This is a toggleable content section.

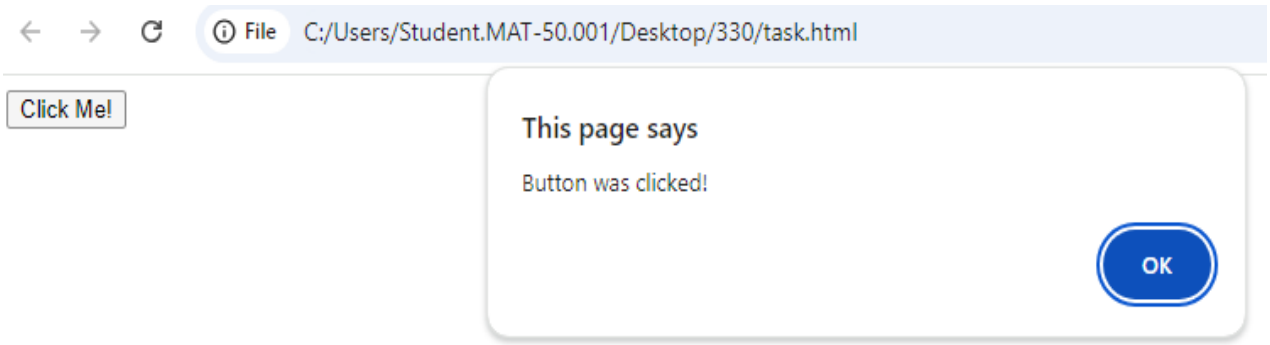
Toggle Visibility

Change Image Attributes

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:

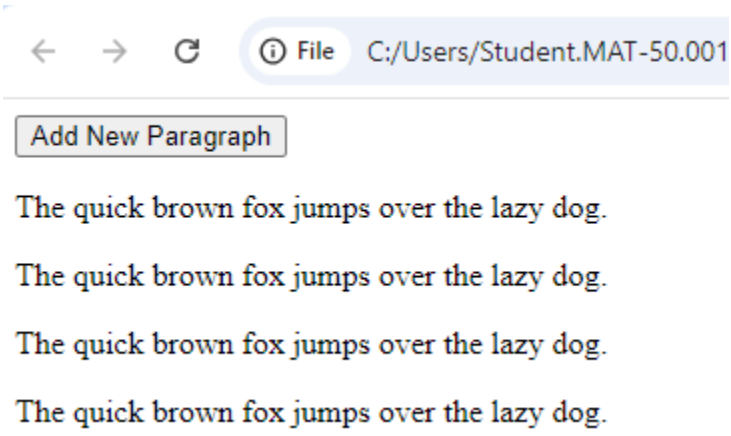


TASK 33:

```
<!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:

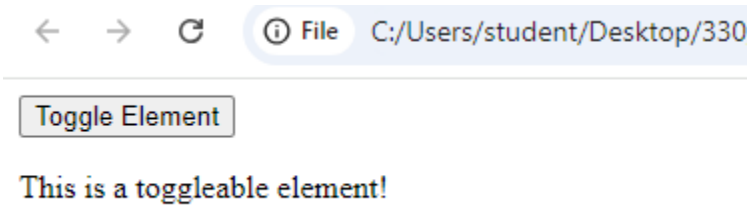


TASK 34:

```
<!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>
    <style>
        .hidden {
            display: none;
        }
    </style>
</head>
<body>
    <button onclick="toggleVisibility()">Toggle Element</button>
    <div id="myElement">
        <p>This is a toggleable element!</p>
    </div>
    <script>
        function toggleVisibility() {
```

```
            var element = document.getElementById('myElement');
            element.classList.toggle('hidden');
        }
    </script>
</body>
</html>
```

OUTPUT:



TASK 35:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DOM Attribute Example</title>
</head>
<body>
    <h2>Image and Link Example</h2>
    <img id="myImage" src="Image1.jpg" alt="Image 1">

    <br>
    <a id="myLink" href="https://example.com" target="_blank">Go to Example</a>

    <button onclick="modifyAttributes()">Modify Attributes</button>
    <script>
        function modifyAttributes() {
            var imgElement = document.getElementById('myImage');
            var linkElement = document.getElementById('myLink');

            console.log('Current Image Source: ',
imgElement.getAttribute('src'));
            imgElement.setAttribute('src', 'images.jpg');
            console.log('New Image Source: ', imgElement.getAttribute('src'));

            console.log('Current Link Href: ', linkElement.getAttribute('href'));

            linkElement.setAttribute('href', 'https://openai.com');
```

```
            console.log('New Link Href: ', linkElement.getAttribute('href'));
        }
    </script>
</body>
</html>
```

OUTPUT:

Image and Link Example

Go to Example Modify Attributes

File C:/Users/studen

Image and Link Example

Go to Example Modify Attributes

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Current Image Source:  Image1.jpg
New Image Source:  images.jpg
Current Link Href:  https://example.com
New Link Href:  https://openai.com
```