Practical 1

Task-1

Aim: Declare a variable using var, let, and const. Assign different data types to each variable and print their values.

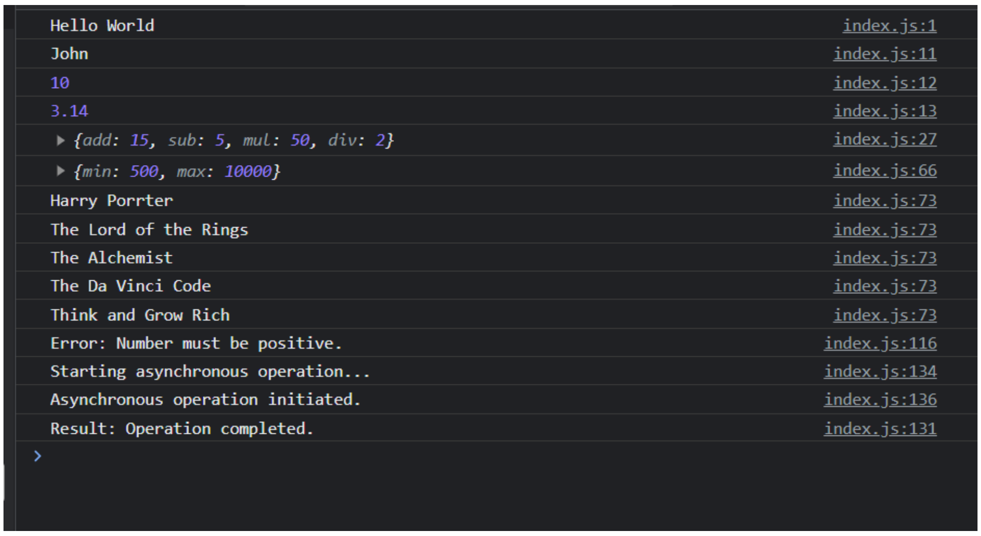
Theoretical Background: in JavaScript, there are three keywords that can be used to declare variables: var, let, and const.

- **var** is the oldest keyword, and it has been the default way to declare variables in JavaScript since the language was first created. Variables declared with var have global scope by default, which means that they can be accessed from anywhere in the JavaScript code. var variables can also be reassigned, which means that their value can be changed after they have been declared.
- **let** is a newer keyword that was introduced in ECMAScript 6 (ES6). Variables declared with let have block scope, which means that they can only be accessed within the block in which they are declared. let variables can also be reassigned, just like var variables.
- **const** is the newest keyword, and it was also introduced in ES6. Variables declared with const have constant scope, which means that their value cannot be changed after they have been declared. const variables cannot be reassigned, and they cannot be declared without being initialized.

Source Code:

```
// Task 1: Variables and Data Types

var Name = "John";
let Number = 10;
const PI = 3.14;

console.log(Name);
console.log(Number);
console.log(PI);
```

Output:



```
Hello World                                   index.js:1
John                                          index.js:11
10                                            index.js:12
3.14                                          index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}          index.js:27
▸ {min: 500, max: 10000}                      index.js:66
Harry Porrter                                 index.js:73
The Lord of the Rings                         index.js:73
The Alchemist                                 index.js:73
The Da Vinci Code                             index.js:73
Think and Grow Rich                           index.js:73
Error: Number must be positive.               index.js:116
Starting asynchronous operation...            index.js:134
Asynchronous operation initiated.             index.js:136
Result: Operation completed.                  index.js:131
>
```

Task-2

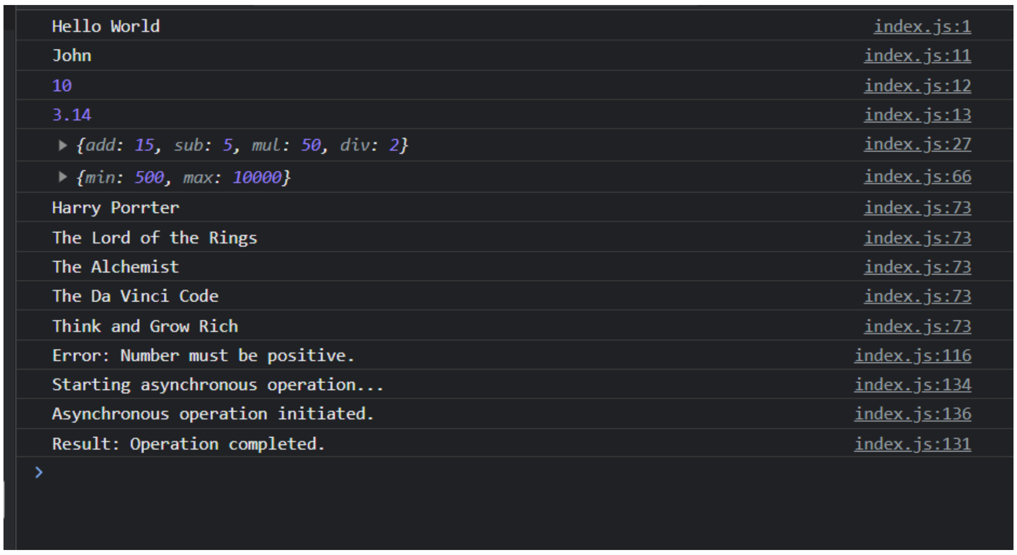
Aim: Write a function that takes two numbers as arguments and returns their sum, difference, product, and quotient using arithmetic operators.

Theoretical Background: In JavaScript, arithmetic operators are used to perform mathematical operations on numbers. The most common arithmetic operators are addition (+), subtraction (-), multiplication (*), division (/), and modulus (%).

Source Code:

```
let calc = function (num1, num2) {
    return {
        add: num1 + num2,
        sub: num1 - num2,
        mul: num1 * num2,
        div: num1 / num2,
    }
}

console.log(calc(10, 5));
```

Output:



```
Hello World                                        index.js:1
John                                               index.js:11
10                                                 index.js:12
3.14                                               index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}               index.js:27
▸ {min: 500, max: 10000}                           index.js:66
Harry Porrter                                      index.js:73
The Lord of the Rings                              index.js:73
The Alchemist                                      index.js:73
The Da Vinci Code                                  index.js:73
Think and Grow Rich                                index.js:73
Error: Number must be positive.                    index.js:116
Starting asynchronous operation...                 index.js:134
Asynchronous operation initiated.                  index.js:136
Result: Operation completed.                       index.js:131
›
```

Task-3

Aim: Write a program that prompts the user to enter their age. Based on their age, display different messages:

○ If the age is less than 18, display "You are a minor."

○ If the age is between 18 and 65, display "You are an adult."

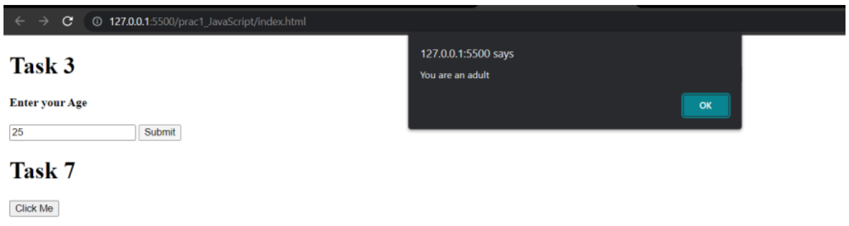
○ If the age is 65 or older, display "You are a senior citizen."

Theoretical Background: In JavaScript, you can use the prompt() function to prompt the user to enter input or from the Html using Input Tag . The prompt() function takes a message as an argument, and it returns a string containing the user's input.

Source Code:

```
let getage = function () {
    var age = document.getElementById("id_Age").value;
    // age = parseInt(age);
    if (age < 18) {
        alert("You are a minor");
    } else if (age >= 18 && age < 65) {
        alert("You are an adult");
    } else {
        alert("You are a senior citizen"); }}
```

Output:



127.0.0.1:5500/prac1_JavaScript/index.html

Task 3

Enter your Age

25 Submit

Task 7

Click Me

127.0.0.1:5500 says

You are an adult

OK

Task-4

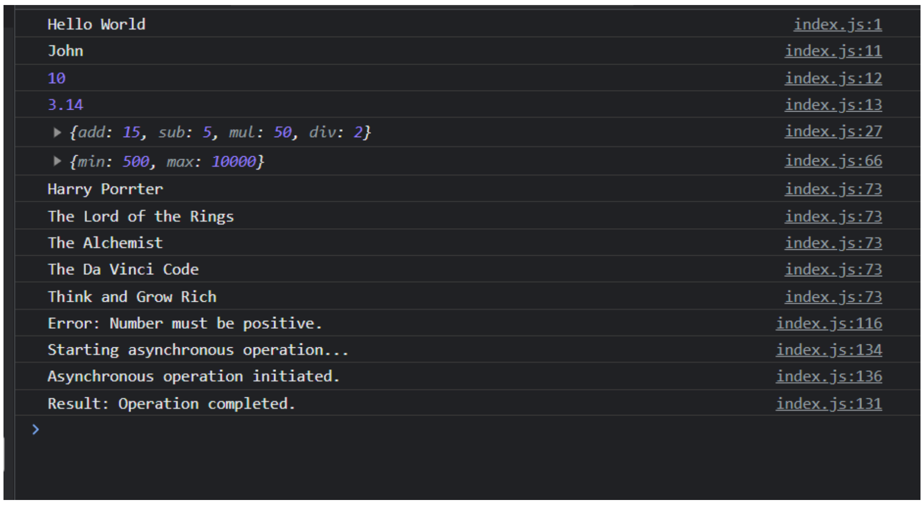
Aim: Write a function that takes an array of salary as an argument and returns the min/max salary in the array.

Theoretical Background: in JavaScript, you can use the min() and max() functions to find the minimum and maximum values in an array. The min() function takes an array as an argument and returns the smallest element in the array. The max() function takes an array as an argument and returns the largest element in the array.

Source Code:

```
// Task 4: Functions

let getMinMaxSalary = function (salaries) {

    let min = Math.min(…salaries);
    let max = Math.max(…salaries);
    return {
        min: min,
        max: max
    };
}

let salaries = [1000, 500, 2000, 3000, 4000, 6000, 7000, 8000, 9000, 10000];
console.log(getMinMaxSalary(salaries));
```

Output:



```
Hello World                                         index.js:1
John                                                index.js:11
10                                                  index.js:12
3.14                                                index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}                index.js:27
▸ {min: 500, max: 10000}                            index.js:66
Harry Porrter                                       index.js:73
The Lord of the Rings                               index.js:73
The Alchemist                                       index.js:73
The Da Vinci Code                                   index.js:73
Think and Grow Rich                                 index.js:73
Error: Number must be positive.                     index.js:116
Starting asynchronous operation...                  index.js:134
Asynchronous operation initiated.                   index.js:136
Result: Operation completed.                        index.js:131
>
```

Task-5

Aim: Create an array of your favorite books. Write a function that takes the array as an argument and displays each book title on a separate line.

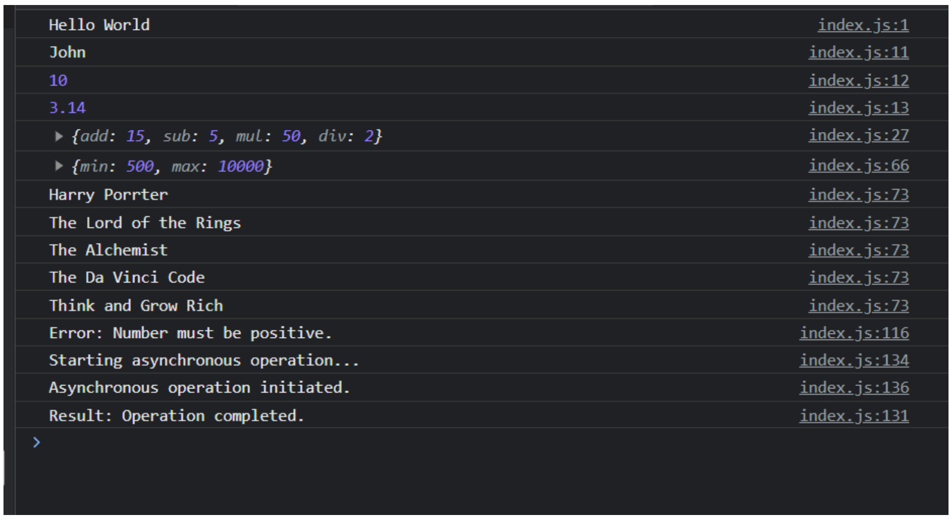
Theoretical Background:

- Arrays are a data structure that allows you to store a collection of data items. In JavaScript , arrays are indexed, which means that you can access individual items in the array by their index.
- Functions are blocks of code that can be reused. Functions take arguments as input and return a value as output.
- For loops are used to iterate through a sequence of items. In JavaScript , for loops can be used to iterate through arrays.
- Print is a function that is used to display text to the console.

Source Code:

```
// Task 5: Arrays and Objects

let displayFavBooks = function (books) {
    for (let i = 0; i < books.length; i++) {
        console.log(books[i]);
    }
}

let books = ["Harry Porrter", "The Lord of the Rings", "The Alchemist", "The Da Vinci
Code", "Think and Grow Rich"];
displayFavBooks(books);
```

Output:



```
Hello World                                         index.js:1
John                                                index.js:11
10                                                  index.js:12
3.14                                                index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}                index.js:27
▸ {min: 500, max: 10000}                            index.js:66
Harry Porrter                                       index.js:73
The Lord of the Rings                               index.js:73
The Alchemist                                       index.js:73
The Da Vinci Code                                   index.js:73
Think and Grow Rich                                 index.js:73
Error: Number must be positive.                     index.js:116
Starting asynchronous operation...                  index.js:134
Asynchronous operation initiated.                   index.js:136
Result: Operation completed.                        index.js:131
>
```

Task-6

Aim: Declare a variable inside a function and try to access it outside the function. Observe the scope behavior and explain the results. [var vs let vs const]

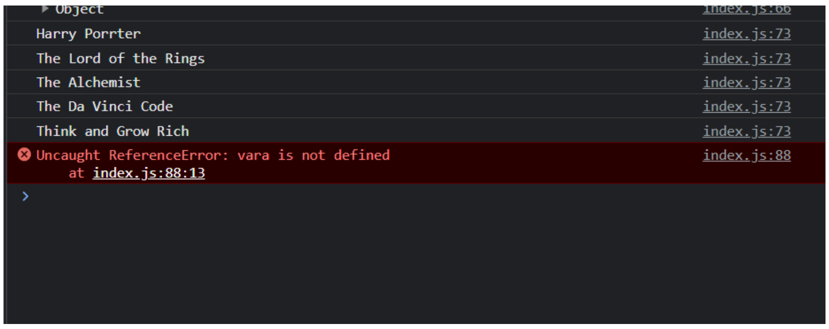
Theoretical Background: In JavaScript, there are three types of variables:

- var variables are the default type of variable in JavaScript. var variables have global scope, which means that they can be accessed from anywhere in the program.
- let variables were introduced in JavaScript . let variables have block scope, which means that they can only be accessed within the block in which they are declared.
- const variables were also introduced in JavaScript. const variables have constant scope, which means that their value cannot be changed after they have been declared

Source Code:

```
// Task 6: Scope and Hoisting

let ScopeandHoisting = function () {
    let vara = 100;
    var varb = 200;
    const varc = 300;
}

console.log(vara);
console.log(varb);
console.log(varc);
```

Output:



```
 ▸ Object                                              index.js:66
Harry Porrter                                          index.js:73
The Lord of the Rings                                  index.js:73
The Alchemist                                          index.js:73
The Da Vinci Code                                      index.js:73
Think and Grow Rich                                    index.js:73
✖ Uncaught ReferenceError: vara is not defined         index.js:88
     at index.js:88:13
>
```

Task-7

Aim: Create an HTML page with a button. Write JavaScript code that adds an event listener to the button and changes its text when clicked.

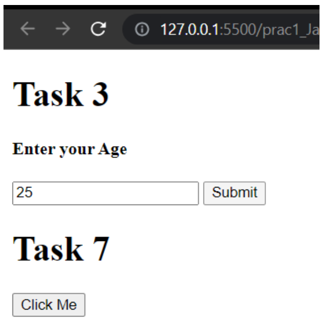
Theoretical Background:

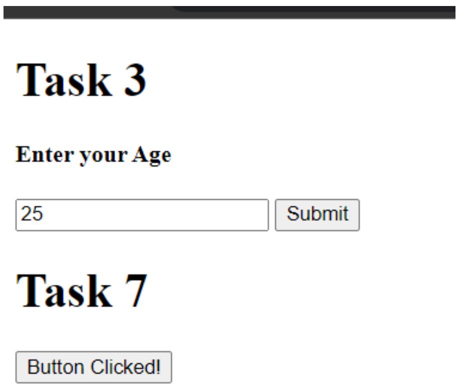
Source Code:

- HTML is a markup language that is used to create web pages. HTML pages are made up of elements, which are the basic building blocks of HTML.
- Buttons are HTML elements that are used to trigger an action. When a button is clicked, it fires an event.
- JavaScript is a programming language that is used to add interactivity to web pages. JavaScript can be used to add event listeners to buttons and change their text when they are clicked.

```
// Task 7: DOM Manipulation

var button = document.getElementById("myButton");

button.addEventListener("click", function () {
    button.textContent = "Button Clicked!";
});
```

Output:





Task-8

Aim: Write a function that takes a number as an argument and throws an error if the number is negative. Handle the error and display a custom error message.

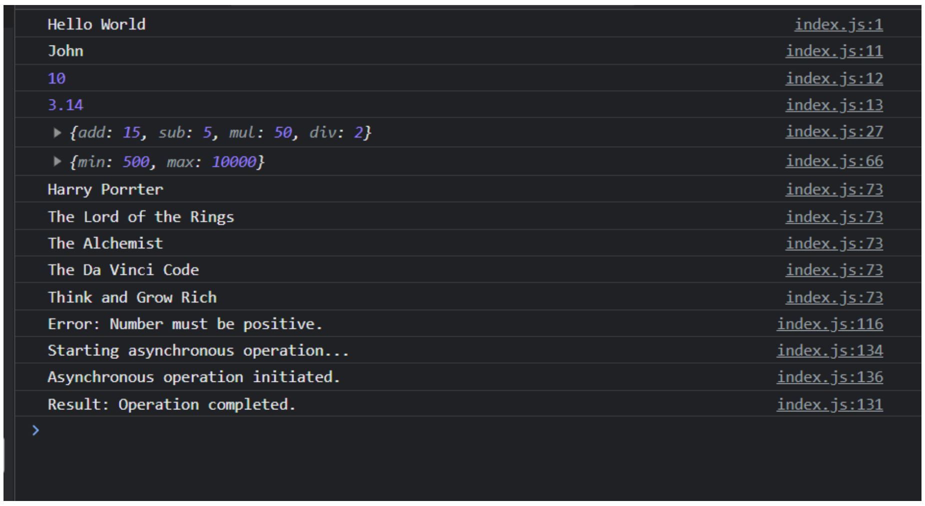
Theoretical Background:

- Errors are unexpected events that occur during the execution of a program. Errors can be caused by a variety of reasons, such as invalid input, invalid operations, or hardware failures.
- Error handling is the process of detecting and responding to errors. Error handling can be done using try-catch blocks, which allow you to catch and handle errors that occur during the execution of a program.
- Custom error messages are error messages that you can create yourself. Custom error messages can be used to provide more information about the error that occurred.

Source Code:

```
// Task 8: Error Handling


function checkPositiveNumber(number) {
    if (number < 0) {
        throw new Error("Number must be positive.");
    }

    return number;
}

try {
    var result = checkPositiveNumber(-5);
    console.log("Result:", result);
} catch (error) {
    console.log("Error:", error.message);
}
```

Output:



```
Hello World                                  index.js:1
John                                         index.js:11
10                                           index.js:12
3.14                                         index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}         index.js:27
▸ {min: 500, max: 10000}                     index.js:66
Harry Porrter                                index.js:73
The Lord of the Rings                        index.js:73
The Alchemist                                index.js:73
The Da Vinci Code                            index.js:73
Think and Grow Rich                          index.js:73
Error: Number must be positive.              index.js:116
Starting asynchronous operation...           index.js:134
Asynchronous operation initiated.            index.js:136
Result: Operation completed.                 index.js:131
>
```

Task-9

Aim: Write a function that uses setTimeout to simulate an asynchronous operation. Use a callback function to handle the result.

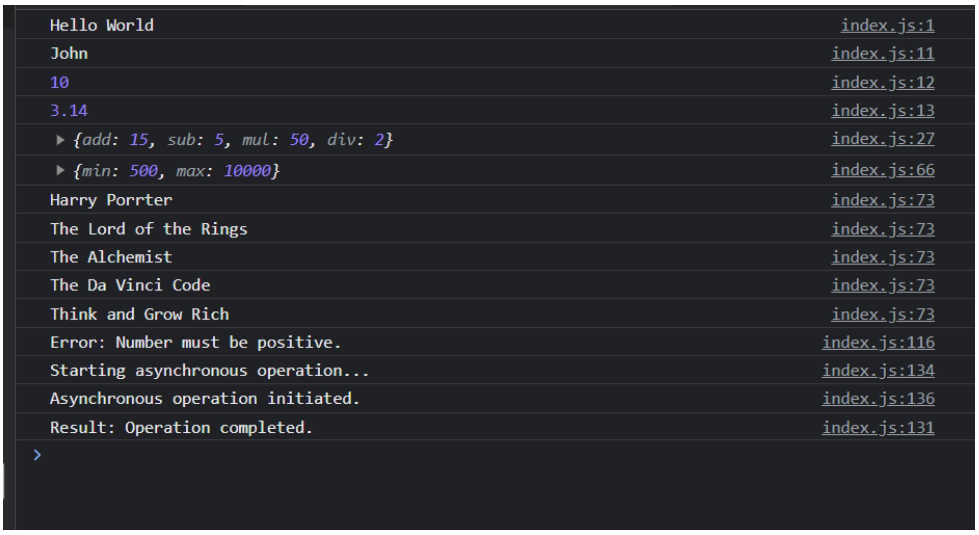
Theoretical Background:

- Timeouts are a way to delay the execution of a function for a specified amount of time. Timeouts are often used to simulate asynchronous operations, which are operations that take some time to complete.
- Callback functions are functions that are passed as arguments to other functions. Callback functions are executed when the other function completes.

Source Code:

```
// Task 9: Asynchronous JavaScript

function asyncFunction(callback) {
    setTimeout(function () {

        var result = "Operation completed.";

        callback(result);
    }, 10000);
}

function callback(result) {
    console.log("Result:", result);
}

console.log("Starting asynchronous operation...");
asyncFunction(callback);
console.log("Asynchronous operation initiated.")
```

Output:



```
Hello World                                         index.js:1
John                                                index.js:11
10                                                  index.js:12
3.14                                                index.js:13
▸ {add: 15, sub: 5, mul: 50, div: 2}                index.js:27
▸ {min: 500, max: 10000}                            index.js:66
Harry Porrter                                       index.js:73
The Lord of the Rings                               index.js:73
The Alchemist                                       index.js:73
The Da Vinci Code                                   index.js:73
Think and Grow Rich                                 index.js:73
Error: Number must be positive.                     index.js:116
Starting asynchronous operation...                  index.js:134
Asynchronous operation initiated.                   index.js:136
Result: Operation completed.                        index.js:131
>
```