```html
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>


<body>


    <button onclick="switchTrigger()">switchtrigger</button>


    <script>



        // 1. Let, Const, Var
        console.log("----------------------1. Let,Const,Var--------------------")


        var nameVar = "var variable";


        function blockScopeVar() {
            const nameConst = 'const variable';
            let nameLet = 'let variable';
            console.log(nameVar);

        }

        console.log(nameVar)
        // console.log(nameConst)
        // console.log(nameLet)

        // 2. Javascript Types
        console.log("----------------------2. JS Types--------------------")

        let typeName = "jstypes";
        let varName = "letconstvar";

        // Number
        let length = 20;
        let weight = 9.5;

        // BigInt
        let bigNum = 4323420203949320932;
        let bigNum_ = BigInt(4323420203949320932)
```

```javascript
// Boolean
let boolX = true;
let boolY = false;

// Object
const person = { firstName: "vasi", age: 25, gender: "Male" };

// Array object
const students = ["Gyanavel", "GuruPrasath", "Aaryan"];

// Date object
const date = new Date("2026-02-02");

// Undefined
let undefinedA;
let undefinedB;

// Null
let nullA = null;
let nullB = null;

// Symbol
```

```javascript
const symbolX = Symbol();

const symbolY = Symbol();



console.log("----for Symbol example x === y---");

console.log(symbolX === symbolY)


// 3.Operators




console.log("----------------------3. Operators--------------------")



console.log("-----i. Assignment Operators");

let x = 10;

x += 5;

console.log("Assignment Operator x += 5: " + x);



x -= 3;

console.log("Assignment Operator x -= 3: " + x);



x *= 2;

console.log("Assignment Operator x *= 2: " + x);
```

```
x /= 4;
console.log("Assignment Operator x /= 4: " + x);


console.log("-----ii. Arithmetic Operators");


let addition = 50 + 90;
console.log("Addition Operator 50 + 90: " + addition);
let subtract = 50 - 90;
console.log("subtract Operator 50 - 90: " + subtract);
let divOp = 50 / 90;
console.log("divOp Operator 50 / 9: " + divOp);
let reminder = 50 % 9;
console.log("reminder Operator 50 % 90: " + reminder);
let a = 50;
let incr = ++a + 1;
console.log("Increment Operator ++50 + 1: " + incr);


let b = 50;
let decr = b-- - 1;
console.log("Decrement Operator 50-- - 1: " + decr);




console.log("-----iii. Comparison operators")
let equalLoose = 50 == "50";
```

```
        console.log("Loose Equality Operator 50 == '50': " + equalLoose);


        let equalStrict = 50 === "50";

        console.log("Strict Equality Operator 50 === '50': " + equalStrict);


        let notEqualLoose = 50 != "90";

        console.log("Loose Not Equal Operator 50 != '90': " + notEqualLoose);


        let notEqualStrict = 50 !== "50";

        console.log("Strict Not Equal Operator 50 !== '50': " + notEqualStrict);


        let greaterThan = 90 > 50;

        console.log("Greater Than Operator 90 > 50: " + greaterThan);


        let greaterThanEqual = 50 >= 50;

        console.log("Greater Than or Equal Operator 50 >= 50: " +
greaterThanEqual);


        let lessThanEqual = 40 <= 50;

        console.log("Less Than or Equal Operator 40 <= 50: " + lessThanEqual);
        // 4. JS Conditional

        console.log("----------------------4. JS Conditional--------------------
")

        console.log("-----i. If Statement")

        let marks = 36;



        if (marks > 35) {

            console.log("Pass Mark ")

            console.log("Condition Passed through if statement greater than 35")

        }




        console.log("-----ii. If...else")

        var ifelsemarks = 32;



        if (ifelsemarks > 35) {

            console.log("Pass mark")

        } else {

            console.log("Fail Mark")

            console.log("Condition else block passed through ifelse statement ")

        }




        var ifelsemarks = 100;

        console.log("-----iii. If...elseif")

        if (ifelsemarks < 35) {
            console.log("fail mark")

        } else if (ifelsemarks > 35 && ifelsemarks === 100) {

            console.log("Centem")

            console.log("Condition else if with condition block passed through 
ifelseif statement ")

        }

        else {

            console.log("Pass mark")

        }



        console.log("-----iv. Ternary Operator")



        ternaryCheck = 40;



        console.log("Checking limit trough ternary operator")

        ternaryCheck > 25 ? console.log("Exceeded limit") : 
console.log("Accessed");




        //   5. JS Switch



        function switchTrigger() {



            console.log("-----i. Check through Switch statement");
            let actionSwitch = parseInt(prompt("Select Action\n1. Check Balance 
\n2. Withdraw \n3. Deposite"));



            switch (actionSwitch) {

                case 1:

                    alert("Your Bank Balance is XXXXXXXX");

                    break;

                case 2:

                    alert("Amout Deducted");

                    break;

                case 3:

                    alert("Amount Deposited");

                    break;

                default:

                    alert("Check the input and give the valid one")

            }





        }



        //  6. JS Loops



        console.log("----------------------5. JS Looping Statements--------------
------")
```

```javascript
console.log("-----i. for Statement")
let arrValu = [43, 66, 11, 22, 90];



console.log("Print Array values through for loop")
for (let i = 0; i < arrValu.length; i++) {
    console.log(arrValu[i]);


}


console.log("-----ii. while Statement")
let arrValWhile = [2, 3, 4, 5, 6, 7, 8, 8, 4, 3, 2];
console.log("Stop when the duplicate exists 2,3,4,5,6,7,8,8,4,3,2")
let arrAdditional = [];
let i = 0;
while (!arrAdditional.includes(arrValWhile[i])) {


    arrAdditional.push(arrValWhile[i]);
    i++;
}


console.log(arrAdditional);


console.log("-----iii. do while Statement");
```

```
        let num = 1;

        console.log("Print numbers from 1 to 5 using do while loop");



        do {

            console.log(num);

            num++;

        } while (num <= 5);




        console.log("----------------------7. JS Strings--------------------");



        console.log("-----i. Strings");

        let stringVal = "this is the string value";

        console.log(stringVal);



        console.log("-----ii. Template String");

        let stringValTemplate = `Backtics used for template string 3+4 => ${3 +
4}`;

        console.log(stringValTemplate);



        console.log("-----iii. Escape Character");

        let escapeChar = "here the escape character \"ESCAPE CHARACTER\"
implemented";
```

```
console.log(escapeChar);



console.log("-----iv. String length");

console.log("Length of string: " + stringVal.length);



console.log("-----v. String charAt()");

console.log("Character at index 5: " + stringVal.charAt(5));



console.log("-----vi. String charCodeAt()");

console.log("Char code at index 5: " + stringVal.charCodeAt(5));



console.log("-----vii. String codePointAt()");

console.log("Code point at index 5: " + stringVal.codePointAt(5));



console.log("-----viii. String concat()");

let concatStr = stringVal.concat(" added text");

console.log(concatStr);



console.log("-----ix. String at()");

console.log("Character at index 2: " + stringVal.at(2));



console.log("-----x. String []");

console.log("Character at index 3: " + stringVal[3]);
```

```
        console.log("-----xi. String slice()");
        console.log("Slice from index 0 to 4: " + stringVal.slice(0, 4));


        console.log("-----xii. String substring()");
        console.log("Substring from index 5 to 10: " + stringVal.substring(5,
10));


        console.log("-----xiii. String substr()");
        console.log("Substr from index 5 length 6: " + stringVal.substr(5, 6));


        console.log("-----xiv. String toUpperCase()");
        console.log(stringVal.toUpperCase());


        console.log("-----xv. String toLowerCase()");
        console.log(stringVal.toLowerCase());


        console.log("-----xvi. String isWellFormed()");
        console.log("Is well formed: " + stringVal.isWellFormed());


        const toWellformedVal = "\uD800";
        console.log("-----xvii. String toWellFormed() broken character filled
with  �");
```

```javascript
    console.log("Well formed string: " + toWellformedVal.toWellFormed());


    console.log("-----xviii. String trim()");
    let trimStr = "   trim this string   ";
    console.log(trimStr.trim());


    console.log("-----xix. String trimStart()");
    console.log(trimStr.trimStart());


    console.log("-----xx. String trimEnd()");
    console.log(trimStr.trimEnd());


    console.log("-----xxi. String padStart()");
    let padStr = "5";
    console.log(padStr.padStart(4, "0"));


    console.log("-----xxii. String padEnd()");
    console.log(padStr.padEnd(4, "0"));


    console.log("-----xxiii. String repeat()");
    console.log("JS ".repeat(3));


    console.log("-----xxiv. String replace()");
```

```
        console.log(stringVal.replace("string", "text"));



        console.log("-----xxv. String replaceAll()");

        console.log(stringVal.replaceAll(" ", "-"));



        console.log("-----xxvi. String split()");

        let splitStr = stringVal.split(" ");

        console.log(splitStr);



        // 8. JS Functions



        console.log("----------------------8. JS Function--------------------")

        console.log("-----i. Function Declaration")



        function simpleFunc() {

            let length = 20;

            let breadth = 50;

            return length * breadth

        }



        console.log("Calling function with return value along with calculations:
" + simpleFunc());
```

```
console.log("-----ii. Call Method");


function studVal(name, age) {


    return `${name} and age: ${age} and department ${this.department}`
}
console.log(studVal.call({ department: "development" }, "vasiraja", 25));



console.log("-----iii. Apply Method");


projects = ["codevamp", "automation view", "testing software"];


function printDev(dep) {


    for (let i of dep) {
        console.log(i);
    }




}
printDev.apply(null, [projects])
```

```
        console.log("-----iv. Bind Method");

        console.log("-----Bind together function and user details into one
through bind method")



        function userAccess() {

            console.log(this.name + " have the special access to trigger that
function")

            console.log("Age is: " + this.age)

        }



        const user = {

            name: "vasi",

            age: 25

        };



        const bindTogether = userAccess.bind(user);

        bindTogether();



        console.log("-----v. IIFE Method");



        (function () {

            console.log("This functionality immediately implement without any
trigger through this IIFE(Immediately Invoked Function Expression) method ")
```

```
        })();




        console.log("-----vi. Closure Method");

        console.log("Inner function able to access the variable from the outer function even outer function executed completed")



        function firstFunction() {

            let availBalance = 3400;



            function checkBalance() {

                return availBalance;

            }



            return checkBalance;

        }

        const checked = firstFunction()



        console.log(checked());





        console.log("----------------------9. JS Objects-------------------")

        console.log("-----i. Object assign");
```

```javascript
        const objectUser = {

            name: "Rahmen",

            age: 24,

            dob: new Date(14, 12, 2000)

        }

        const objectAddress = {

            city: "Madurai",

            pincode: "625535"

        }



        console.log(Object.assign({}, objectUser, objectAddress))

        console.log(objectUser)

        console.log(objectAddress)



        console.log("-----ii. Object Create along with proto")

        console.log("We can reuse the object properties into another one through
proto like inherit parent behavior into child")



        const server = {

            port: 3000,

            tech: "Nodejs",

            user: "admin"

        };
```

```
const anotherUser = Object.create(server);

console.log(anotherUser.user);

console.log(anotherUser.tech);

console.log(anotherUser.port);



console.log("-----iii. Object Entries")

const productStocks = {

    laptop: 20,

    mobile: 100,

    keypad: 120,




};



for (const i of Object.entries(productStocks)) {

    console.log(i)

}

console.log("-----iv. Object Keys")



for (const i of Object.keys(productStocks)) {

    console.log(i)
}

console.log("-----v. Object Values")



for (const i of Object.values(productStocks)) {

    console.log(i)

}



console.log("-----vi. Object get property")



const accesser = {



    firstName: "vasiraja",

    get name() {



        return this.firstName;

    }





}



console.log("Print name through object using get property")

console.log(accesser.name);
```

```
        console.log("-----vii. Object set property");



        const modifier = {

            set name(value) {

                this.firstName = value;



            }

        }



        modifier.name = "modifier name ";

        console.log("Setting name through object using set property")

        console.log(modifier.firstName);

        console.log("-----viii. Object preventExtensions and Extensible");



        const preventExtensionObj = {

            "firstPre": 23,

            "secondPre": 33,

        };

        Object.preventExtensions(preventExtensionObj);

        preventExtensionObj.thirdPre = 34;

        console.log("Below object \"thirdPre\" not added due to preventextension
function implements")

        console.log(preventExtensionObj);
        console.log("Below check true or false which preventextension implement 
or not through isextensible   ")


        console.log(Object.isExtensible(preventExtensionObj));



        console.log("-----ix. Object seal and isSealed");

        const sealObj = {

            sealA: "firstuser",

            sealB: "seconduser"

        };

        Object.seal(sealObj);

        sealObj.sealC = "thirduser";

        delete sealObj.sealA;

        sealObj.sealA = "modified user"


        console.log("Below answer is sealed so we can't add or delete properties 
in object but we can modify that ");

        console.log(sealObj);


        console.log("Check object whether sealed or not through isSealed 
property");

        console.log(Object.isSealed(sealObj));
        console.log("-----ix. Object freeze and isFreezen");

        const freezeObj = {

            freezeA: "firstuser",

            freezeB: "seconduser"

        };

        Object.freeze(freezeObj);

        freezeObj.sealC = "thirduser";

        delete freezeObj.sealA;

        freezeObj.sealA = "modified user"



        console.log("Below answer is sealed so we can't add or delete properties
in object and cannot modify though");

        console.log(freezeObj);



        console.log("Check object wheter frozen or not through isFrozen
property");

        console.log(Object.isFrozen(freezeObj));




        console.log("----------------------10. JS Array Methods-----------------
--");



        let arr = [10, 20, 30, 40, 50];

        let arr2 = ["a", "b", "c"];
```

```
console.log("----- Array length");
console.log(arr.length);


console.log("----- Array toString()");
console.log(arr.toString());


console.log("----- Array at()");
console.log(arr.at(2));
console.log(arr.at(-1));


console.log("----- Array join()");
console.log(arr.join(" - "));


console.log("----- Array pop()");
console.log(arr.pop());
console.log(arr);


console.log("----- Array push()");
arr.push(60);
console.log(arr);


console.log("----- Array shift()");
console.log(arr.shift());

console.log(arr);



console.log("----- Array unshift()");

arr.unshift(5);

console.log(arr);



console.log("----- Array delete()");

delete arr[1];

console.log(arr);



console.log("----- Array concat()");

let merged = arr.concat(arr2);

console.log(merged);



console.log("----- Array copyWithin()");

let copyArr = [1, 2, 3, 4, 5];

copyArr.copyWithin(0, 3);

console.log(copyArr);



console.log("----- Array flat()");

let flatArr = [1, [2, [3, 4]]];

console.log(flatArr.flat(2));
console.log("----- Array splice()");
let spliceArr = [1, 2, 3, 4, 5];
spliceArr.splice(2, 1, 99);
console.log(spliceArr);


console.log("----- Array toSpliced()");
let newArr = spliceArr.toSpliced(1, 1);
console.log(newArr);
console.log(spliceArr);


console.log("----- Array slice()");
console.log(spliceArr.slice(1, 3));


console.log("----- Array indexOf()");
console.log(spliceArr.indexOf(99));


console.log("----- Array lastIndexOf()");
let dupArr = [1, 2, 3, 2, 4];
console.log(dupArr.lastIndexOf(2));


console.log("----- Array includes()");
console.log(dupArr.includes(3));
console.log("----- Array find()");
console.log(dupArr.find(n => n > 2));


console.log("----- Array findIndex()");
console.log(dupArr.findIndex(n => n > 2));


console.log("----- Array findLast()");
console.log(dupArr.findLast(n => n > 2));


console.log("----- Array findLastIndex()");
console.log(dupArr.findLastIndex(n => n > 2));


console.log("----- Array toSorted()");
let sortArr = [40, 10, 30, 20];
console.log(sortArr.toSorted());
console.log(sortArr);


console.log("----- Array toReversed()");
console.log(sortArr.toReversed());


console.log("----- Array sort()");
sortArr.sort((a, b) => a - b);
```

```
        console.log(sortArr);



        console.log("----- Array reverse()");

        sortArr.reverse();

        console.log(sortArr);



        console.log("----------------------11. JS For..in , For..of--------------
------");




        console.log("----- for...in loop (index)");

        for (let index in arr) {

            console.log(index, arr[index]);

        }



        console.log("----- for...of loop (value)");

        for (let value of arr) {

            console.log(value);

        }



        console.log("----------------------12. JS map,reduce,filter--------------
------");



        const arrvalMap = [12, 33, 54, 11, 90];
        console.log("-----i. map function");

        console.log("Map function transform each data and return same lenght
values");



        const mapTraversed = arrvalMap.map((items) => {

            return items * 20

        })

        console.log(mapTraversed);



        const arrvalFilter = [1, 2, 3, 4, 5, 6, 7, 8];

        console.log("Filter function return result which condition satisfied by
that function only");



        const filterResult = arrvalFilter.filter((items) => items % 2 !== 0);

        console.log("Filtered result which are odd numbers: " + filterResult);



        const arrvalReduce = [1, 2, 3, 4, 5, 6, 7, 8];



        const reduceSumResult = arrvalReduce.reduce((prev, items) => {

            return prev + items;

        }

        )
        console.log("Result of reduce function implement sum of num in array
values: " + reduceSumResult)



        const arrvalEach = [10, 20, 30, 40, 50];



        console.log("-----iv. forEach function");

        console.log("forEach function executes a provided function once for each
array element (does not return new array)");



        arrvalEach.forEach((item, index) => {

            console.log(`Index ${index} has value: ${item}`);

        });



        console.log("-----v. reduceRight function");

        console.log("reduceRight works like reduce but traverses array from right
to left");



        const reduceRightResult = arrvalReduce.reduceRight((prev, item) => prev +
item);

        console.log("Result of reduceRight (sum from right to left): " +
reduceRightResult);



        console.log("-----vi. every function");

        console.log("every checks if all array elements satisfy the condition and
returns true/false");
```

```
        const everyResult = arrvalFilter.every(item => item > 0);

        console.log("Are all numbers > 0 " + everyResult);



        console.log("-----vii. some function");

        console.log("some checks if at least one element satisfies the
condition");



        const someResult = arrvalFilter.some(item => item > 5);

        console.log("Is at least one number > 5 " + someResult);



        console.log("-----viii. Array.from function");

        console.log("Array.from converts array-like or iterable objects into an
array");



        const strExample = "Vasi";

        const arrayFromStr = Array.from(strExample);

        console.log("Array from string 'Vasi': " + arrayFromStr);



        const setExample = new Set([1, 2, 3, 4]);

        const arrayFromSet = Array.from(setExample);

        console.log("Array from Set: " + arrayFromSet);
        console.log("----------------------13. JS Regular Expressions------------
--------");



        let instruction = "Hello this is Vasi, age is 25"



        console.log("-----i. Square brackets");

        console.log("Check globally a or e or o using --/[aeo]/g-- exists in
between this")



        let squareBracketReg = /[aeo]/g;

        console.log(squareBracketReg.test(instruction));



        console.log("-----ii. [^] negations")

        console.log("Check globally a or e or o using --/[^Hello]/-- never exists
in between this")

        let negationBracketReg = /^[is]/;



        console.log(negationBracketReg.test(instruction))



        console.log("-----iii. + - match one or more proceding character")



        let matchoneormore = /\d+/g

        console.log(matchoneormore.test(instruction))

        console.log("-----iv. * - match one or more proceding character")
```

```
let matchzeroormore = /\d*/g
console.log(matchzeroormore.test(instruction));


console.log("-----v. ? - optional zero or none")


let optionalCase = /^[is]?/;
console.log(optionalCase.test(instruction));


console.log("-----vi. ? - match stringword")


let matchWord = /^Hello/;
console.log(matchWord.test(instruction))


console.log("-----vii. $ - match End word")


let matchWordEnd = /25$/;
console.log(matchWordEnd.test(instruction))


console.log("-----viii. () - Group between many cases this or that ")
```

```
let thisorthat = /(Hello | test)/g;

console.log(thisorthat.test(instruction))



console.log("-----ix. {} exact quantity of words ");



let countQuantity = /\d{2}/;

console.log(countQuantity.test(instruction));

console.log("-----x. . - match any single character");

let matchany = /./g;

console.log(matchany.test(instruction));




console.log("-----xi. | - OR operator");



let orOperator = /Vasi|Raja/;

console.log(orOperator.test(instruction));



console.log("-----xii. Modifiers - g, i, m");



console.log("Global modifier (g) example:");

let globalMod = /is/g;

console.log(globalMod.test(instruction));
```

```
        console.log("Case-insensitive modifier (i) example:");
        let caseInsensitive = /hello/i;
        console.log(caseInsensitive.test(instruction));


        console.log("Multiline modifier  ");
        let multilineText = `Hello
this
        is
        Vasi
this
        `;
        let multilineMod = /^this/m;
        console.log(multilineMod.test(multilineText));


        console.log("-----xiii. \\d - match digit");
        let digitCheck = /\d/g;
        console.log(digitCheck.test(instruction));


        console.log("-----xiv. \\w - match word character (letter, digit, _)");
        let wordChar = /\w/g;
        console.log(wordChar.test(instruction));


        console.log("-----xv. \\s - match whitespace");
        let whiteSpace = /\s/g;
        console.log(whiteSpace.test(instruction));



        console.log("----------------------14. JS RegEx Methods-------------------");



        console.log("-----i. test() - check if pattern exists (true/false)");

        let testMethod = /Vasi/;

        console.log(testMethod.test(instruction));



        console.log("-----ii. match() - return array of all matches");

        let matchMethod = instruction.match(/\d+/g);

        console.log(matchMethod);



        console.log("-----iii. exec() - return first match object with details");

        let execMethod = /Vasi/;

        console.log(execMethod.exec(instruction));



        console.log("-----iv. replace() - replace matched text");

        let replaceMethod = instruction.replace(/Vasi/, "Raja");

        console.log(replaceMethod);



        console.log("-----v. split() - split string by pattern");

        let splitMethod = instruction.split(/\s/);

        console.log(splitMethod);
        console.log("----------------------15. JS Promise - Simple Example-------
-------------");

        console.log("-----i. Simple Promise creation");

        let simplePromise = new Promise((resolve, reject) => {
            let success = true;
            if (success) {
                resolve("Promise resolved successfully!");
            } else {
                reject("Promise rejected!");
            }
        });

        simplePromise
            .then(result => console.log("Then: " + result))
            .catch(error => console.log("Catch: " + error));

        console.log("-----ii. async/await ")
        console.log("Using this we can handle delayed function for parallel
access")
```

```javascript
function fetchMessage() {

    return new Promise((resolve, reject) => {

        setTimeout(() => {

            resolve("Hello from async function!");

        }, 1000);

    });

}



console.log("-----ii. Simple Async/Await function");



async function showMessage() {

    let message = await fetchMessage();

    console.log(message);

}



showMessage();
```

```
    </script>

</body>



</html>
```