JavaScript Theory and Lab Assignment

JavaScript Introduction

Theory Assignment

Q1: JavaScript is a lightweight scripting language for adding interactivity to websites.
Q2: It differs from Python/Java as it runs mainly in browsers, uses dynamic typing, and focuses on web interaction.
Q3: <script> tag links JS internally or externally (via src="script.js").

Lab Assignment

HTML Example:

```
<script>
alert("Welcome to JavaScript!");
</script>
```

Variables and Data Types

Theory Assignment

Q1: Variables store data using var, let, const.
Q2: Data types – String, Number, Boolean, Null, Undefined, Object, Array.
Q3: undefined = no value assigned; null = intentional empty value.

Lab Assignment

JS Example:

```
var name = "Ajay";
let age = 25;
const student = true;
let value = null;
let score;
console.log(name, typeof name);
console.log(age, typeof age);
console.log(student, typeof student);
console.log(value, typeof value);
console.log(score, typeof score);
```

JavaScript Operators

Theory Assignment

Arithmetic, Assignment, Comparison, Logical Operators.
== compares value only, === compares value and type.

Lab Assignment

```
let a = 10, b = 5;
console.log(a+b, a-b, a*b, a/b);
console.log(a==b, a>b);
console.log(a>5 && b<10);
```

Control Flow (If-Else, Switch)

Theory Assignment

If-Else controls decision-making; switch handles multiple conditions efficiently.

Lab Assignment

If-Else Example:

```
let num = 0;
if(num>0) console.log("Positive");
else if(num<0) console.log("Negative");
else console.log("Zero");
```

Switch Example:

```
let day = 3;
switch(day){
 case 1: console.log("Mon"); break;
 case 2: console.log("Tue"); break;
 case 3: console.log("Wed"); break;
 default: console.log("Invalid");
}
```

Loops (For, While, Do-While)

Theory Assignment

For = fixed loop, While = condition based, Do-While = executes once before checking.

Lab Assignment

For Loop:

```
for(let i=1;i<=10;i++) console.log(i);
```

While Loop (sum even 1–20):

```
let sum=0,i=1;
while(i<=20){ if(i%2==0) sum+=i; i++; }
console.log(sum);
```

Do-While:

```
let n;
do{ n=prompt("Enter >10"); }while(n<=10);
```

Functions

Theory Assignment

Functions group reusable code. Declaration vs Expression differ in hoisting. Parameters allow input, return gives output.

Lab Assignment

```
function greetUser(name){ console.log(`Hello, ${name}!`); }
function calculateSum(a,b){ return a+b; }
console.log(calculateSum(5,10));
```

Arrays

Theory Assignment

Arrays store ordered values. Methods – push(), pop(), shift(), unshift().

Lab Assignment

```
let fruits = ["apple","banana","cherry"];
fruits.push("mango");
fruits.shift();
console.log(fruits);

let numbers = [1,2,3,4,5];
let sum = 0;
for(let n of numbers) sum+=n;
console.log(sum);
```

Objects

Theory Assignment

Objects hold key-value pairs. Access via dot/bracket notation.

Lab Assignment

```
let car={brand:"Tata",model:"Nexon",year:2024};
console.log(car.brand,car.model);
car.year=2025;
car.color="Blue";
console.log(car);
```

JavaScript Events

Theory Assignment

Events respond to user actions. addEventListener() binds event handlers.

Lab Assignment

```
<button id="btn">Click</button>
<script>
document.getElementById("btn").addEventListener("click",()=>alert("Button clicked!"));
</script>
```

DOM Manipulation

Theory Assignment

DOM represents HTML structure accessible via JS. Methods – getElementById(), getElementsByClassName(), querySelector().

Lab Assignment

```
<p id="text">Hello, World!</p>
<script>
let para=document.getElementById("text");
para.textContent="JavaScript is fun!";
para.style.color="blue";
</script>
```

Timing Events

Theory Assignment

setTimeout() executes after delay, setInterval() repeats execution.

Lab Assignment

```
setTimeout(()=>document.body.style.backgroundColor="lightblue",5000);
setInterval(()=>console.log(new Date().toLocaleTimeString()),1000);
```

Error Handling

Theory Assignment

try-catch-finally manages runtime errors, improving program stability.

Lab Assignment

```
try{
 let num=10;
 if(num/0===Infinity) throw new Error("Cannot divide by zero");
}catch(e){ console.log("Error:",e.message); }
```