3. Arrow Functions:

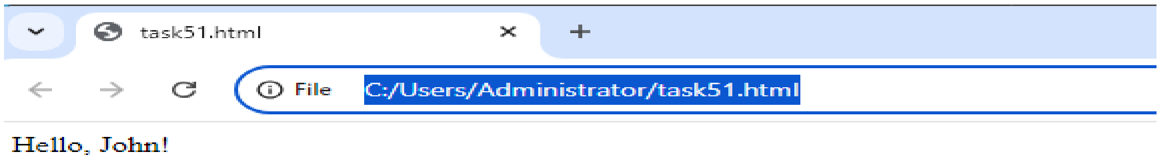
Task 51

Declare a simple arrow function named greet that takes one parameter name and returns the string “Hello, name!”. Test your function with various names.

CODE:

```
<html>
    <body>
        <script>
            let greet=(name)=>{
                document.writeln("Hello, "+name+"!");
            }
            greet("John")
        </script>
    </body>
</html>
```

OUTPUT:



```
Hello, John!
```

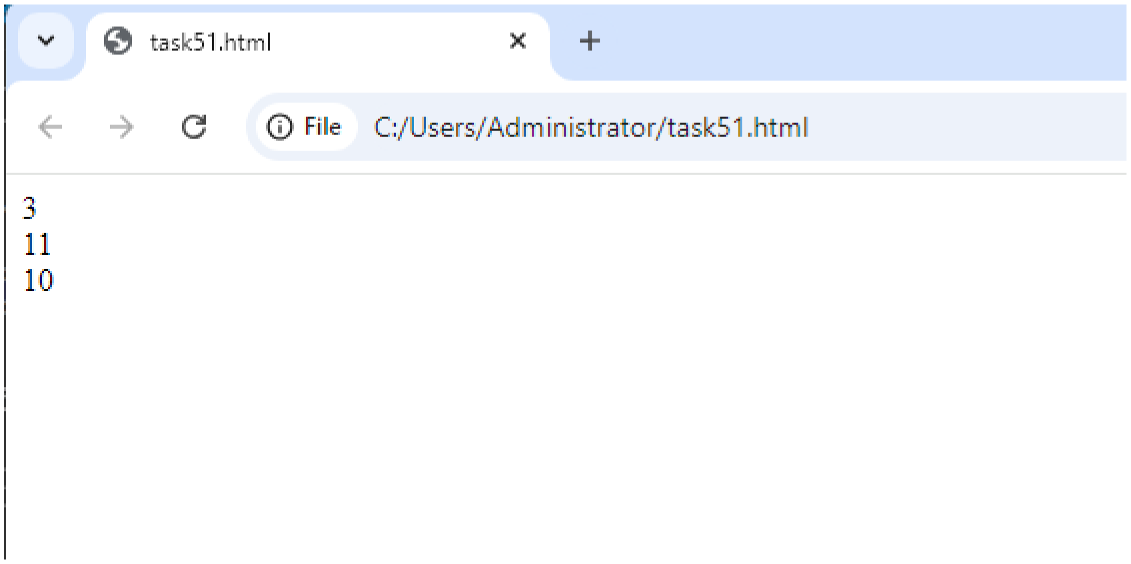
Task 52

Write an arrow function named add that takes two parameters and returns their sum. Validate your function with several pairs of numbers.

CODE:

```
<html>
    <body>
        <script>
            let add=(n1,n2)=>{
                return n1+n2;
            }
            document.writeln(add(1,2)+"<br>");
            document.writeln(add(9,2)+"<br>");
            document.writeln(add(10,0)+"<br>");
        </script>
    </body>
</html>
```

OUTPUT:



```
3
11
10
```

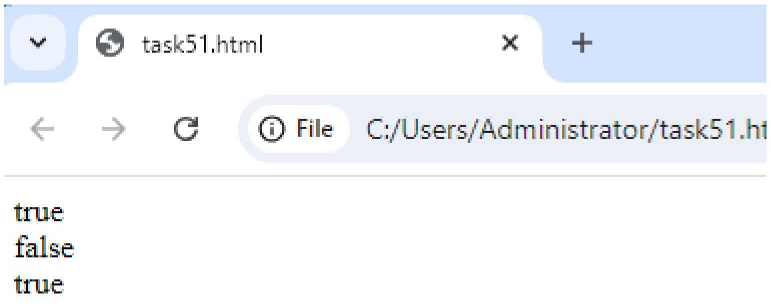
Task 53

Declare an arrow function named isEven that checks if a number is even. If the number is even, it should return true; otherwise, false. Remember that if the arrow function body has a single statement, you can omit the curly braces.

CODE:

```
<html>
    <body>
        <script>
        let isEven = (num) => num % 2 === 0;
            document.writeln(isEven(14)+"<br>");
            document.writeln(isEven(19)+"<br>");
            document.writeln(isEven(10)+"<br>");
        </script>
    </body>
</html>
```

OUTPUT:



```
true
false
true
```

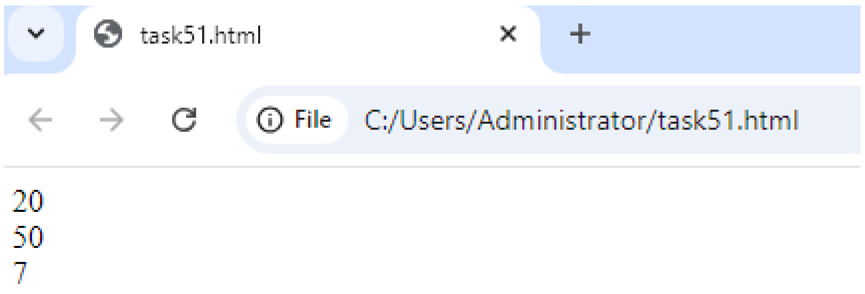
Task 54

Implement an arrow function named maxValue that takes two numbers as parameters and returns the larger number. Here, you'll need to use curly braces for the function body and the return statement.

CODE:

```
<html>
    <body>
        <script>
        let maxValue = (a, b) => {
        return a > b ? a : b;
        };

        document.writeln(maxValue(10, 20)+"<br>");
        document.writeln(maxValue(50, 30)+"<br>");
        document.writeln(maxValue(7, 7));

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

OUTPUT:



```
20
50
7
```

Task 55

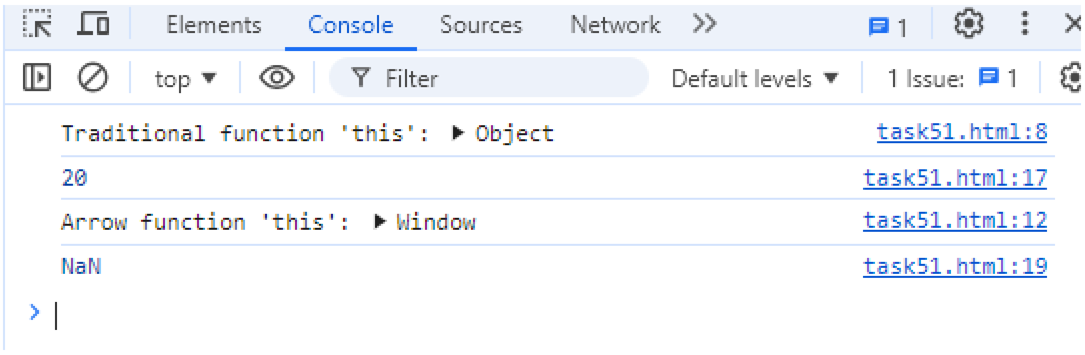
Examine the behavior of the this keyword inside an arrow function vs a traditional function. Create an object named myObject with a property value set to 10 and two methods: multiplyTraditional using a traditional function and multiplyArrow using an arrow function. Both methods should attempt to multiply the value property by a number passed as a parameter. Check the value of this inside both methods.

CODE:

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

const myObject = {
  value: 10,
  multiplyTraditional: function (num) {
    console.log("Traditional function 'this':", this);
    return this.value * num;
  },
  multiplyArrow: (num) => {
    console.log("Arrow function 'this':", this);
    return this.value * num;
  }
};

console.log(myObject.multiplyTraditional(2));

console.log(myObject.multiplyArrow(2));



        </script>
    </body>
```

```
</html>
```

OUTPUT:



```
Traditional function 'this':  ▶ Object          task51.html:8
20                                              task51.html:17
Arrow function 'this':  ▶ Window                task51.html:12
NaN                                             task51.html:19
```