

## Task -1

**Aim:** Variables and Data Types,

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

### Description:

In this practical we are supposed to declare a variable using

1. Var – Variable declared with var is accessible within the function in which it is declared, or globally if declared outside of any function.
2. Let – Let is used to declare a variable which is block scoped in JavaScript.
3. Const – Const is declared when you don't want to change the value of the variable stored inside const.

### Source Code:

```
var x = 10;  
let y = 20;  
const c = 30;  
console.log(x);  
console.log(y);  
console.log(c);
```

### Output:



```
node /tmp/Xew4ok2Rqj.js  
10  
20  
30  
|
```

### Learning Outcome:

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -2

**Aim:** Operators and Expressions,

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

**Description:**

In this practical we are asked to print / calculate the basic math operations performed on two numbers which are user input values.

**Source Code:**

```
var aa = prompt("first number :");
var num1 = parseInt(aa);
var bb = prompt("second number :");
var num2 = parseInt(bb);
var add = add(num1, num2);
var sub = sub(num1, num2);
var mul = mul(num1, num2);
var div = div(num1, num2);
```

```
function add(n1,n2){
    return n1 + n2;
}
function sub(n1,n2){
    return n1 - n2;
}
function mul(n1,n2){
    return n1 * n2;
}
function div(n1,n2){
    return n1 / n2;
}
```

```
console.log("add :"+add);
console.log("sub :"+sub);
console.log("mul :"+mul);
console.log("div :"+div);
```

**Output:**

```
node /tmp/Xew4ok2Rqj.js
first number :23
second number :21
add :44
sub :2
mul :483
div :1.0952380952380953
```

## Learning Outcome:

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -3

### Aim: Control Flow

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."

### Description:

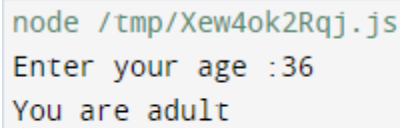
In this program we are asked to take user input of age and then classify them on the basis of the age as:

- If age is less than 18 , “ You are minor ”
- If age is between 18 and 65, “You are adult”
- If age is above 65, “You are Senior Citizen”

### Source Code:

```
var cc = prompt("Enter your age :");
var num3 = parseInt(cc);
//console.log("Your age is :"+ num3);
if(num3<18){
    console.log("You are minor");
}
else if(num3>18 && num3<65){
    console.log("You are adult");
}
```

```
else{  
    console.log("You are senior citizen");  
}
```

**Output:**

```
node /tmp/Xew4ok2Rqj.js  
Enter your age :36  
You are adult
```

**Learning Outcome:**

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -4

**Aim:** Functions

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

**Description:**

In this practical we are asked to find out the maximum and minimum salaries from a list of salaries which is a static list.

**Source Code:**

```
function findMinMaxSalary(salaries) {  
    var minSalary = Math.min(...salaries);  
    var maxSalary = Math.max(...salaries);  
    console.log("Minimum :"+minSalary);  
    console.log("Maximum :"+maxSalary);  
}  
var salaries = [30000, 50000, 25000, 80000, 60000];  
var result = findMinMaxSalary(salaries);
```

**Output:**

```
node /tmp/Xew4ok2Rqj.js  
Minimum :25000  
Maximum :80000  
|
```

**Learning Outcome:**

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -5

**Aim:** Arrays and Objects

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.

**Description:**

In This practical we are asked to make a array of our favorites books and also create a function that would display all the books in new line.

**Source Code:**

```
var favoriteBooks = [  
  "Zero to One",  
  "unchained",  
  "48 laws of power",  
  "Do It",  
  "The Ultimate Power"  
];  
  
function displayBookTitles(books) {  
  for (var i = 0; i < books.length; i++) {  
    console.log(books[i]);  
  }  
}  
  
// Call the function with the array of favorite books  
displayBookTitles(favoriteBooks);
```

**Output:**

```
node /tmp/Xew4ok2Rqj.js
Zero to One
unchained
48 laws of power
Do It
The Ultimate Power
```

## Learning Outcome:

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -6

### Aim: Scope and Hoisting

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].

### Description:

In this practical we are asked to declare var, let and const in separate function and then asked to check that whether we will be able to display the value outside the function.

### Source Code:

```
function testScope() {
  var varVariable = 'This is a var .';
  let letVariable = 'This is a let .';
  const constVariable = 'This is a const .';
}

testScope();

console.log(varVariable); // Error: varVariable is not defined
console.log(letVariable); // Error: letVariable is not defined
console.log(constVariable); // Error: constVariable is not defined
```

### Output:

```
node /tmp/Xew4ok2Rqj.js
ERROR!
/tmp/Xew4ok2Rqj.js:9
console.log(varVariable);    // Error: varVariable is not defined
      |      ^

ReferenceError: varVariable is not defined
    at Object.<anonymous> (/tmp/Xew4ok2Rqj.js:9:13)
    at Module._compile (node:internal/modules/cjs/loader:1254:14)
    at Module._extensions..js (node:internal/modules/cjs/loader:1308:10)
    at Module.load (node:internal/modules/cjs/loader:1117:32)
    at Module._load (node:internal/modules/cjs/loader:958:12)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:81:12)
    at node:internal/main/run_main_module:23:47

Node.js v18.16.0
```

## Learning Outcome:

CO1 : Understand various technologies and trends impacting single page web applications.

## Task -7

### Aim: DOM Manipulation

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

### Description:

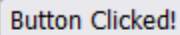
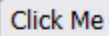
In this practical we are asked to add a Event Listener which would change the value inside a button when clicked on using JavaScript.

### Source Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Button Event Listener</title>
</head>
<body>
  <button id="myButton">Click Me</button>

  <script>
    var button = document.getElementById("myButton");
```

```
        button.addEventListener("click", function() {
            button.textContent = "Button Clicked!";
        });
    </script>
</body>
</html>
```

**Output:****Learning Outcome:**

CO1 : Understand various technologies and trends impacting single page web applications.

CO2 : Apply a deep knowledge of MVC(ModelViewController) architecture,making the development process easier and faster using open-source technologies.

## Task -8

**Aim:** Error Handling

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.

**Description:**

In this practical we are asked to check that the value that the user has input is positive or not:

If yes, then display error message

Else, Display the number

**Source Code:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Negative Number Error</title>
```



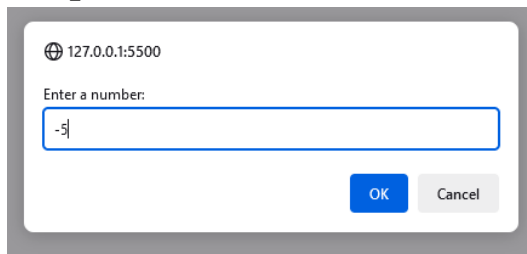
```
</head>
<body>
  <p id="error"></p>
  <p id="result"></p>

  <script>
    function checkPositiveNumber(number) {
    if (number < 0) {
      throw new Error("Number cannot be negative.");
    }

    // If the number is not negative, do something with it or return it.
    // For this example, let's just return the number.
    return number;
  }

  try {
    var userInput = parseInt(prompt('Enter a number:'));
    var result = checkPositiveNumber(userInput);
    document.getElementById("result").innerHTML=result;
  }
  catch (error) {
    document.getElementById("error").innerHTML=error.message;
  }
  </script>

</body>
</html>
```

**Output:**

Number cannot be negative.

**Learning Outcome:**

CO1 : Understand various technologies and trends impacting single page web applications.

CO2 : Apply a deep knowledge of MVC(ModelViewController) architecture,making the development process easier and faster using open-source technologies.

## Task -9

**Aim:** Asynchronous JavaScript

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

### Description:

Use Asynchronous JavaScript to call a function.

### Source Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Async Function Example</title>
</head>
<body>
  <h1>Async Function Example</h1>
  <button onclick="runAsyncFunction()">Run Async Function</button>

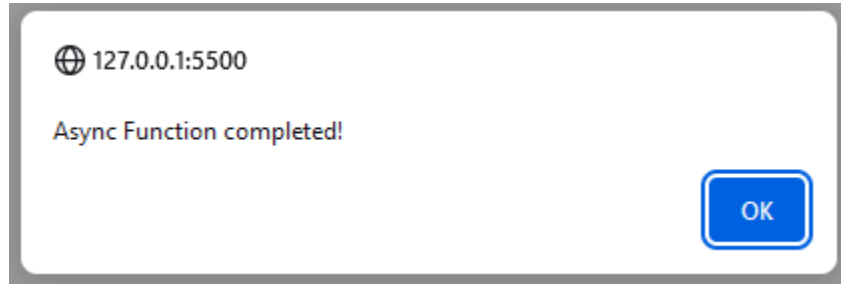
  <script>
    function runAsyncFunction() {
      // Simulate an asynchronous operation using setTimeout
      setTimeout(function() {
        // Call the callback function with the result
        asyncFunctionCallback('Async Function completed!');
      }, 2000);
    }

    function asyncFunctionCallback(result) {
      // Handle the result of the asynchronous operation
      alert(result);
    }
  </script>
</body>
</html>
```

### Output:

# Async Function Example

Run Async Function



## Learning Outcome:

C01 : Understand various technologies and trends impacting single page web applications.

C02 : Apply a deep knowledge of MVC(ModelViewController) architecture,making the development process easier and faster using open-source technologies.