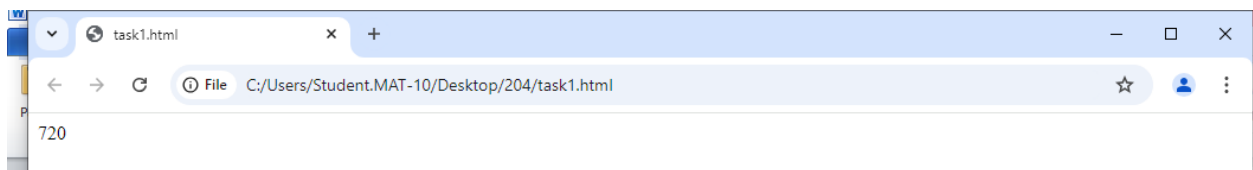
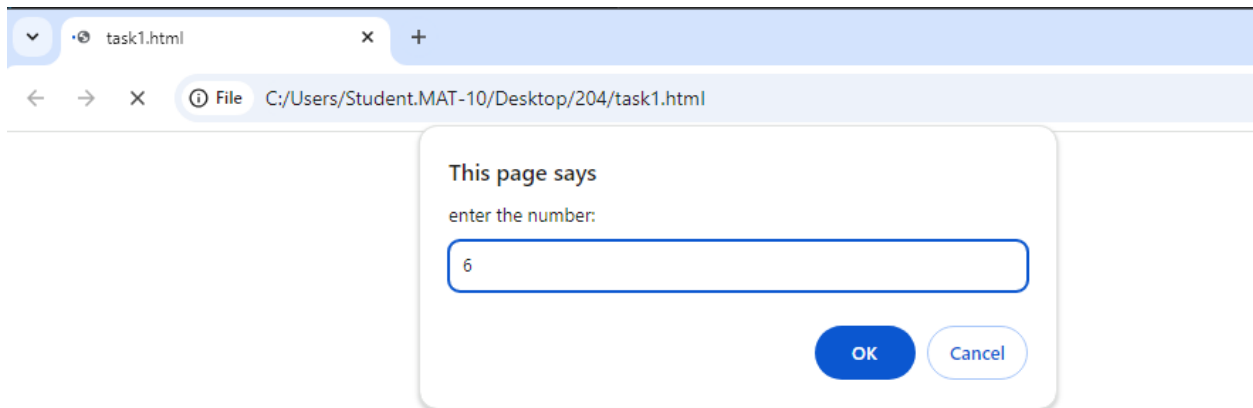


TASK 1:

```
<html>
  <body>
    <script>
      function factorial(n)
      {
        if(n==0)
          return 1;
        else
          return n*factorial(n-1);
      }
      let a=parseInt(prompt("enter the number:"));
      document.write(factorial(a));
    </script>
  </body>
</html>
```

OUTPUT:



TASK 2:

```
<html>
  <body>
    <script>
      function fibonnacci(n){
        if(n<=1)
```

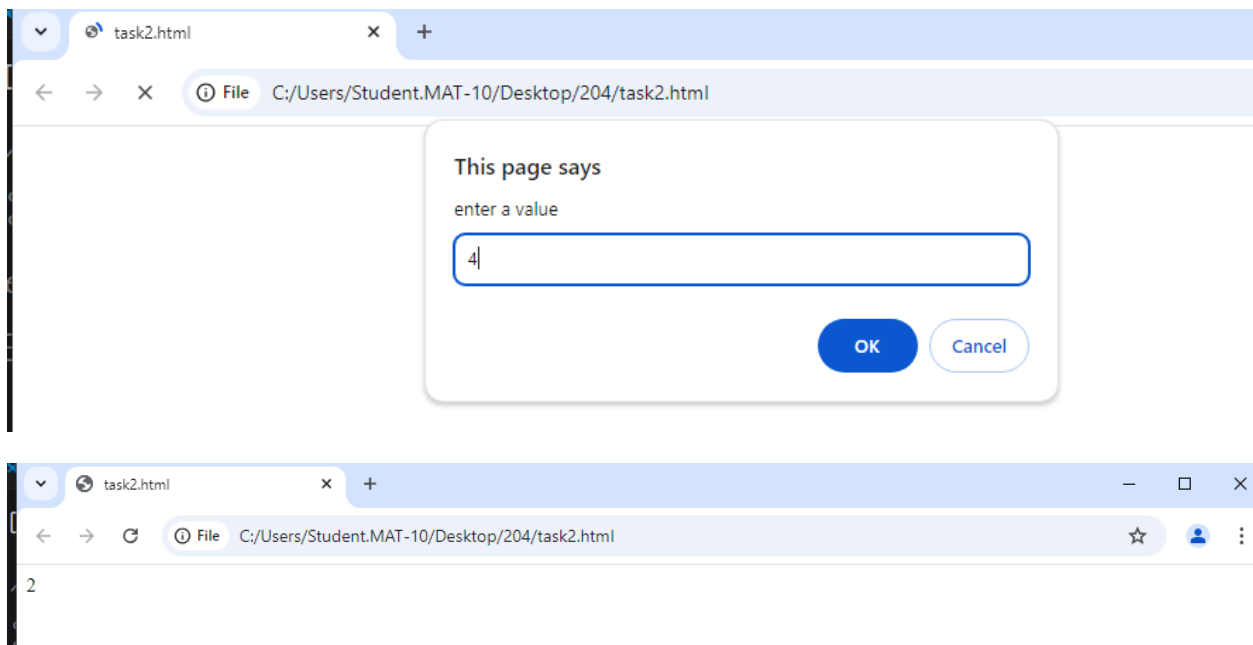
```

        return n;
        return fibonacci(n-1)+fibonacci(n-2);
    }
    let a=parseInt(prompt("enter a value"));
    document.write(fibonacci(a));

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

```

OUTPUT:



TASK 3:

```

<html>
  <body>
    <script>
      function countWays(n) {
        if (n === 0) {
          return 1;
        } else if (n === 1) {
          return 1;
        } else if (n === 2) {
          return 2;
        } else if (n === 3) {
          return 4;
        }
      }
    </script>
  </body>
</html>

```

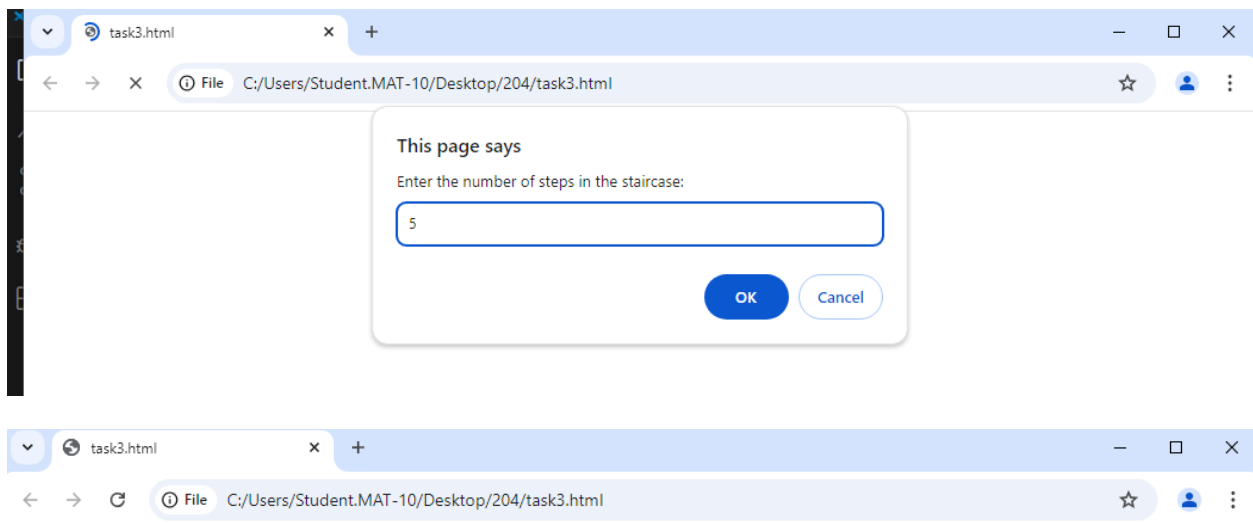
```

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

let n = parseInt(prompt("Enter the number of steps in the
staircase:"));
document.write("Total number of ways " + countWays(n));
</script>
</body>
</html>

```

OUTPUT:



Total number of ways 13

TASK 4:

```

<html>
  <body>
    <script>
      function flattenArray(arr) {
        var result = [];
        for (var i = 0; i < arr.length; i++) {
          var item = arr[i];
          if (Array.isArray(item)) {
            result = result.concat(flattenArray(item));
          } else {
            result.push(item);
          }
        }
        return result;
      }
    </script>
  </body>
</html>

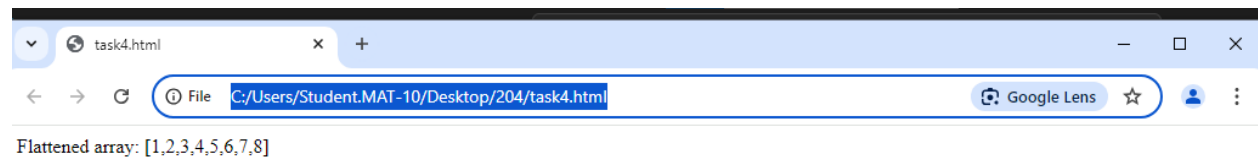
```

```

    var nestedArray = [1, [2, 3, [4, 5]], [6, 7], 8];
    var flattened = flattenArray(nestedArray);
    document.write("Flattened array: " + JSON.stringify(flattened));
  </script>
</body>
</html>

```

OUTPUT:



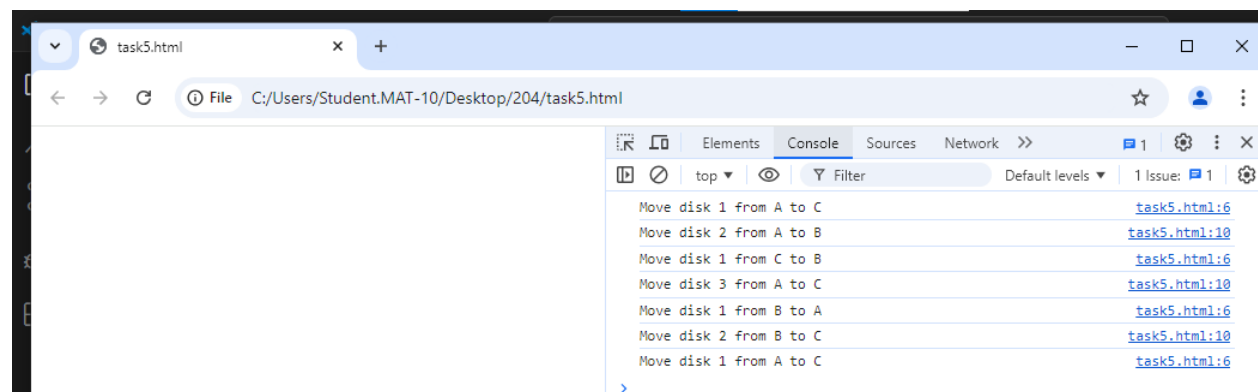
TASK 5:

```

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

```

OUTPUT:



JSON VARIABLE LENGTH ARGUMENTS/SPREAD SYNTAX

TASK-1

```
<html>

<body>

<script>

function sum(...arguments)

{

let total = 0;

for(let num of arguments)

{

total+=num;

}

return total;

}

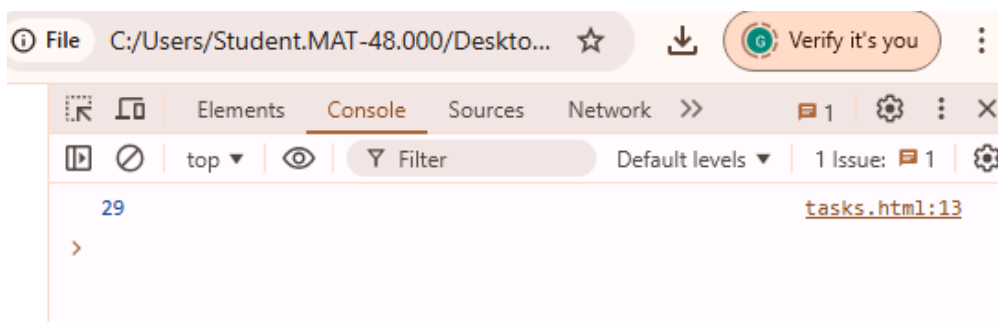
console.log(sum(9,8,5,7));

</script>

</body>

</html>
```

Output:



Task 2:

```
<html>

<body>

<script>

function sum(...arguments)

{

let total = 0;

for(let num of arguments)

{

total+=num;

}

return total;

}

const arr = [7,6,8,3]

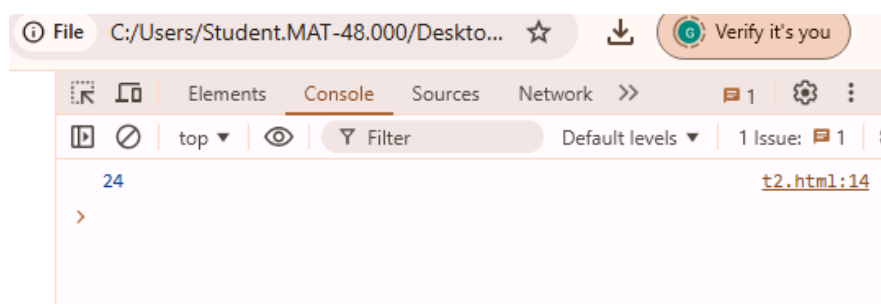
console.log(sum(...arr));

</script>

</body>

</html>
```

OUTPUT:



TASK-3

```
<html>

<body>

<script>

function deepclone(obj)

{

return JSON.parse(JSON.stringify(obj));

}

const originalobj = {name:"John",details:{age: "45",Dept : "IT"}};

const cloneobj = deepclone(originalobj);

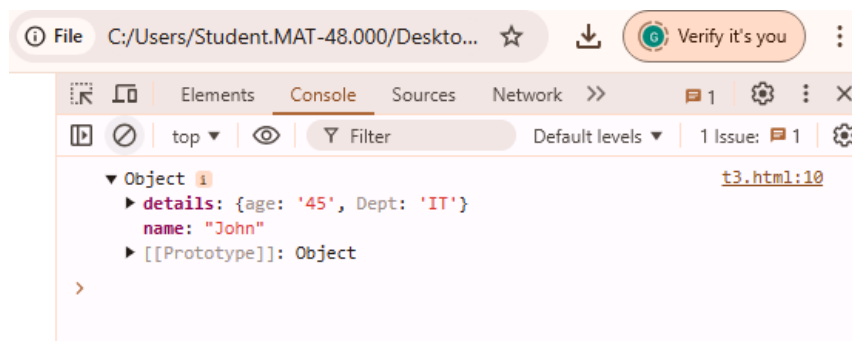
console.log(cloneobj);

</script>

</body>

</html>
```

OUTPUT:



TASK-4

```
<html>

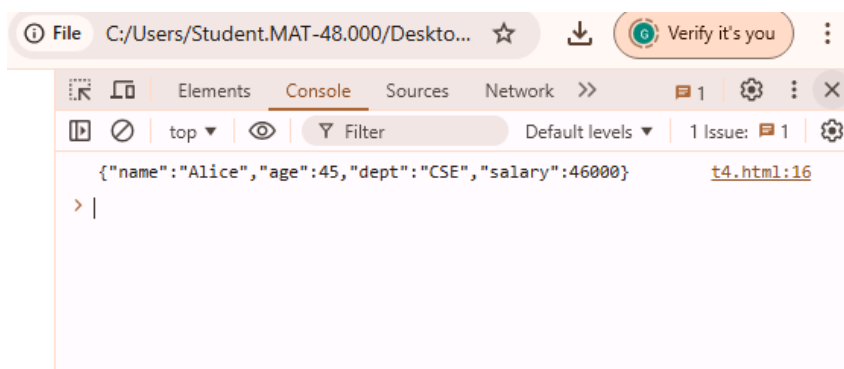
<body>

<script>

let student =
```

```
{  
  name:"Alice",  
  age:45,  
  dept:"CSE"  
};  
  
let Employee = {  
  name:"Alice",  
  age:45,  
  salary:46000  
};  
  
let obj = {...student,...Employee};  
  
console.log(JSON.stringify(obj));  
  
</script>  
  
</body>  
  
</html>
```

OUTPUT:



TASK-5

<html>

<body>


```
<script>

const obj = {name:"Sam",Salary:35000,Dept:"CSE",Age:21};

let jsonString = JSON.stringify(obj);

console.log(jsonString);

let parseobj = JSON.parse(jsonString);

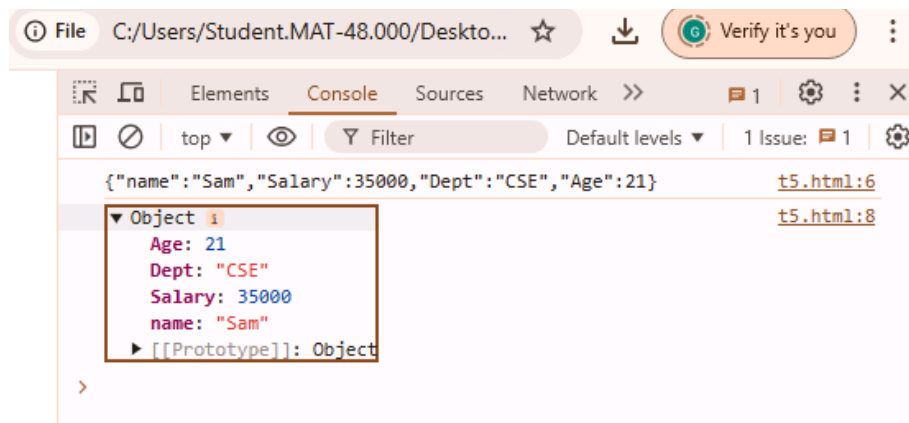
console.log(parseobj);

</script>

</body>

</html>
```

OUTPUT:



ASYNCR/AWAIT

TASK 1:

<html>

<body>

<script>

```
    async function getUserData(userId) {  
const users = {  
  1: { name: 'Alice', age: 25 },  
  2: { name: 'Bob', age: 30 },  
};  
  
const getUser = new Promise((resolve, reject) => {  
  setTimeout(() => {  
    if (users[userId]) {  
      resolve(users[userId]);  
    } else {  
      reject('User not found');  
    }  
  }, 1000);  
});  
  
try {  
  const user = await getUser;  
  console.log('User data:', user);  
} catch (error) {
```

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

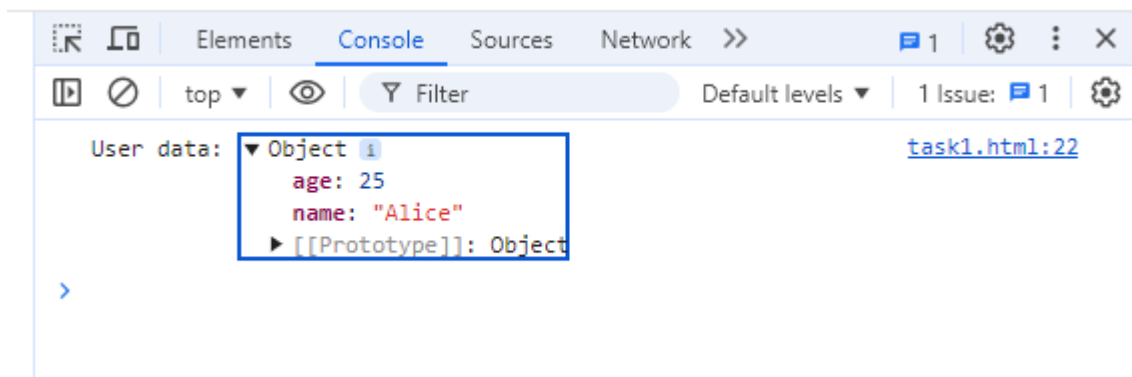
```
getUserData(1);
```

```
    </script>
```

```
    </body>
```

```
</html>
```

OUTPUT:



TASK 2:

```
<html>
```

```
    <body>
```

```
        <script>
```

```
async function fetchAndProcessData() {
```

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

```
    try {
```

```
        const response = await fetch(apiUrl);
```

```
        if (!response.ok) {
```

```
            throw new Error(`HTTP error! Status: ${response.status}`);
```

```
}  
  
const data = await response.json();  
console.log('Processing fetched data...');  
data.forEach(user => {  
    console.log(`User Name: ${user.name}`);  
});  
  
} catch (error) {  
    console.error('Error fetching or processing data:', error);  
}  
}  
  
fetchAndProcessData();
```

</script>

</body>

</html>

OUTPUT:



TASK 3:

```
<html>
```

```
<body>
```

```
<script>
```

```
  async function getUserInfo(userId) {
```

```
    try {
```

```
      const response = await fetch(`https://jsonplaceholder.typicode.com/users/1`);
```

```
      if (!response.ok) {
```

```
        throw new Error(`HTTP error! Status: ${response.status}`);
```

```
      }
```

```
      const user = await response.json();
```

```
      if (!user || !user.name || !user.email) {
```

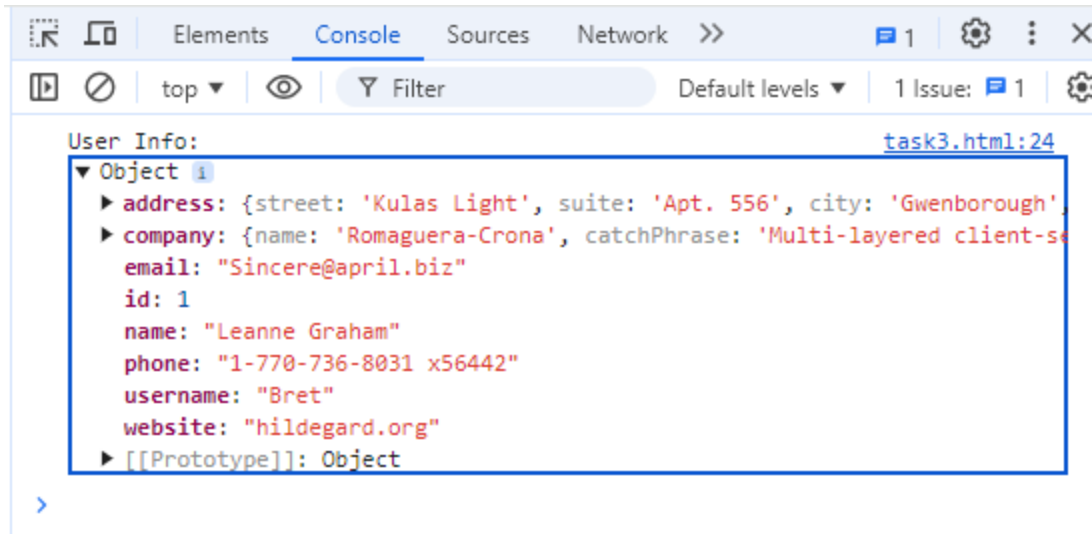
```
        throw new Error('Invalid user data received');
```

```
      }
```

```
    return user;
  } catch (error) {
    console.error('Error getting user info:', error.message);
    return null;
  }
}

getUserInfo(1)
  .then(user => {
    if (user) {
      console.log('User Info:', user);
    } else {
      console.log('Failed to get user info');
    }
  })
  .catch(error => {
    console.log('Unexpected error:', error);
  });
</script>
</body>
</html>
```

OUTPUT:



TASK 4:

```
<html>
```

```
  <body>
```

```
    <script>
```

```
    async function fetchUser(userId) {
```

```
      const response = await fetch(`https://jsonplaceholder.typicode.com/users/${userId}`);
```

```
      if (!response.ok) throw new Error(`User with ID ${userId} not found`);
```

```
      return response.json();
```

```
    }
```

```
    async function fetchUserPosts(userId) {
```

```
      const response = await fetch(`https://jsonplaceholder.typicode.com/posts?userId=${userId}`);
```

```
      if (!response.ok) throw new Error(`Posts for user with ID ${userId} not found`);
```

```
      return response.json();
```

```
    }
```

```
    async function fetchUserAndPosts(userId) {
```

```
      try {
```

```
        const [user, posts] = await Promise.all([
```

```
          fetchUser(userId),
```

```
          fetchUserPosts(userId),
```

```
]);
```

```
console.log('User Info:', user);
```

```
console.log('User Posts:', posts);
```

```
} catch (error) {
```

```
console.error('Error:', error.message);
```

```
}
```

```
}
```

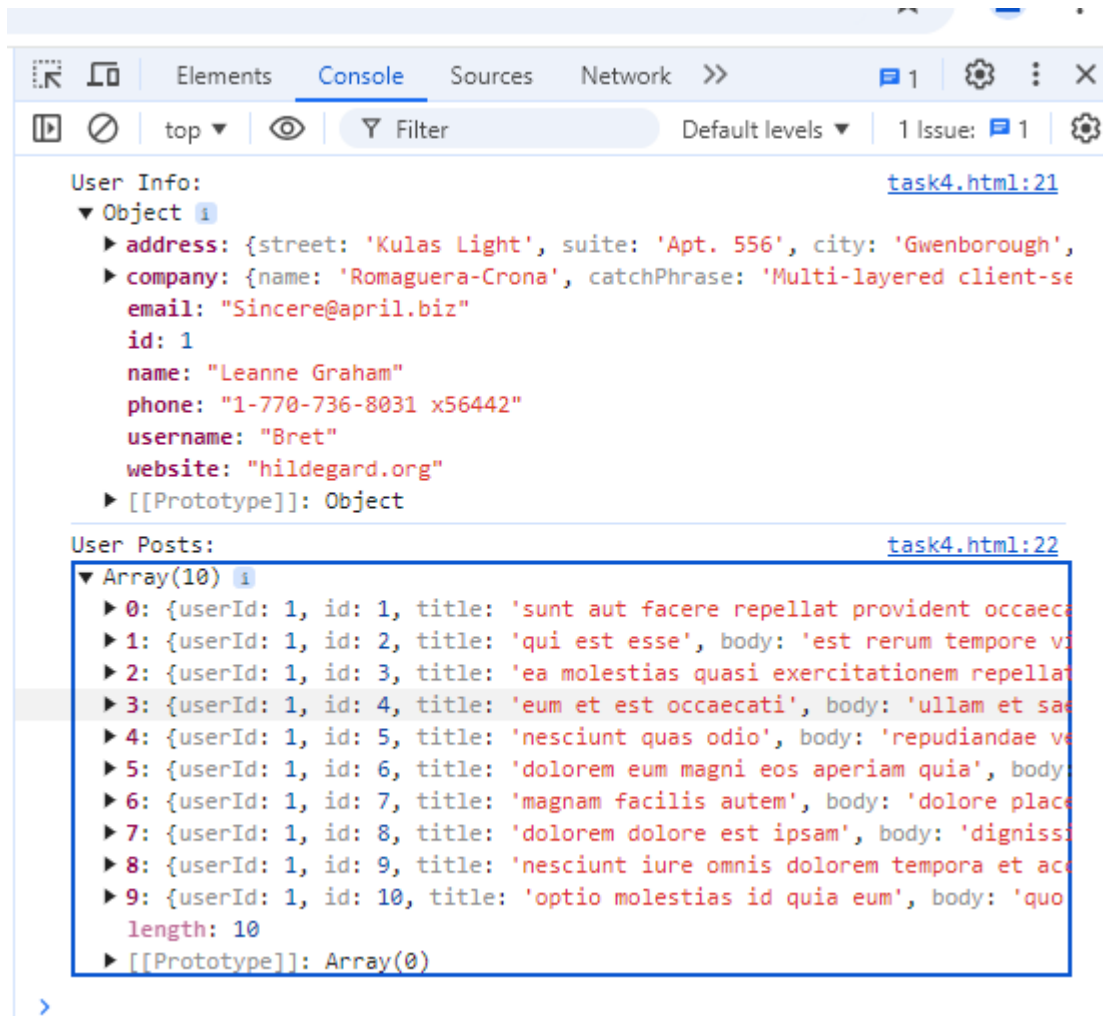
```
fetchUserAndPosts(1);
```

```
</script>
```

```
</body>
```

```
</html>
```

OUTPUT:



TASK 5:

```
<html>
```

```
<body>
```

```
<script>
```

```
async function fetchUser(userId) {
```

```
  const response = await new Promise((resolve) =>
```

```
    setTimeout(() => resolve({ id: userId, name: 'John Doe' }), 2000)
```

```
  );
```

```
  return response;
```

```
}
```

```
async function fetchUserPosts(userId) {
```

```

const response = await new Promise((resolve) =>
  setTimeout(() => resolve([
    { postId: 1, title: 'First Post' },
    { postId: 2, title: 'Second Post' }
  ]), 1500)
);
return response;
}

async function fetchUserComments(postId) {
  const response = await new Promise((resolve) =>
    setTimeout(() => resolve([
      { commentId: 1, body: 'Great post!' }
    ]), 1000)
  );
  return response;
}

async function fetchData(userId) {
  try {
    const [user, posts, comments] = await Promise.all([
      fetchUser(userId),
      fetchUserPosts(userId),
      fetchUserComments(1),
    ]);

    console.log('User Data:', user);
    console.log('User Posts:', posts);
    console.log('User Comments:', comments);

  } catch (error) {
    console.error('Error fetching data:', error.message);
  }
}

fetchData(1);

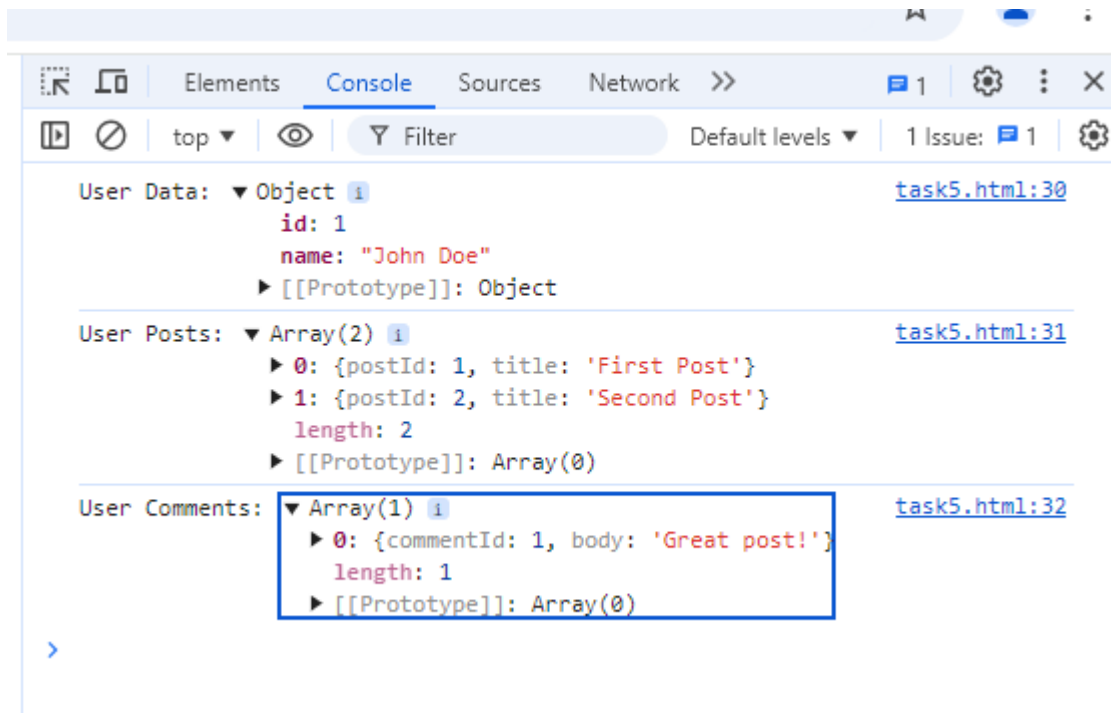
```

</script>

</body>

</html>

OUTPUT:



TASK1:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>Change Content</title>
```

```
</head>
```

```
<body>
```

```
  <p id="my">Hello, World!</p>
```

```
  <script>
```

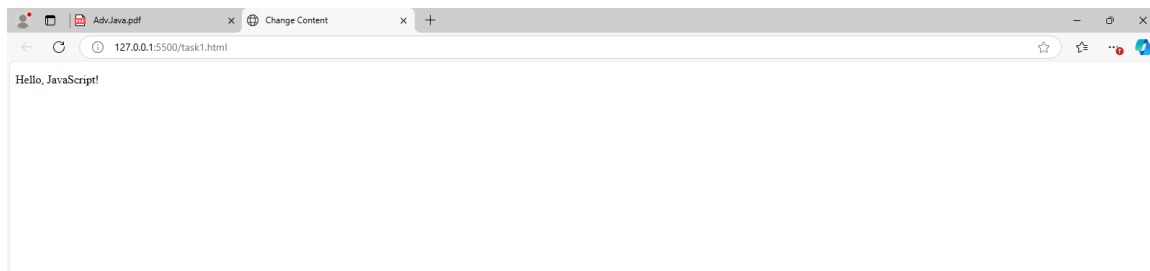
```
    document.getElementById("my").innerHTML = "Hello, JavaScript!";
```

```
  </script>
```

```
</body>
```

```
</html>
```

OUTPUT:



TASK2:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>Button Click Event</title>
```

```
</head>
```

```
<body>
```

```
  <button id="myButton">Click Me!</button>
```

```

<script>

    document.getElementById("myButton").addEventListener("click", function() {

        alert("Button was clicked!");

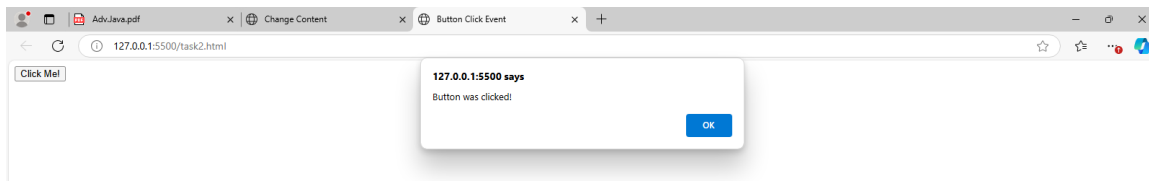
    });

</script>

</body>

</html>

```



TASK3:

```

<!DOCTYPE html>

<html>

<head>

    <title>Append Element</title>

</head>

<body>

    <div id="container"></div>

    <script>

        var newElement = document.createElement("p");

        newElement.innerHTML = "This is a new paragraph.";

        document.getElementById("container").appendChild(newElement);

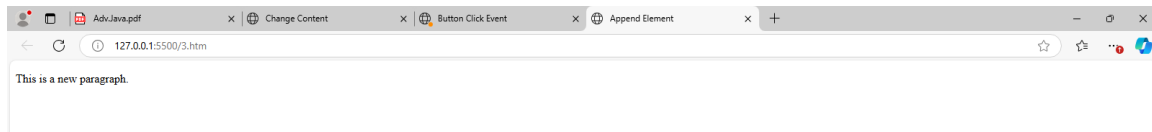
    </script>

</body>

</html>

```

OUTPUT:



TASK4:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>Toggle Visibility</title>
```

```
</head>
```

```
<body>
```

```
  <p id="toggleParagraph">Toggle my visibility!</p>
```

```
  <button id="toggleButton">Toggle</button>
```

```
  <script>
```

```
    document.getElementById("toggleButton").addEventListener("click", function() {
```

```
      var paragraph = document.getElementById("toggleParagraph");
```

```
      if (paragraph.style.display === "none") {
```

```
        paragraph.style.display = "block";
```

```
      } else {
```

```
        paragraph.style.display = "none";
```

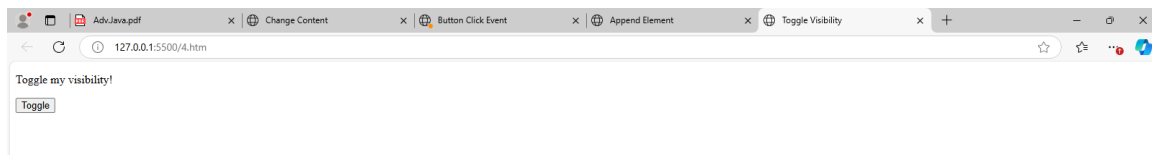
```
      }
```

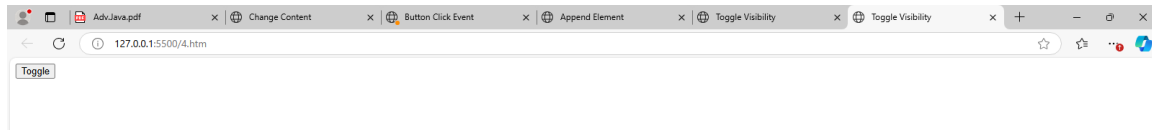
```
    });
```

```
  </script>
```

```
</body>
```

```
</html>
```





TASK5:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>Modify Attributes</title>
```

```
</head>
```

```
<body>
```

```
  
```

```
  <script>
```

```
    var image = document.getElementById("myImage");
```

```
    console.log(image.src);
```

```
    image.src = "newImage.jpg";
```

```
    console.log(image.alt);
```

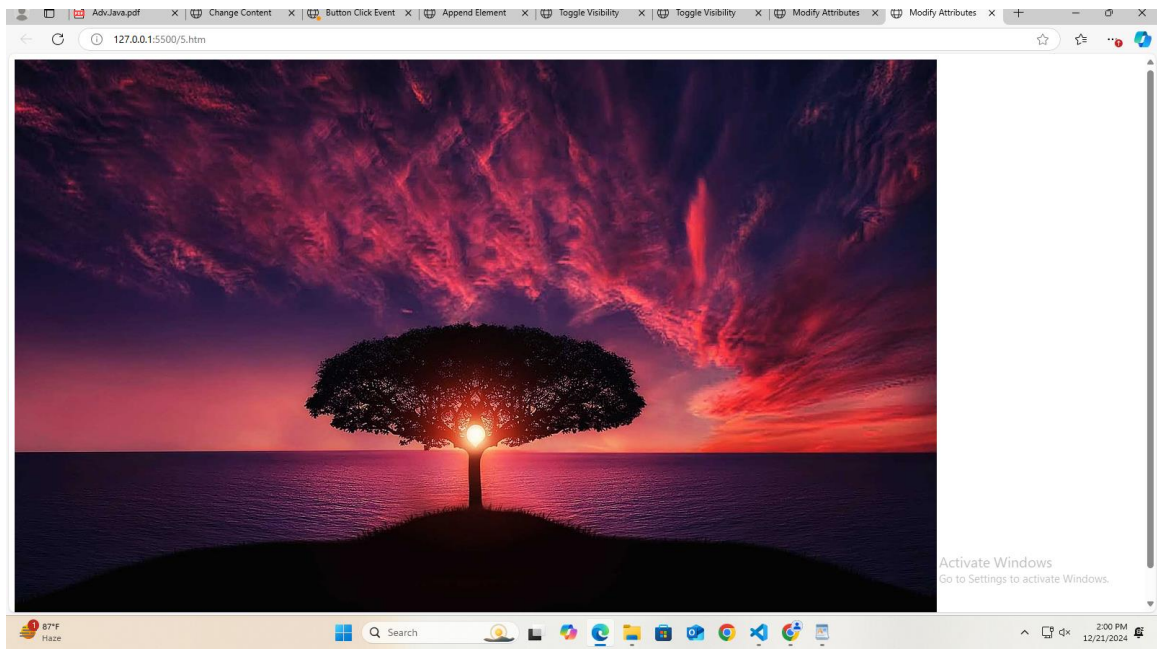
```
    image.alt = "Updated Image";
```

```
  </script>
```

```
</body>
```

```
</html>
```

OUTPUT:



CLOSURE:

TASK 1:

```
<html>

  <body>

    <script>

function outer() {

  let message = "Hello from the outer function!";


  return function inner() {

    console.log(message);

  }

}

const closureFunction = outer();

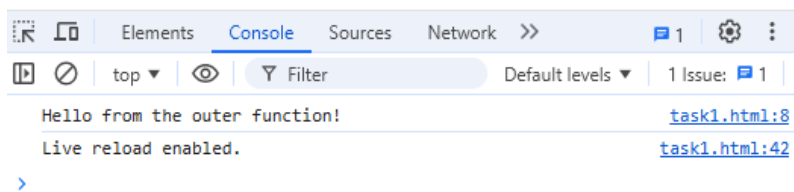
closureFunction();

</script>

</body>

</html>
```

OUTPUT:



TASK 2:

```
<html>

  <body>
```

```
<script>

function createCounter() {

  let count = 0;

  return {

    increment: function() {

      count++;

    },

    getCount: function() {

      console.log(count);

    }

  };

}

const counter = createCounter();

counter.increment();

counter.increment();

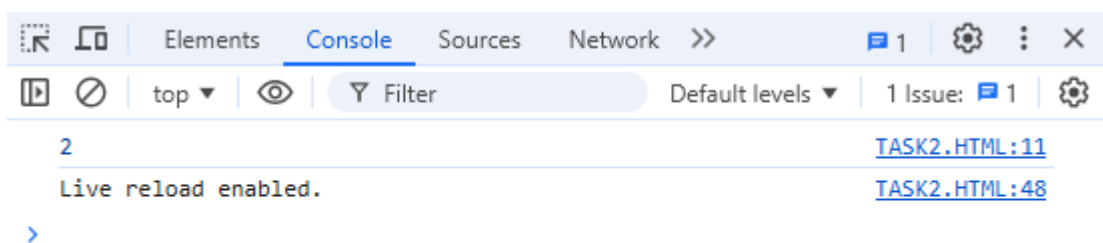
counter.getCount();

</script>

</body>

</html>
```

OUTPUT:



TASK 3:

```
<html>

  <body>

    <script>

      function createCounter() {

        let count = 0;

        return {

          increment: function() {

            count++;

          },

          getCount: function() {

            console.log(count);

          }

        };

      }

      const counter1 = createCounter();

      const counter2 = createCounter();

      counter1.increment();

      counter1.getCount();

      counter2.increment();

      counter2.getCount();

      counter1.increment();

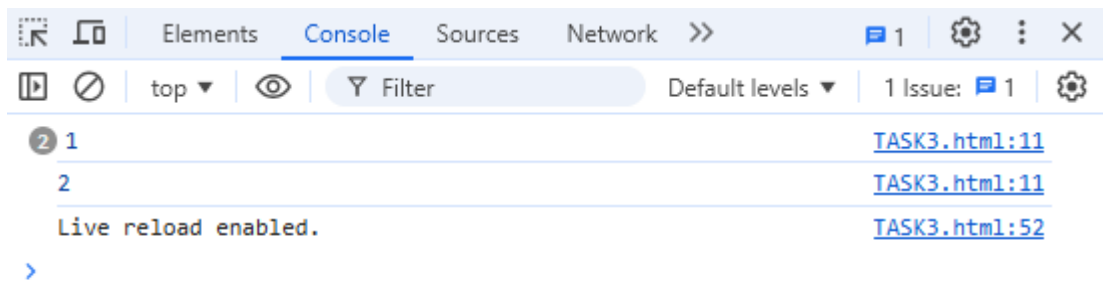
      counter1.getCount();

    </script>

  </body>

</html>
```

OUTPUT:



TASK 4:

<html>

<body>

<script>

function createPerson(name, age) {

let _name = name;

let _age = age;

return {

getName: function() {

return _name;

},

getAge: function() {

return _age;

},

setAge: function(newAge) {

if (newAge > _age) {

_age = newAge;

}

}

};

}

```

const person = createPerson('Alice', 30);

console.log(person.getName());

console.log(person.getAge());

person.setAge(31);

console.log(person.getAge());

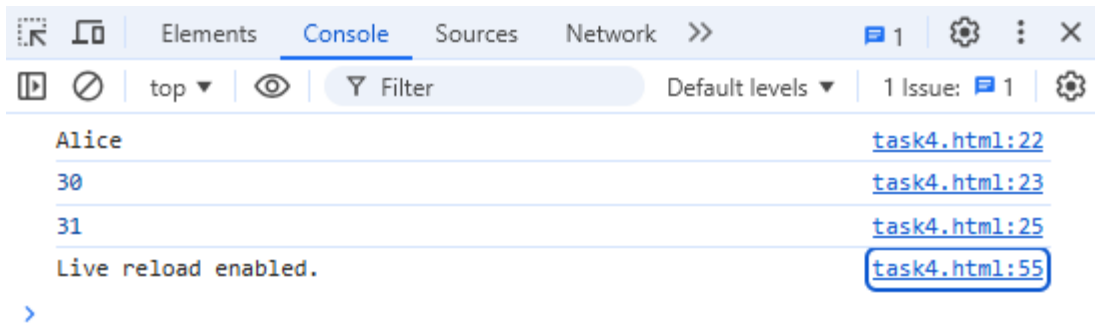
</script>

</body>

</html>

```

OUTPUT:



TASK 5:

```

<html>

<body>

  <script>

    function functionFactory(type) {

      if (type === 'greet') {

        return function(name) {

          console.log(`Hello, ${name}!`);

        };

      } else if (type === 'farewell') {

        return function(name) {

          console.log(`Goodbye, ${name}!`);

        };

      }

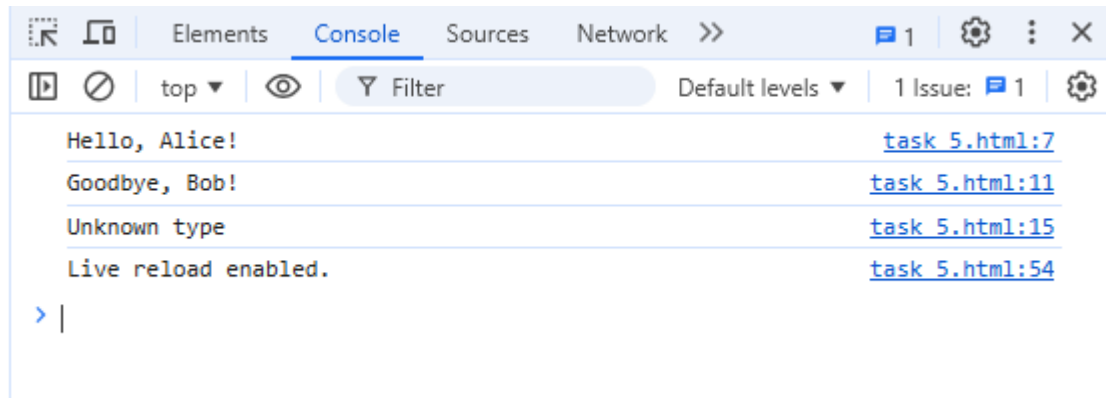
    }

  }

```

```
    } else {  
      return function() {  
        console.log('Unknown type');  
      };  
    }  
  }  
}  
  
const greetFunction = functionFactory('greet');  
greetFunction('Alice');  
  
const farewellFunction = functionFactory('farewell');  
farewellFunction('Bob');  
  
const unknownFunction = functionFactory('unknown');  
unknownFunction();  
  
</script>  
  
</body>  
  
</html>
```

OUTPUT:



6.MODULES,INTRODUCTION IMPORT AND EXPORT

TASK-1

```
<!DOCTYPE html>
<html>
<head>
<title>Module Example</title>
</head>
<body>
<script type="module" src = "App.js">
</script>
</body>
</html>
```

```
export function greet(name)
{
  return `Hello ${name}`;
}
export class Person {
  constructor(name,age){
    this.name = name;
    this.age = age;
  }
  introduce()
  {
    return `Name : ${this.name} , Age : ${this.age}`;
  }
}
export const pi = 3.14;
```

TASK-2

IMPORT FUNCTION:

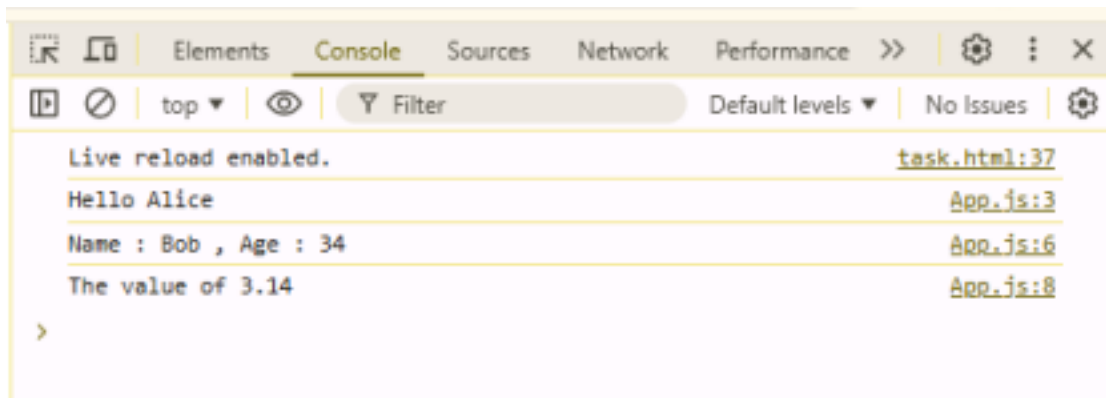
```
import {greet,Person,pi} from "./myModule.js";
```

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

```
const person1 = new Person("Bob","34");
console.log(person1.introduce());
```

```
console.log(`The value of ${pi}`);
```

OUTPUT:



TASK-3

```
<!DOCTYPE html>
<html>
<head>
<title>Module Example</title>
</head>
<body>
<script type="module" src = "App.js">
</script>
</body>
</html>
```

```
export function multiply(a,b)
{
  return a * b;
}
export function subtract(a,b)
{
  return a - b;
}
export function add(a,b,c)
{
  return a+b+c;
}
export class Person{
  constructor(name,age){
    this.name=name;
    this.age=age;
  }
  introduce()
  {
    return `My name is ${this.name},Age is ${this.age}`;
  }
}
```



```
}  
export const pi = 3.14;
```

TASK-4

IMPORT FUNCTION:

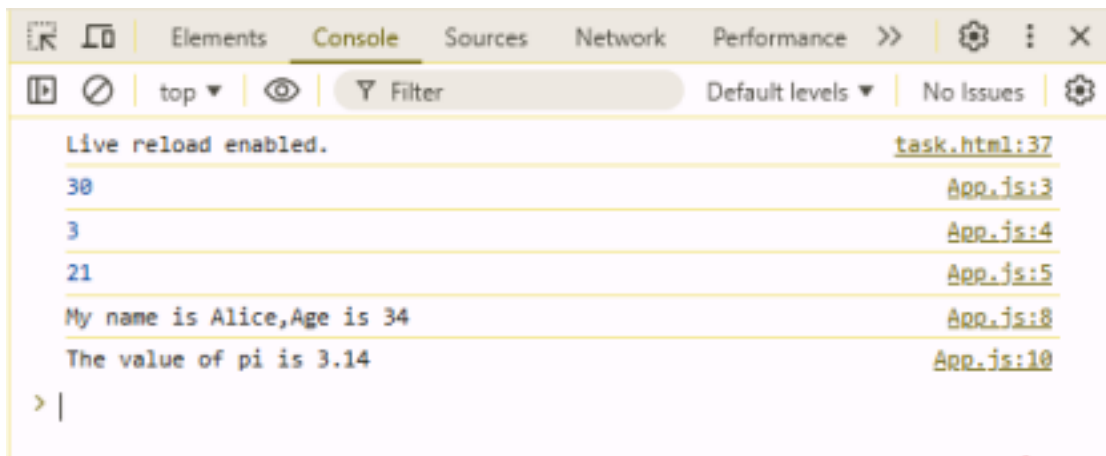
```
import {multiply,subtract,add,Person,pi} from "./myModule.js";
```

```
console.log(multiply(5,6));  
console.log(subtract(7,4));  
console.log(add(4,8,9));
```

```
const person1 = new Person("Alice",34);  
console.log(person1.introduce());
```

```
console.log(`The value of pi is ${pi}`);
```

OUTPUT:



TASK-5

```
<!DOCTYPE html>  
<html>  
<head>  
<title>Module Example</title>  
</head>  
<body>  
<script type="module" src = "App.js">  
</script>  
</body>  
</html>
```

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

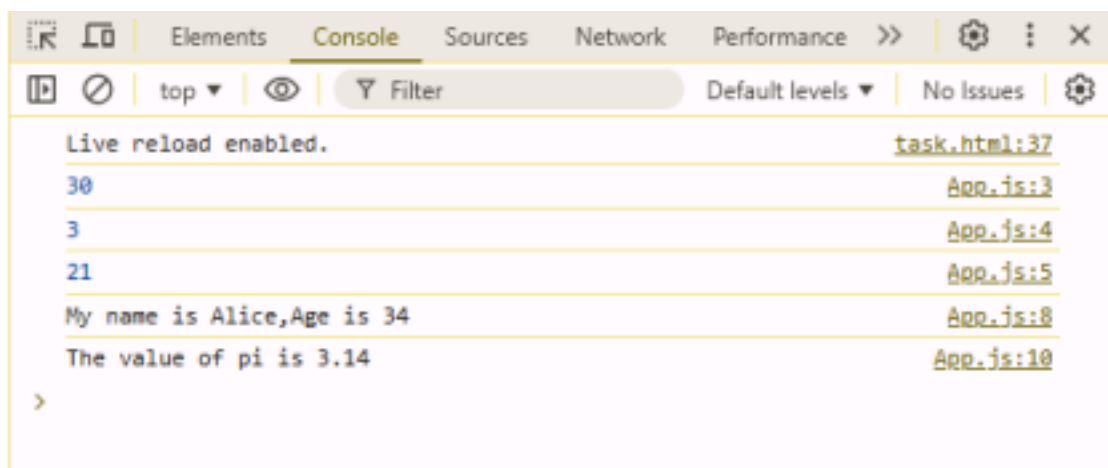
```
}
export function subtract(a,b)
{
  return a - b;
}
export function add(a,b,c)
{
  return a+b+c;
}
export class Person{
  constructor(name,age){
    this.name=name;
    this.age=age;
  }
  introduce()
  {
    return `My name is ${this.name},Age is ${this.age}`; }
}
export const pi = 3.14;

import multiply, {subtract ,add,Person,pi} from "./myModule.js";

console.log(multiply(5,6));
console.log(subtract(7,4));
console.log(add(4,8,9));

const person1 = new Person("Alice",34);
console.log(person1.introduce());

console.log(`The value of pi is ${pi}`);
OUTPUT:
```



TASK 1:

```
<html>
```

```
<script>
```

```
    const mypromise=new Promise((resolve)=>{
```

```
        setTimeout(()=>
```

```
        {
```

```
            resolve("welcome");
```

```
        },1000)
```

```
    });
```

```
    mypromise.then((value)=>{
```

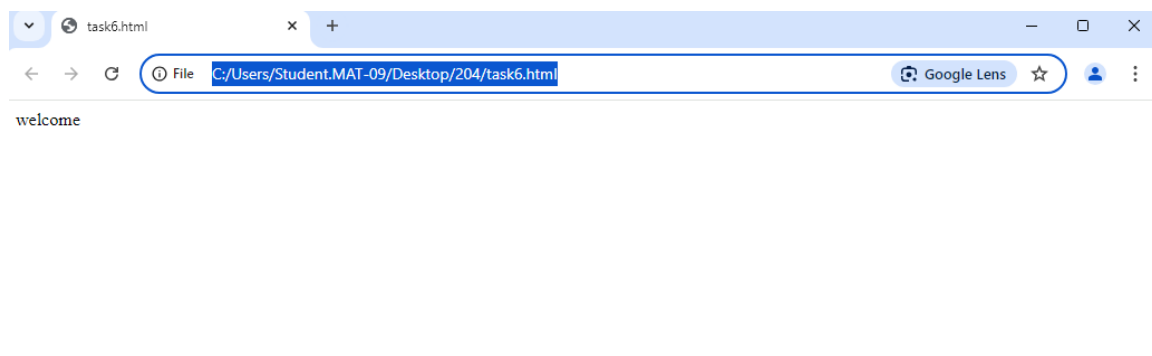
```
        document.write(value);
```

```
    })
```

```
</script>
```

```
</html>
```

OUTPUT:



TASK2:

```
<html>
```

```
<script>
```

```
    // Fetch data from an API (e.g., JSONPlaceholder)
```

```
function fetchData(url) {
```

```

return fetch(url)

.then(response => {
  if (!response.ok) {
    throw new Error('Network response was not ok');
  }

  return response.json(); // Parse the JSON data
});
}

// Process the fetched data (e.g., extract the name of the first user)
function processData(data) {
  return new Promise((resolve, reject) => {
    if (data && data.length > 0) {
      // For example, return the name of the first user
      resolve(`First user's name is: ${data[0].name}`);
    } else {
      reject('No data found to process. ');
    }
  });
}

```

```

// Use the fetchData function to get data, then chain the processData function
const apiUrl = 'https://jsonplaceholder.typicode.com/users';

```

```

fetchData(apiUrl)

.then((data) => {
  // Chain another promise to process the data
  return processData(data);
}

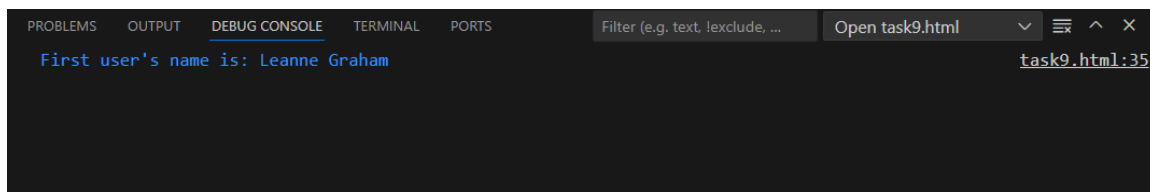
```

```
    })  
    .then((processedData) => {  
        console.log(processedData); // Output the processed data  
    })  
    .catch((error) => {  
        console.error('Error:', error); // Handle any errors  
    });
```

</script>

</html>

OUTPUT:



TASK3:

<html>

<script>

```
function random(){  
    return new Promise((resolve,reject)=>{  
        let random1=Math.random();  
        if(random1>0.5){  
            resolve("success");  
        }  
        else{  
            reject("failure");  
        }  
    });
```

```

    }

    random()

    .then((value)=>{

        document.write(value);

    })

    .catch((error)=>{

        document.write(error);

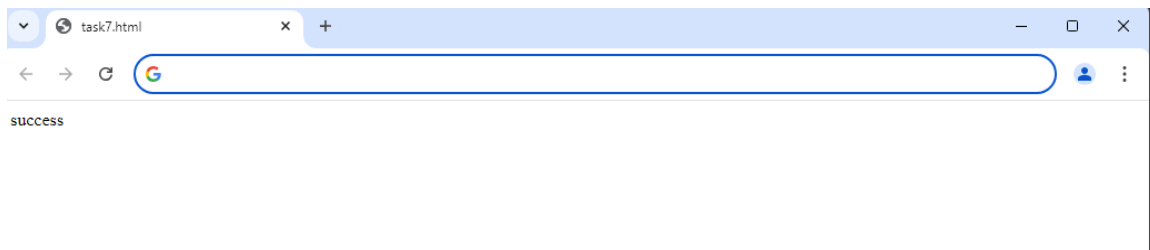
    })

```

</script>

</html>

OUTPUT:



TASK4:

<html>

<script>

// URLs to fetch data from

const url1 = '<https://jsonplaceholder.typicode.com/users>';

const url2 = '<https://jsonplaceholder.typicode.com/posts>';

const url3 = '<https://jsonplaceholder.typicode.com/albums>';

// Create an array of fetch promises

const fetchPromises = [

fetch(url1),

```

    fetch(url2),
    fetch(url3)
  ];

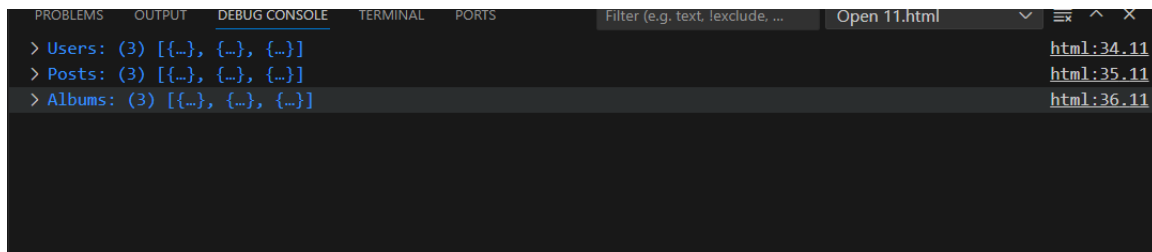
  // Use Promise.all to fetch all resources in parallel
  Promise.all(fetchPromises)
    .then(responses => {
      // All fetch requests are resolved, now process the responses
      return Promise.all(responses.map(response => response.json())); // Convert all responses to JSON
    })
    .then(data => {
      // Process the data from all the responses
      const users = data[0];
      const posts = data[1];
      const albums = data[2];

      console.log('Users:', users); // Output the users
      console.log('Posts:', posts); // Output the posts
      console.log('Albums:', albums); // Output the albums
    })
    .catch(error => {
      // Handle any error from the fetch requests or data processing
      console.error('Error:', error);
    });

</script>
</html>

```

OUTPUT:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, lexclude. ... Open 11.html
> Users: (3) [{...}, {...}, {...}] html:34.11
> Posts: (3) [{...}, {...}, {...}] html:35.11
> Albums: (3) [{...}, {...}, {...}] html:36.11
```

TASK5:

<html>

<script>

// Simulate an asynchronous task with a delay

```
function asyncTask(message, delay) {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      console.log(message);
      resolve(message);
    }, delay);
  });
}
```

// Chain promises to perform tasks in sequence

```
asyncTask("Task 1: Starting", 1000)
  .then((result1) => {
    // Task 1 is done, now run Task 2
    return asyncTask("Task 2: Starting", 1500);
  })
  .then((result2) => {
    // Task 2 is done, now run Task 3
    return asyncTask("Task 3: Starting", 2000);
  })
```



```
.then((result3) => {  
    // Task 3 is done, now run Task 4  
    return asyncTask("Task 4: Finished", 1000);  
})  
.catch((error) => {  
    console.error("Error:", error);  
});
```

</script>

</html>

