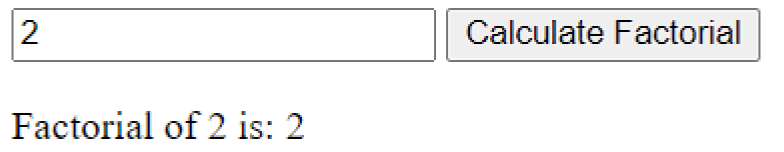
MERN Stack Training

Weekly Tasks

1. Recursion and stack:

Task 1:

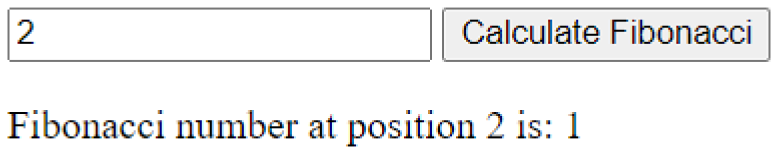
```
    function factorial(n) {
        if (n === 0 || n === 1) {
            return 1;
        }
        return n * factorial(n - 1);
    }
    function calculateFactorial() {
        const number =
parseInt(document.getElementById("numberInput").value);

        if (isNaN(number) || number < 0) {
            document.getElementById("result").textContent = "Please
enter a valid non-negative number.";
            return;
        }

        const result = factorial(number);
        document.getElementById("result").textContent = `Factorial
of ${number} is: ${result}`;
    }
```



Factorial of 2 is: 2

Task 2:

```
    function fibonacci(n) {
            if (n === 0) {
                return 0;
            }
            if (n === 1) {
                return 1;
            }
            return fibonacci(n - 1) + fibonacci(n - 2);
        }
        function calculateFibonacci() {
            const number =
parseInt(document.getElementById("numberInput").value);
            if (isNaN(number) || number < 0) {
                document.getElementById("result").textContent = "Please
enter a valid non-negative number.";
                return;
            }
            const result = fibonacci(number);
            document.getElementById("result").textContent = "Fibonacci
number at position " + number + " is: " + result;
        }
```

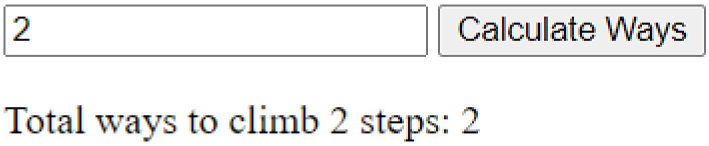


```
Fibonacci number at position 2 is: 1
```

Task 3:

```
  function waysToClimb(n) {
            if (n === 0) {
                return 1;
            }
            if (n < 0) {
                return 0;
            }
```

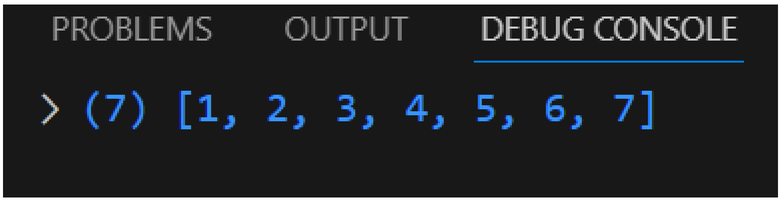
```
      return waysToClimb(n - 1) + waysToClimb(n - 2) +
waysToClimb(n - 3);
    }
    function calculateWays() {
      const steps =
parseInt(document.getElementById("stepsInput").value);
      if (isNaN(steps) || steps < 0) {
        document.getElementById("result").textContent = "Please
enter a valid non-negative number.";
        return;
      }
      const result = waysToClimb(steps);
      document.getElementById("result").textContent = "Total
ways to climb " + steps + " steps: " + result;
    }
```



Total ways to climb 2 steps: 2

Task 4:

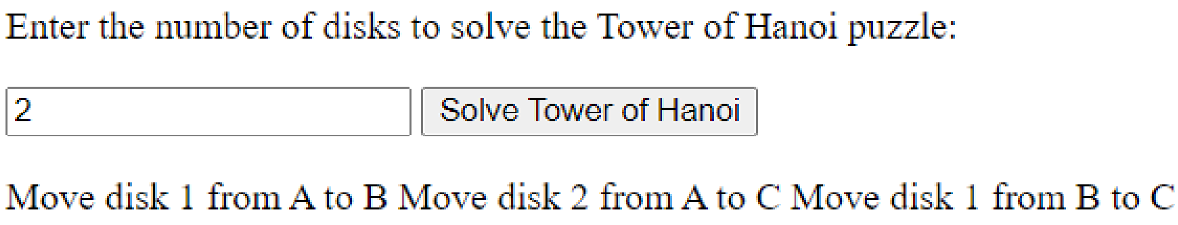
```
function flatten(arr) {
  let result = [];
  for (let i = 0; i < arr.length; i++) {
    if (Array.isArray(arr[i])) {
      result = result.concat(flatten(arr[i]));
    } else {
      result.push(arr[i]);
    }
  }
  return result;
}
const nestedArray = [1, [2, 3], [4, [5, 6]], 7];
const flattenedArray = flatten(nestedArray);
console.log(flattenedArray);
```



```
> (7) [1, 2, 3, 4, 5, 6, 7]
```

Task 5:

```
function towerOfHanoi(n, source, destination, auxiliary, moves) {
            if (n === 1) {
                moves.push(`Move disk 1 from ${source} to
${destination}`);
                return;
            }
            towerOfHanoi(n - 1, source, auxiliary, destination, moves);
            moves.push(`Move disk ${n} from ${source} to
${destination}`);
            towerOfHanoi(n - 1, auxiliary, destination, source, moves);
        }
        function solveHanoi() {
            const numDisks =
parseInt(document.getElementById("diskInput").value);
            if (isNaN(numDisks) || numDisks <= 0) {
                document.getElementById("result").textContent = "Please
enter a valid positive number of disks.";
                return;
            }
            let moves = [];
            towerOfHanoi(numDisks, 'A', 'C', 'B', moves);
            document.getElementById("result").textContent =
moves.join("\n");
        }
```



Enter the number of disks to solve the Tower of Hanoi puzzle:

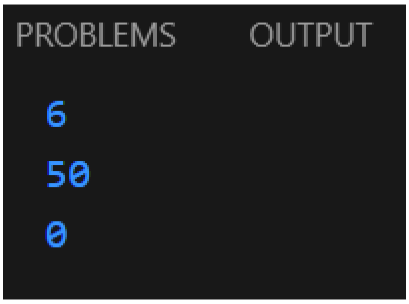
2

Move disk 1 from A to B Move disk 2 from A to C Move disk 1 from B to C

2. JSON and variable length arguments/spread syntax:

Task 1:

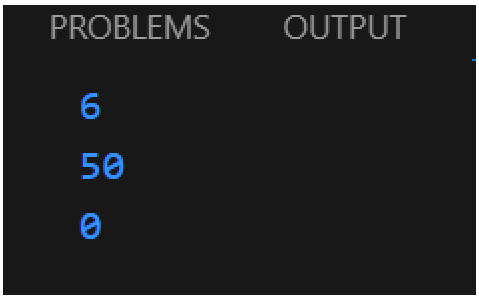
```
 function sum(...args) {
            return args.reduce((acc, curr) => acc + curr, 0);
        }
        console.log(sum(1, 2, 3));
        console.log(sum(5, 10, 15, 20));
        console.log(sum(1, -1, 2, -2));
```



```
6
50
0
```

Task 2:

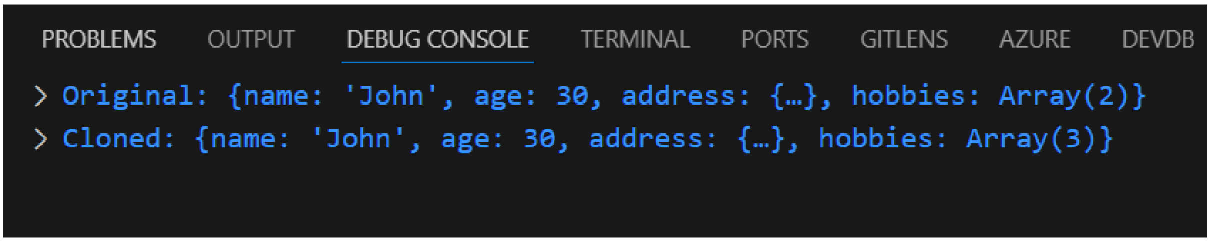
```
 function sumArray(...numbers) {
            return numbers.reduce((acc, curr) => acc + curr, 0);
        }
        const arr1 = [1, 2, 3];
        console.log(sumArray(...arr1));
        const arr2 = [5, 10, 15, 20];
        console.log(sumArray(...arr2));
        const arr3 = [1, -1, 2, -2];
        console.log(sumArray(...arr3));
```



```
6
50
0
```

Task 3:

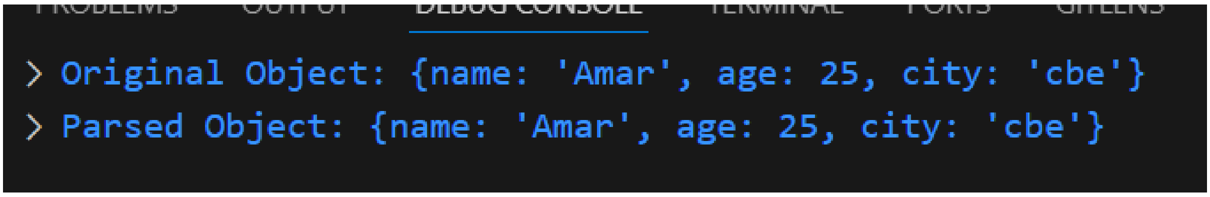
```
function deepClone(obj) {
        return JSON.parse(JSON.stringify(obj));
    }
    const original = {
        name: "John",
        age: 30,
        address: {
            city: "New York",
            country: "USA"
        },
        hobbies: ["reading", "traveling"]
    };
    const cloned = deepClone(original);
    cloned.address.city = "Los Angeles";
    cloned.hobbies.push("coding");
    console.log("Original:", original);
    console.log("Cloned:", cloned);
```



```
> Original: {name: 'John', age: 30, address: {…}, hobbies: Array(2)}
> Cloned: {name: 'John', age: 30, address: {…}, hobbies: Array(3)}
```

Task 4:

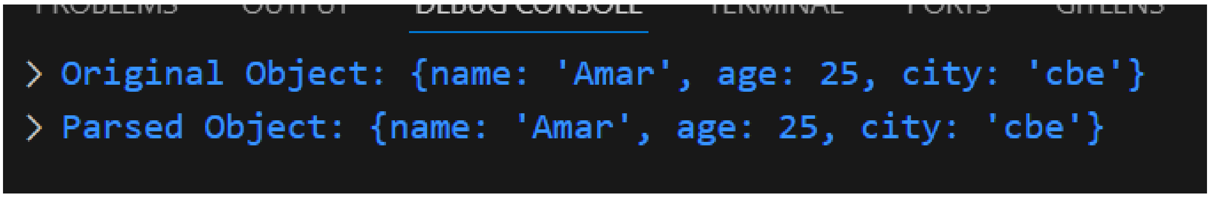
```
function mergeObjects(obj1, obj2) {
        return { ...obj1, ...obj2 };
    }
    const object1 = { name: "John", age: 30 };
    const object2 = { city: "New York", country: "USA" };
    const mergedObject = mergeObjects(object1, object2);
    console.log(mergedObject);
```



```
> Original Object: {name: 'Amar', age: 25, city: 'cbe'}
> Parsed Object: {name: 'Amar', age: 25, city: 'cbe'}
```

Task 5:

```
function serializeAndParse(obj) {

    const jsonString = JSON.stringify(obj);
    const parsedObject = JSON.parse(jsonString);

    return parsedObject;
}
const person = {
    name: "Alice",
    age: 25,
    city: "London"
};
const newPerson = serializeAndParse(person);
console.log("Original Object:", person);
console.log("Parsed Object:", newPerson);
```



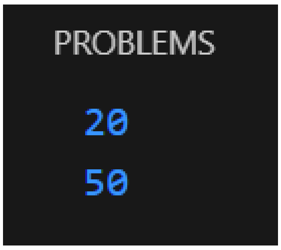
```
> Original Object: {name: 'Amar', age: 25, city: 'cbe'}
> Parsed Object: {name: 'Amar', age: 25, city: 'cbe'}
```

3. Closure:

Task 1:

```
        function createMultiplier(factor) {
```

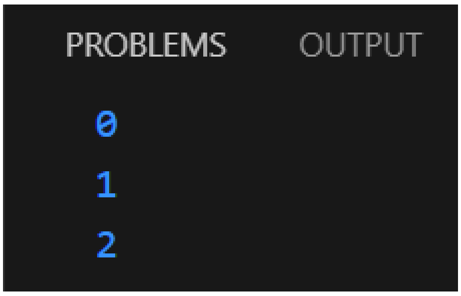
```
      return function(number) {
        return number * factor;
      };
    }
    const multiplyBy2 = createMultiplier(2);
    const multiplyBy5 = createMultiplier(5);

    console.log(multiplyBy2(10));
    console.log(multiplyBy5(10));
```



```
20
50
```

Task 2:

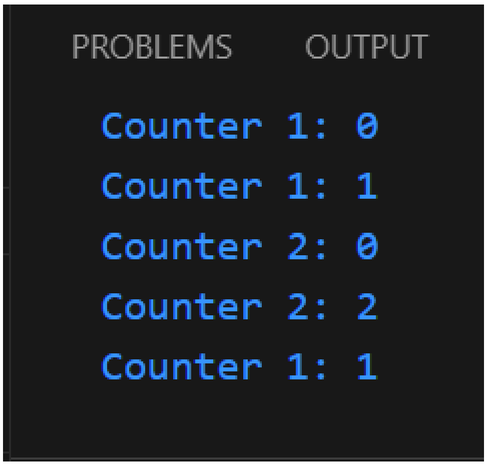
```
function createCounter() {
    let count = 0;
    return {
      increment: function() {
        count++;
      },
      getCount: function() {
        return count;
      }
    };
  }

  const counter = createCounter();
  console.log(counter.getCount());
  counter.increment();
  console.log(counter.getCount());
  counter.increment();
  console.log(counter.getCount());
```



```
0
1
2
```

Task 3:

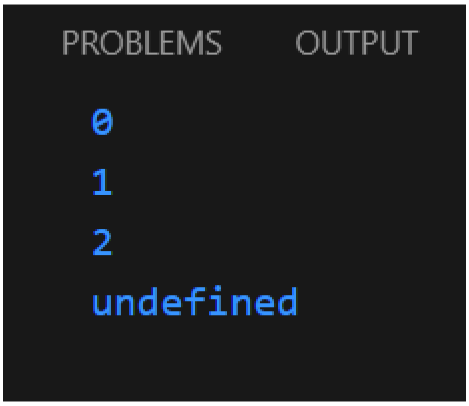
```
function createCounter() {
        let count = 0;
        return {
           increment: function() {
              count++;
           },
           getCount: function() {
              return count;
           }
        };
     }
     const counter1 = createCounter();
     const counter2 = createCounter();
     console.log("Counter 1:", counter1.getCount());
     counter1.increment();
     console.log("Counter 1:", counter1.getCount());
     console.log("Counter 2:", counter2.getCount());
     counter2.increment();
     counter2.increment();
     console.log("Counter 2:", counter2.getCount());
     console.log("Counter 1:", counter1.getCount());
```



```
Counter 1: 0
Counter 1: 1
Counter 2: 0
Counter 2: 2
Counter 1: 1
```

Task 4:

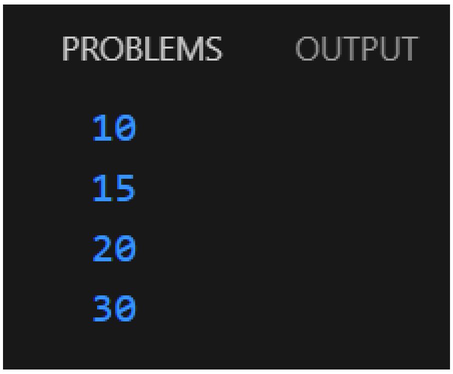
```
  function createCounter() {
          let count = 0;
          return {
              increment: function() {
                  count++;
              },
              getCount: function() {
                  return count;
              }
          };
      }
      const counter = createCounter();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



```
0
1
2
undefined
```

Task 5:

```
function createMultiplier(multiplier) {
            return function(number) {
                return number * multiplier;
            };
        }
        const double = createMultiplier(2);
        const triple = createMultiplier(3);
        console.log(double(5));
        console.log(triple(5));
        console.log(double(10));
        console.log(triple(10));
```

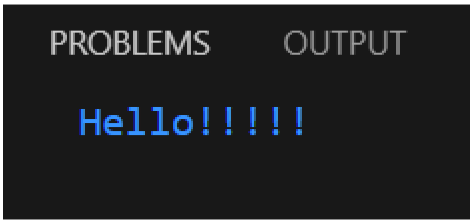


```
10
15
20
30
```

4. Promise, Promises chaining:

Task 1:

```
function delayedGreeting(seconds) {
          return new Promise((resolve, reject) => {
             if (seconds < 0) {
                reject("Time cannot be negative!");
             } else {
                setTimeout(() => {
                   resolve("Hello! This is your greeting.");
                }, seconds * 1000);
             }
          });
       }
       delayedGreeting(3)
          .then((greeting) => {
             console.log(greeting);
          })
          .catch((error) => {
             console.error(error);
          });
```



```
Hello!!!!!
```

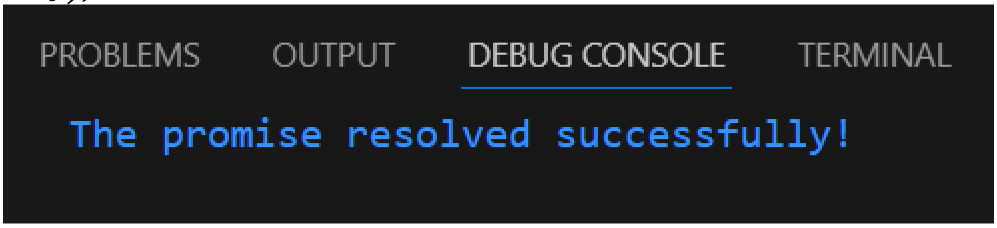
Task 2:

```
function fetchData() {
   fetch("https://jsonplaceholder.typicode.com/posts")
      .then(function (response) {
         return response.json();
      })
      .then(function (data) {
         const titles = data.map(function (post) {
            return post.title;
         });
         console.log("Here are the post titles:");
         console.log(titles);
```

```
        })
        .catch(function (error) {
            console.error("Something went wrong!", error);
        });
}

fetchData();
```

Task 3:

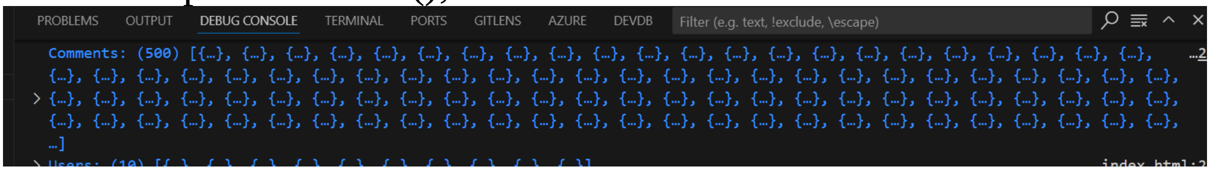
```
function randomPromise() {
    return new Promise(function (resolve, reject) {
        const randomNumber = Math.random();
        if (randomNumber > 0.5) {
            resolve("The promise resolved successfully!");
        } else {
            reject("The promise was rejected!");
        }
    });
}

randomPromise()
    .then(function (message) {
        console.log(message);
    })
    .catch(function (error) {
        console.error(error);
    });
```



```
The promise resolved successfully!
```

Task 4:

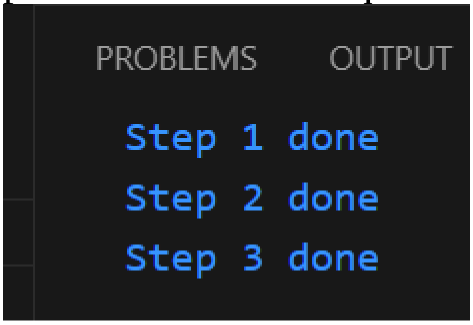
```
function fetchMultipleResources() {
    const url1 = "https://jsonplaceholder.typicode.com/posts";
    const url2 = "https://jsonplaceholder.typicode.com/comments";
    const url3 = "https://jsonplaceholder.typicode.com/users";

    Promise.all([fetch(url1), fetch(url2), fetch(url3)])
        .then(function (responses) {
            return Promise.all(responses.map(function (response) {
                return response.json();
            }));
        })
        .then(function (data) {
            console.log("Posts:", data[0]);
            console.log("Comments:", data[1]);
            console.log("Users:", data[2]);
        })
        .catch(function (error) {
            console.error("Error fetching resources:", error);
        });
}

fetchMultipleResources();
```



```
  Comments: (500) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},     …2
  {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
> {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
  …]
> Users: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]                                                        index.html:2
```

Task 5:

```
function performActionsInSequence() {
    const step1 = new Promise(function (resolve) {
        setTimeout(function () {
            resolve("Step 1 done");
        }, 1000);
    });

    step1
        .then(function (result1) {
            console.log(result1);
            return new Promise(function (resolve) {
                setTimeout(function () {
                    resolve("Step 2 done");
                }, 1000);
            });
        })
        .then(function (result2) {
            console.log(result2);
            return new Promise(function (resolve) {
                setTimeout(function () {
                    resolve("Step 3 done");
                }, 1000);
            });
        })
        .then(function (result3) {
            console.log(result3);
        });
}

performActionsInSequence();
```

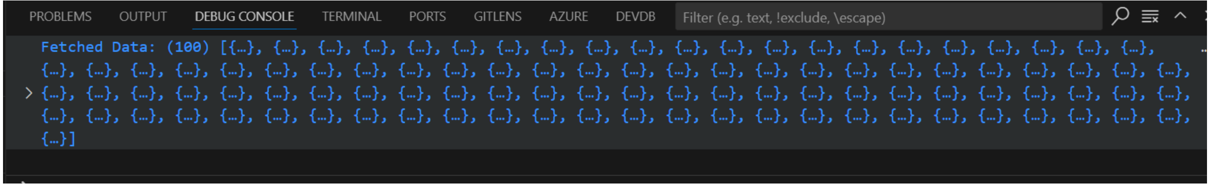


```
Step 1 done
Step 2 done
Step 3 done
```

5. Async/await:

Task 1:

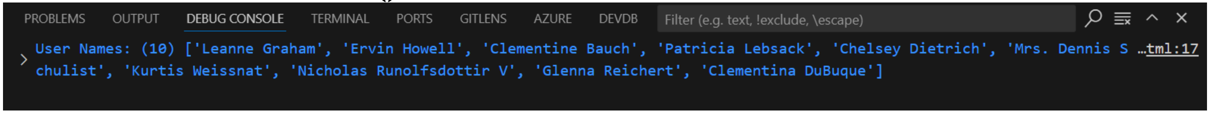
```
async function fetchData() {
    try {
        const response = await
fetch("https://jsonplaceholder.typicode.com/posts");
        const data = await response.json();
        console.log("Fetched Data:", data);
    } catch (error) {
        console.error("Error fetching data:", error);
    }
}

fetchData();
```



```
Fetched Data: (100) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}]
```

Task 2:

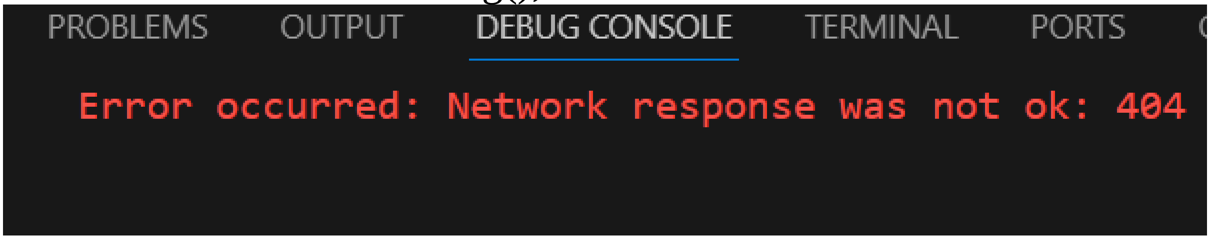
```
async function fetchAndProcessData() {
    try {
        const response = await
fetch("https://jsonplaceholder.typicode.com/users");
        const data = await response.json();
        const userNames = data.map(user => user.name);
        console.log("User Names:", userNames);
    } catch (error) {
        console.error("Error fetching data:", error);
    }
}

fetchAndProcessData();
```



```
User Names: (10) ['Leanne Graham', 'Ervin Howell', 'Clementine Bauch', 'Patricia Lebsack', 'Chelsey Dietrich', 'Mrs. Dennis S …tml:17
chulist', 'Kurtis Weissnat', 'Nicholas Runolfsdottir V', 'Glenna Reichert', 'Clementina DuBuque']
```

Task 3:

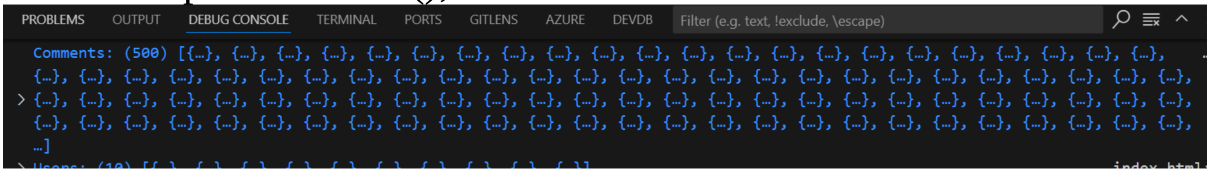
```
async function fetchDataWithErrorHandling() {
    try {
        const response = await
fetch("https://jsonplaceholder.typicode.com/invalid-url");

        if (!response.ok) {
            throw new Error("Network response was not ok: " +
response.status);
        }

        const data = await response.json();
        console.log("Fetched Data:", data);
    } catch (error) {
        console.error("Error occurred:", error.message);
    }
}

fetchDataWithErrorHandling();
```



```
Error occurred: Network response was not ok: 404
```

Task 4:

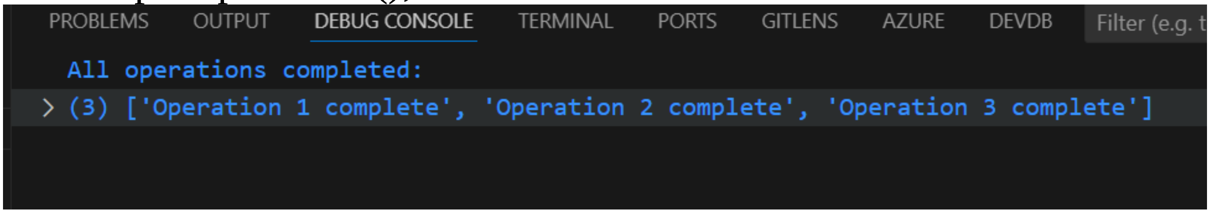
```
async function fetchMultipleResources() {
    try {
        const url1 = "https://jsonplaceholder.typicode.com/posts";
        const url2 = "https://jsonplaceholder.typicode.com/comments";
        const url3 = "https://jsonplaceholder.typicode.com/users";

        const responses = await Promise.all([
            fetch(url1),
            fetch(url2),
            fetch(url3)
        ]);

        const data = await Promise.all(responses.map(response =>
response.json()));

        console.log("Posts:", data[0]);
        console.log("Comments:", data[1]);
        console.log("Users:", data[2]);
    } catch (error) {
        console.error("Error occurred:", error.message);
    }
}

fetchMultipleResources();
```



```
Comments: (500) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
…]
Users: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]                                                  index.html:
```

Task 5:

```
async function waitForMultipleOperations() {
    try {
        const operation1 = new Promise(resolve => setTimeout(() =>
         resolve("Operation 1 complete"), 1000));
        const operation2 = new Promise(resolve => setTimeout(() =>
         resolve("Operation 2 complete"), 2000));
        const operation3 = new Promise(resolve => setTimeout(() =>
         resolve("Operation 3 complete"), 1500));

        const results = await Promise.all([operation1, operation2,
         operation3]);

        console.log("All operations completed:");
        console.log(results);
    } catch (error) {
        console.error("Error occurred:", error);
    }
}

waitForMultipleOperations();
```

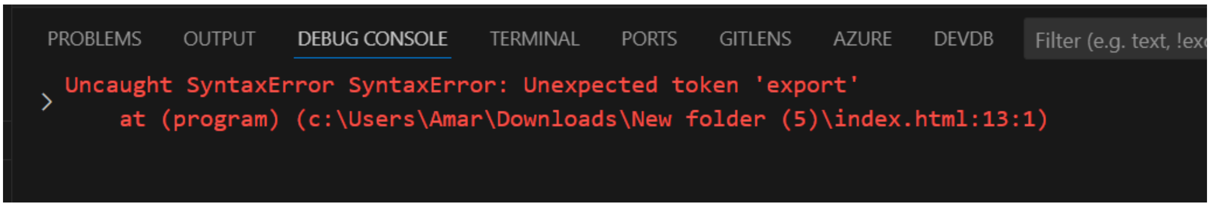


```
All operations completed:
> (3) ['Operation 1 complete', 'Operation 2 complete', 'Operation 3 complete']
```

6. Modules introduction, Export and Import:

Task 1:

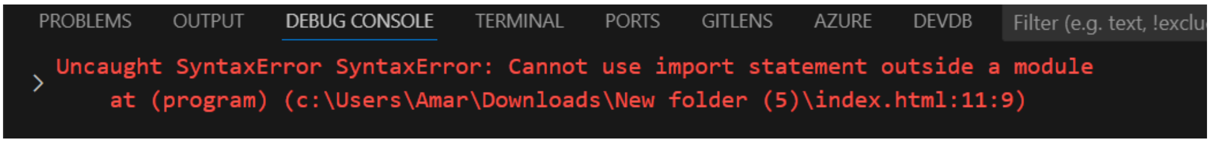
```
export function greet(name) {
   return `Hello, ${name}!`;
}

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

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

export const country = "USA";

import { greet, Person, country } from './module1.js';

console.log(greet("Alice"));

const person1 = new Person("Bob", 25);
console.log(person1.introduce());

console.log("I live in:", country);
```



```
Uncaught SyntaxError SyntaxError: Unexpected token 'export'
    at (program) (c:\Users\Amar\Downloads\New folder (5)\index.html:13:1)
```

Task 2:

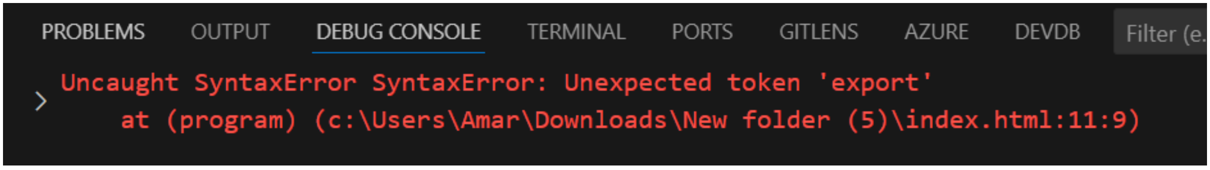
```
 import { greet, Person, country } from './module1.js';
      console.log(greet("Alice"));
      const person1 = new Person("Bob", 25);
      console.log(person1.introduce());
      console.log("I live in:", country);
```



```
> Uncaught SyntaxError SyntaxError: Cannot use import statement outside a module
      at (program) (c:\Users\Amar\Downloads\New folder (5)\index.html:11:9)
```

Task 3:

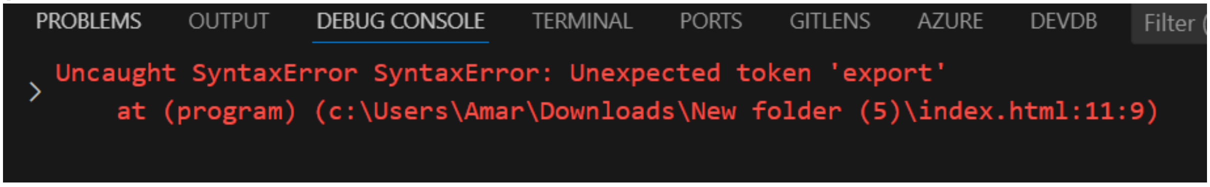
```
export function add(a, b) {
    return a + b;
}

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;
}
```



```
> Uncaught SyntaxError SyntaxError: Unexpected token 'export'
      at (program) (c:\Users\Amar\Downloads\New folder (5)\index.html:11:9)
```

Task 4:

```
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;
}
```



```
Uncaught SyntaxError SyntaxError: Unexpected token 'export'
    at (program) (c:\Users\Amar\Downloads\New folder (5)\index.html:11:9)
```

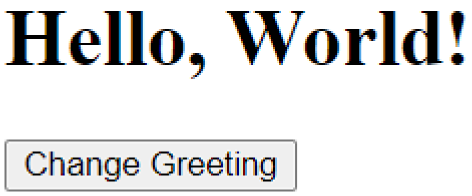
Task 5:

```
        export default function greet(name) {
            return `Hello, ${name}!`;
        }
    </script>

    <script type="module">

        import greet from './index.html';

        console.log(greet("Amar"));
```

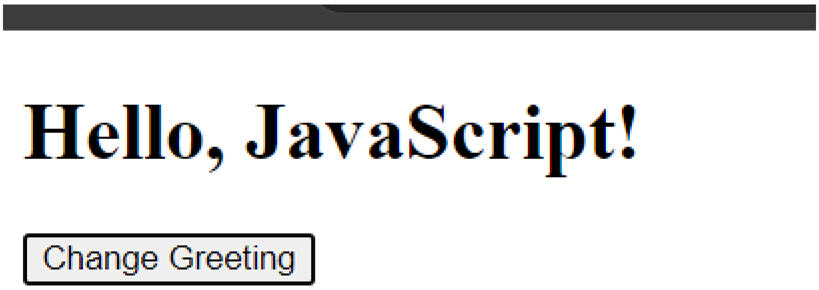
7. Browser: DOM Basics:

Task 1:

```
function changeContent() { var element =
        document.getElementById("greeting"); element.textContent =
        "Hello, JavaScript!"; }
```



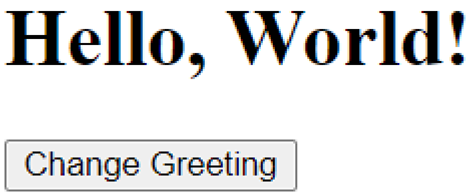
Hello, World!



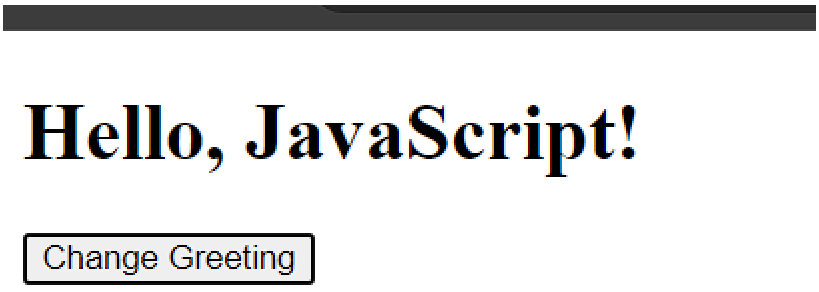
Hello, JavaScript!

Task 2:

```
function changeContent() { var element =
        document.getElementById("greeting"); element.textContent =
        "Hello, JavaScript!"; }
```



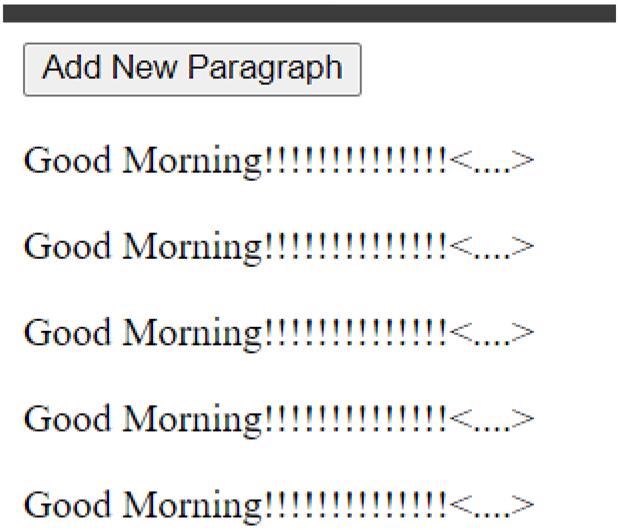
Hello, World!



Hello, JavaScript!

Task 3:

```
function addNewElement() {
        var newParagraph = document.createElement("p");
        newParagraph.textContent = "Good
      Morning!!!!!!!!!!!!!!<....>";
        document.body.appendChild(newParagraph);
     }
```



Good Morning!!!!!!!!!!!!!!<....>

Good Morning!!!!!!!!!!!!!!<....>

Good Morning!!!!!!!!!!!!!!<....>

Good Morning!!!!!!!!!!!!!!<....>

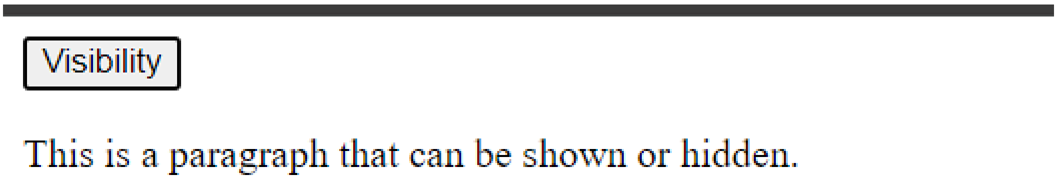
Good Morning!!!!!!!!!!!!!!<....>

Task 4:

```
function Visibility() {
        var element = document.getElementById("myParagraph");
        if (element.style.display === "none") {
           element.style.display = "block";
        } else {
           element.style.display = "none";
        }
```

}**OUTPUT 1:**

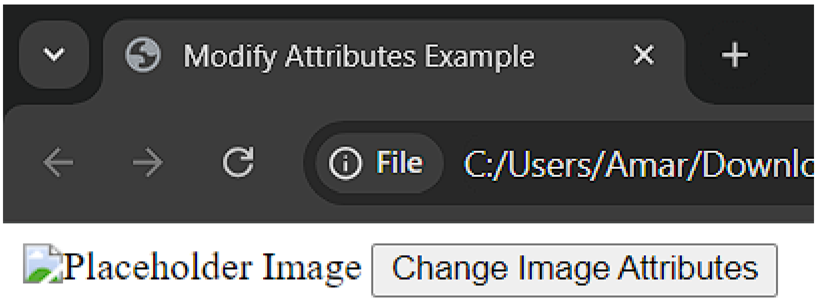
OUTPUT 2:



This is a paragraph that can be shown or hidden.

Task 5:

```
 function modifyAttributes() {
          var imageElement = document.getElementById("myImage");
          console.log("Current alt attribute:",
imageElement.getAttribute("alt"));
          imageElement.setAttribute("src", "bg2.jpg");
          imageElement.setAttribute("alt", "New Placeholder Image");
          console.log("Updated src attribute:",
imageElement.getAttribute("src"));
          console.log("Updated alt attribute:",
imageElement.getAttribute("alt"));
       }
```


Modify Attributes Example
File
C:/Users/Amar/Downlo
Placeholder Image
Change Image Attributes