

Practical-1 (Javascript)

Task-1

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

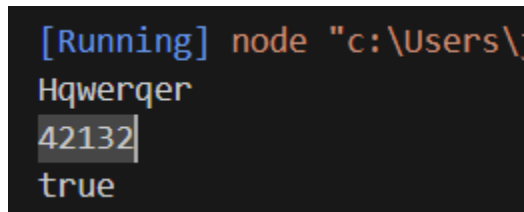
Theoretical Background:

→ If you never want a variable to change, const is the keyword to use. If you want to reassign values: and you want the hoisting behavior, var is the keyword to use. if you don't want it, let is the keyword for you.

Source Code:

```
var example = 'Hqwerqer';  
console.log(example);  
let exapmle1 = 42132;  
console.log(exapmle1);  
const myConst = true;  
console.log(myConst);
```

Output:



```
[Running] node "c:\Users\  
Hqwerqer  
42132  
true
```

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:

- Operators are symbols that perform operations on variables and values.
- Expressions are combinations of operators and operands that evaluate to a single value.
- The order in which operators are evaluated is determined by the operator precedence table.

Source Code:

```
function arithmeticOperations(x, y) {  
  let sum = x + y;  
  let difference = x - y;  
  let product = x * y;  
  let quotient = x / y;  
  return {
```

```

    sum: sum,
    difference: difference,
    product: product,
    quotient: quotient
  };
}
let results = arithmeticOperations(10, 5);
console.log(results);

```

Output:

```

[Running] node "/Users/mac/Desktop/5/trial.js"
{ sum: 15, difference: 5, product: 50, quotient: 2 }

```

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:

- Control flow is the order in which JavaScript code is executed. It can be changed using conditional statements (such as if/else), loops (such as for/while), and functions.
- For example, an if/else statement will only execute the code in the if block if the condition is true. Otherwise, it will execute the code in the else block.

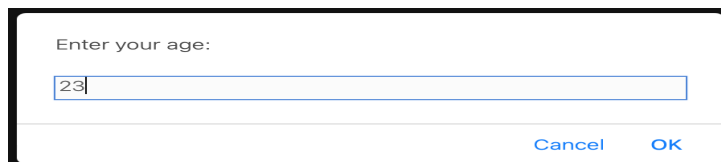
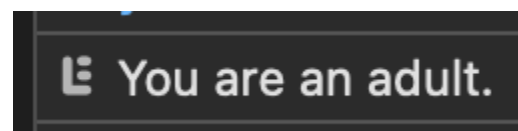
Source Code:

```

var age = parseInt(prompt("Enter your age:"));
if (age < 18) {
  console.log("You are a minor.");
} else if (age >= 18 && age <= 65) {
  console.log("You are an adult.");
} else {
  console.log("You are a senior citizen.");
}

```

Output:

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:

- Functions are reusable blocks of code that can be called to perform a specific task.
- They are declared using the function keyword, followed by the function name, the parameter list, and the body of the function.
- The body of the function is enclosed in curly braces ({}) and can contain any valid JavaScript code.
- When a function is called, the code in its body is executed.

Source Code:

```
function minMaxSalary(salaryArray) {  
    let minSalary = Infinity;  
    let maxSalary = -Infinity;  
    for (const salary of salaryArray) {  
        if (salary < minSalary) {  
            minSalary = salary;  
        }  
        if (salary > maxSalary) {  
            maxSalary = salary;  
        }  
    }  
    return [minSalary, maxSalary];  
}  
  
const salaryArray = [10000, 20000, 30000, 40000, 50000];  
const [minSalary, maxSalary] = minMaxSalary(salaryArray);  
console.log("The minimum salary is", minSalary);  
console.log("The maximum salary is", maxSalary);
```

Output:

```
The minimum salary is 10000  
The maximum salary is 50000
```

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 stores a collection of elements of the same type. Elements of an array can be accessed using their index, which starts at 0.
- Objects are a data structure that stores a collection of key-value pairs. Keys are unique identifiers for the values, and values can be of any type.
- Arrays of objects are a combination of arrays and objects. They store a collection of objects, where each object is identified by its index.

Source Code:

```
const favoriteBooks = ["Hitchhiker", "Rings", "Hunger", "Sorcerer"];
function displayFavoriteBooks(books) {
  for (const book of books) {
    console.log(book);
  }
}
displayFavoriteBooks(favoriteBooks);
```

Output:

```
Hitchhiker
Rings
Hunger
Sorcerer
```

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:

- Scope is the area in which a variable is accessible.
- Hoisting is the process of moving variable declarations to the top of their scope before they are defined.
- This means that a variable can be used even before it is declared, as long as its declaration is hoisted to the top of its scope.
- However, the value of the variable will be undefined until it is assigned a value.

Source Code:

```
function myFunction() {
  var x = 10;
  console.log("The value of x inside the function is: " + x);
}
myFunction();
console.log("The value of x outside the function is: " + x);
```

Output:

```
The value of x inside the function is: 10
/Users/mac/Desktop/5/trial.js:8
  console.log("The value of x outside the function is: " + x);
  ^
```

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:

- DOM manipulation in JavaScript is the process of changing the content, structure, or style of an HTML document using JavaScript. This can be done to add or remove elements, change the text content of elements, or apply CSS styles. DOM manipulation is used in many web applications to create dynamic and interactive user interfaces.

Source Code:

```
function changeText() {  
  var button = document.getElementById("myButton");  
  button.textContent = "Button text changed!";  
}
```

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:

- Error handling in JavaScript is a way to deal with unexpected events that occur during the execution of a program. It allows you to gracefully handle errors and prevent your program from crashing.
- To use error handling, you use the try and catch keywords. The try keyword defines a block of code that you want to test for errors. If an error occurs in the try block, the catch keyword will execute a block of code that you define to handle the error.

Source Code:

```
function validatePositiveNumber(number) {  
  if (number < 0) {  
    throw new Error('Error: Negative numbers are not allowed.');  }  
  return number;  
}  
  
try {  
  console.log(validatePositiveNumber(3));  
  console.log(validatePositiveNumber(-5));  
} catch (error) {  
  console.error(error.message);  
}
```

Output:

```
3  
Error: Negative numbers are not allowed.
```

Task-9

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

Theoretical Background:

- Asynchronous JavaScript is a programming paradigm that allows JavaScript code to run concurrently with other code.
- This can be used to improve the performance of web applications by allowing time-consuming tasks to run in the background while the user continues to interact with the page.

Source Code:

```
function simulateAsyncOperation(callback) {  
  setTimeout(function() {  
    var result = "Async operation completed";  
    callback(result);  
  }, 2000);  
}  
console.log("Start of program");  
simulateAsyncOperation(function(result) {  
  console.log(result);  
});  
console.log("End of program");
```

Output:

```
Start of program  
End of program  
Async operation completed
```

Learning Outcome:

- Understanding javascript technologies(CO1).