Module 8 – Frontend – JavaScript

Theory Assignment

JavaScript Introduction

Question 1: What is JavaScript? Explain the role of JavaScript in web development.

JavaScript is a high-level, interpreted programming language used to make web pages interactive and dynamic. It can manipulate HTML and CSS, handle user events, validate forms, and fetch data from APIs. JavaScript runs on browsers and servers (Node.js), making it essential for modern web development.

Question 2: How is JavaScript different from other programming languages like Python or Java?

JavaScript is dynamically typed, prototype-based, and primarily used for client-side scripting. Python and Java are used mainly for backend development. Java is statically typed and compiled, while JavaScript is interpreted and supports event-driven programming.

Question 3: Discuss the use of <script> tag in HTML. How can you link an external JavaScript file to an HTML document?

The <script> tag is used to include JavaScript in a webpage. Inline scripts are placed between <script>...</script> tags, while external scripts are linked with <script src='file.js'></script>. External files are preferred for reusability and cleaner code.

Variables and Data Types

Question 1: What are variables in JavaScript? How do you declare a variable using var, let, and const?

Variables store data values. 'var' declares function-scoped variables, 'let' declares block-scoped variables, and 'const' declares constants that cannot be reassigned.

Example:

```
var x=1;

let y=2;

const z=3;
```

Question 2: Explain the different data types in JavaScript. Provide examples for each.

JavaScript data types include: string, number, boolean, null, undefined, symbol, bigint, object, and array.

Example:

let name='John';

let age=25;

let isTrue=true;

let arr=[1,2,3];

let obj={a:1,b:2};

Question 3: What is the difference between undefined and null in JavaScript?

undefined means a variable has been declared but no value assigned. null represents an intentional absence of value.

Example:

let a; // undefined, let b=null; // null.

JavaScript Operators

Question 1: What are the different types of operators in JavaScript? Explain with examples.

Arithmetic (+, -, *, /), Assignment (=, +=), Comparison (==, ===, !=, !==, >, <), Logical (&&, ||, !), Bitwise, and Ternary (condition ? value1 : value2).

Example:

let x=5; let y=2;

console.log(x+y); // 7

Question 2: What is the difference between == and === in JavaScript?

== performs type conversion before comparison, while === compares both value and type. Example: (5=='5') → true, (5==='5') → false.

Control Flow (If-Else, Switch)

Question 1: What is control flow in JavaScript? Explain how if-else statements work with an example.

Control flow determines how code executes. if-else executes blocks based on conditions. Example:

if(x>0){

console.log('Positive');

} else {

console.log('Negative');

}

Question 2: Describe how switch statements work in JavaScript. When should you use a switch statement instead of if-else?

switch evaluates an expression and executes the matching case. It's used for multiple discrete conditions.

Example:

switch(day){

case 1: console.log('Mon');

break;

default: console.log('Other');

}

Loops (For, While, Do-While)

Question 1: Explain the different types of loops in JavaScript (for, while, do-while). Provide a basic example of each.

for: for(let i=0;i<5;i++){console.log(i);}

while: let i=0; while(i<5){console.log(i);i++;}

do-while: let j=0; do{console.log(j);j++;}

while(j<5);

Question 2: What is the difference between a while loop and a do-while loop?

A while loop checks the condition before execution, while a do-while loop executes once before checking.

Functions

Question 1: What are functions in JavaScript? Explain the syntax for declaring and calling a function.

A function is a reusable block of code. Syntax: function add(a,b){return a+b;} Call: add(2,3);

Question 2: What is the difference between a function declaration and a function expression?

Function declarations are hoisted and can be called before they appear. Function expressions (const x=function(){}) are not hoisted.

Question 3: Discuss the concept of parameters and return values in functions.

Parameters accept input; return sends output back. Example: function sum(a,b){return a+b;}

Arrays

Question 1: What is an array in JavaScript? How do you declare and initialize an array?

An array is a collection of ordered elements.

Example:

let fruits=['apple','banana'];

let nums=[1,2,3];

Question 2: Explain the methods push(), pop(), shift(), and unshift() used in arrays.

push() adds to end, pop() removes last, shift() removes first, unshift() adds to start.

Objects

Question 1: What is an object in JavaScript? How are objects different from arrays?

Objects store key-value pairs, while arrays store ordered values.

Example:

let car={

brand:'BMW',year:2022

};

Question 2: Explain how to access and update object properties using dot notation and bracket notation.

Dot notation: car.brand='Audi'; Bracket: car['year']=2024;

JavaScript Events

Question 1: What are JavaScript events? Explain the role of event listeners.

Events are actions (click, load, keypress). Event listeners detect them and run code. Example: btn.addEventListener('click',()=>alert('Clicked!'));

Question 2: How does the addEventListener() method work in JavaScript? Provide an example.

It registers a function for an event on an element. Example: document.querySelector('button').addEventListener('click',()=>console.log('Pressed'));

DOM Manipulation

Question 1: What is the DOM (Document Object Model) in JavaScript? How does JavaScript interact with the DOM?

DOM represents HTML structure as a tree. JavaScript uses it to update elements. Example: document.getElementById('demo').innerHTML='Hello';

Question 2: Explain the methods getElementById(), getElementsByClassName(), and querySelector() used to select elements from the DOM.

getElementById('id') selects by ID; getElementsByClassName('class') returns multiple; querySelector('.class') selects first match.

JavaScript Timing Events (setTimeout, setInterval)

Question 1: Explain the setTimeout() and setInterval() functions in JavaScript. How are they used for timing events?

setTimeout(fn, ms) runs a function once after delay; setInterval(fn, ms) repeats after each delay. Example: setTimeout(()=>alert('Hi'),2000);

Question 2: Provide an example of how to use setTimeout() to delay an action by 2 seconds.

setTimeout(()=>{document.body.style.background='lightblue';},2000);

JavaScript Error Handling

Question 1: What is error handling in JavaScript? Explain the try, catch, and finally blocks with an example.

Error handling prevents code from crashing using try-catch-finally. Example: try{let x=1/0; throw new Error('Error');} catch(e){console.log(e.message);} finally{console.log('Done');}

Question 2: Why is error handling important in JavaScript applications?

It ensures the application continues running smoothly and provides user-friendly error messages instead of breaking the entire program.