**JS is a programming language,We use it to give instructions to the computer**

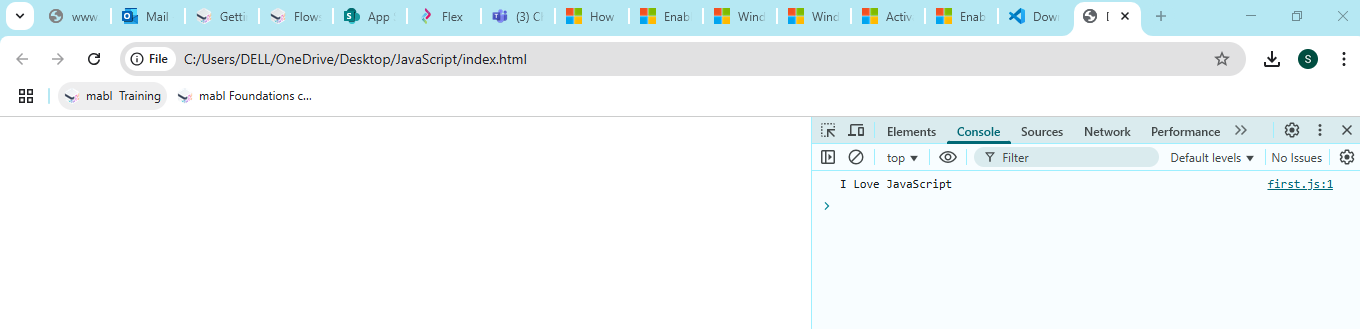
console.log("Apana SK"); // O/P :: Apana SK

**To Run Simple Codein JS and Run in browser**

* Create Folder on desktop , open that folder inside VS Code
* Create newfile and save it with .js extension
* Write js code ,example : console.log(“I Love JS!!”);

**Steps To Run JS file file on browser**

* Create simple file with .html extension , Example index.html in same folder
* To create HTML boiler plate code press **Shift+!**
* Once you get boilerplate code write down <script></script> tag inside body tag
* Write file name inside script tag like this
* <script src="first.js"></script>
* src= file name, save the code
* Run the HTML code , will get open browser
* Select inspect on browser , open console will get our output



**Variables in JS :** Variables are containers for data.

// Variables

fullName="Snehal kakade";

console.log(fullName);

age=25;

console.log(age);

a=null;

console.log(a);

b=undefined;

console.log(b);

price = true;

console.log(price);

**Variable Rules**

* Variable names are case sensitive; “a” & ”A” is different.
* Only letters,digits,underscore(\_) and $ is allowed.(Not even space).
* Only a letter,underscore(\_) or $ should be 1st character.
* Reserved words cannot be variable names.

**Types of Variables**

**var :** Variable can be re-declared & updated. A global scope variable**.(Not in Use ,used before 2015)**

//var - it will work but not a good practice ,don't use like this

var age=24;

var age=45;

var age=34;

console.log(age) // op 34

**let:** Variable Cannot be re-declared but can be updated. A block scope variable.

let x;

console.log(x); //op undefined

//it work and also a good practice

let agee = 45;

agee =54;

agee =35;

console.log(agee); // op 35

let age = 25;

let age = 43;

let age = 54; //op Uncaught SyntaxError: Identifier 'age' has already been declared

**const :**Variable cannot be re-declared or updated. A block scope variable.

//const

const PI =3.14;

console.log(PI); //op 3.14

const a;

console.log(a); //op Uncaught SyntaxError: Missing initializer

**DataTypes in JS** There are Two Typesof DataTypes

**1)Primitive DataTypes**

* There are 7 datatypes in JS
* Number , String , Boolean, undefined, Null, BigInt, Symbol

**Number**

* //datatype Number
* let agge = 30;
* console.log(agge); //op 30
* console.log(typeof agge); //op number

**String**

//datatype String

let schoolName = "SVPM";

console.log(schoolName); //op SVPM

console.log(typeof schoolName); //String

**Boolean**

//datatype Boolean

let isFailed= false;

console.log(isFailed); //op false

console.log(typeof isFailed); //op Boolean

**Null**

//datatype Null

let valuees= null;

console.log(valuees); //op null

console.log(typeof valuees ); //op object

**Unidentified**

//datatype Unidentified

let prices;

console.log(prices); //op undefined

console.log(typeof prices ); //op undefined

**BigInt**

//datatype BigInt

let y = BigInt("1234");

console.log(y); //op 1234n

console.log(typeof y); //op bigint

**Symbol**

//datatype Symbol

let z = Symbol("Hello World!!@$!!");

console.log(z); //op Symbol(Hello World!!@$!!)

console.log(typeof z); //op symbol

**2) Non-Primitiva DataTypes : Object(Arrays,Functions)**

- Object combination of key : Value pair example fullName: Snehal

- To print we can use two types

1) obj.key

console.log(student.ages);

2)obj[“key”]

console.log(student["ages"]);

const student =

{

 fullNames : "Snehal Kakade",

 ages: 26,

 cgpa:8.48,

 isPass:true,

};

console.log(student.ages); //op 26 we can print using twotypes,using first type

console.log(student["ages"]); // op 26 using second type

To Update values from Object or change values

//To update values to new values wecan do like

student ["ages"] = student["ages"]+1;

console.log(student["ages"]) //op 26+1=27 is op

student["name"]="Nikhil Kakade";

console.log(student["name"]); //op Nikhil Kakade

**To Find out Type ofDataTypes**

//To findout Typeof datatypes

console.log(typeof student); //op object

To print whole object

//To print whole object

console.log(student); //op all key:values from student

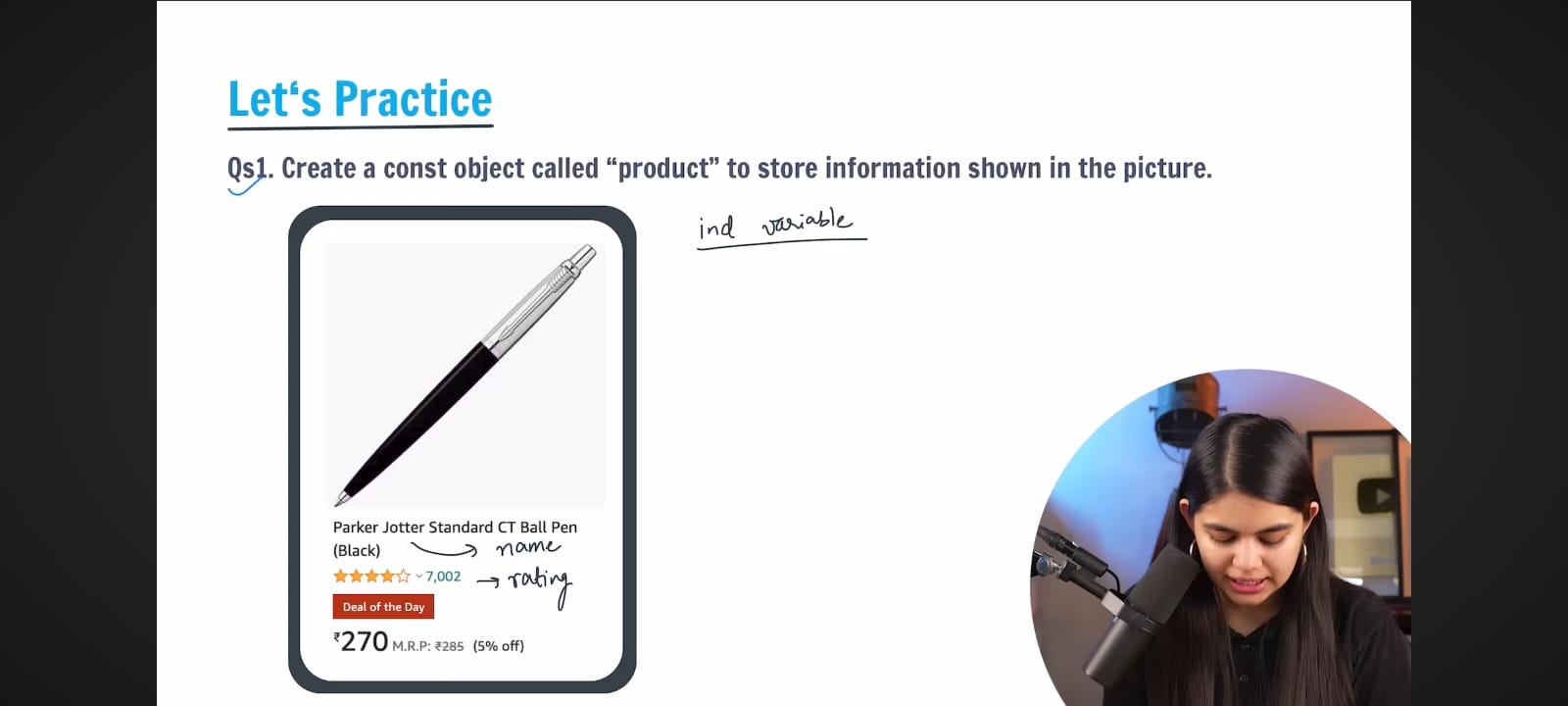
1. Output : *{fullNames: 'Snehal Kakade', ages: 27, cgpa: 8.48, isPass: true, name: 'Nikhil Kakade'}*
   1. **ages**: 27
   2. **cgpa**: 8.48
   3. **fullNames**: "Snehal Kakade"
   4. **isPass**: true
   5. **name**: "Nikhil Kakade"
   6. [[Prototype]]: Object

**VIMP** let == we can update values

const == we can’t update values

**const obj== we can update key**

**Practice TEST 1**



const product =

{

    name: "BallPen",

    rating: 4,

    offer:5,

    price:270

};

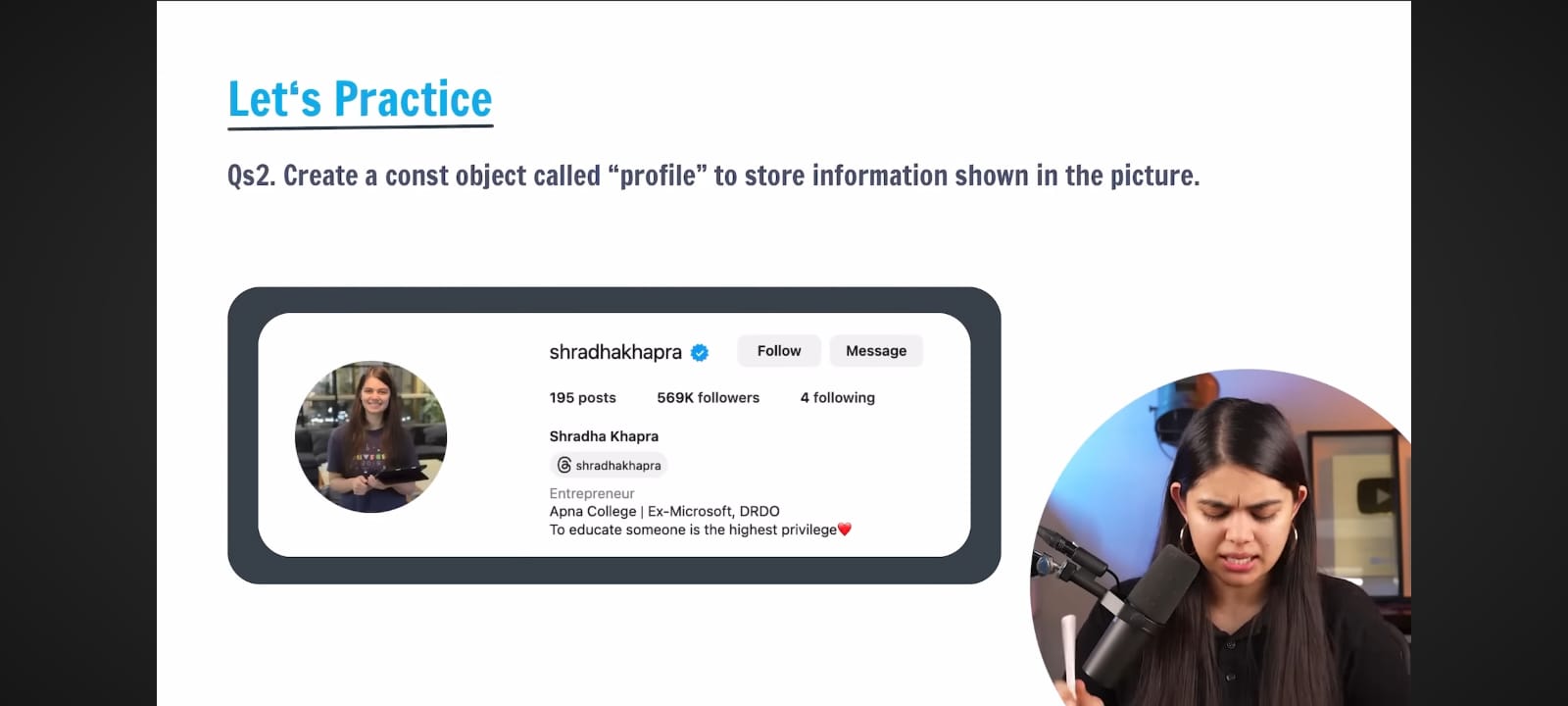
console.log(product);

console.log(typeof product);

**Output :** {name: 'BallPen', rating: 4, offer: 5, price: 270}name: "BallPen"offer: 5price: 270rating: 4[[Prototype]]: Object

practicetest1.js:11 object

**Q2.**

****

const profile=

{

  names : "Shraddha Khapara",

  isfollow: true,

  posts : 195,

  followers : 569,

  following : 4,

  bio: "Apna College|Ex-Microsoft,DRDO To educate some is thehighest privilege",

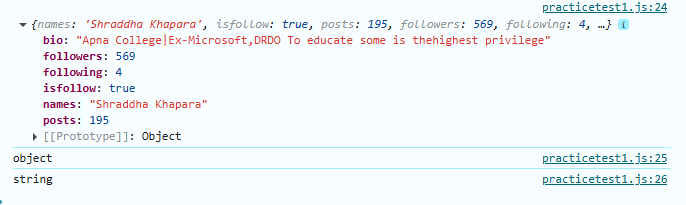
};

console.log(profile);

console.log(typeof profile);

console.log(typeof profile["names"]); //to findout typeof values from object profile //op string

**Output :**

****

**Operators and Conditional Statements**

Comments in JS : Part f code which is not executed

//This is asingle line comment

/\*This is a multiline comment \*/

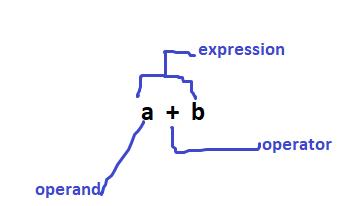
**Operators in JS**

Used to perform some operation on data

**Types of Operators:**

1. Arithmetic Operators : **+ , - ,\*,/,%,\*\***
2. Unary Operators **: pre and post increment and decrement**
3. Assignment Operators :**= , +=, -=, \*=, /= , %=, \*\*=**
4. Comparison Operator **: == ,!= , === ,!==**
5. Logical Operators**: && , || ,!**
6. Ternary Operators **condition ? True Output : False Output;**

**1)Arithmetic Operators + , - ,\*,/**



* Modulus % (output is reminder)
* Exponentiation \*\* (Example raise to power a\*\*b means 5\*\*2

**Example :**

//Arithmetic operator

let a = 10 ;

let b = 5;

console.log("a=",a , "b=", b); // op a=10 b=5

console.log("a + b = ", a+b); //+ operator

console.log("a - b = ", a-b); //- opertor

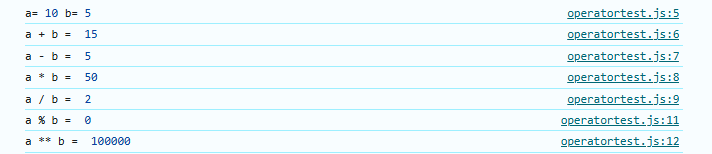
console.log("a \* b = ", a\*b); //\* operator

console.log("a / b = ",  a/b); // /operator

console.log("a % b = ", a%b); // Modulus % op 0

console.log("a \*\* b = ", a\*\*b); // Exponential 10\*\*5

Output



**2)Unary Operators :**

* **Increment Operator ++**

1. Pre Increment ++a (increment by 1 then store)
2. Post Increment a++ (first store the value then increment by 1)

* **Decrement Operator - -**

1. Pre Decrement - - a (Decrement by 1 then store)
2. Post Decrement a - - (first store then decrement by 1)

Example

//Increament and Decrement operator

//pre increment operator

console.log("++a =", ++a); //op 11

console.log("++a + ++b = ",++a + ++b);// op 18

let c = 2;

let d = 4;

//post increment

console.log("c++ =", c++); //op 2

console.log("c++ + d++ = ",c++ + d++);//op 7

console.log("c =", c);

console.log("d =", d);

//pre decrement operator

console.log("--c + --d = ", --c + --d); //--4 + --5 =3+4=op 7

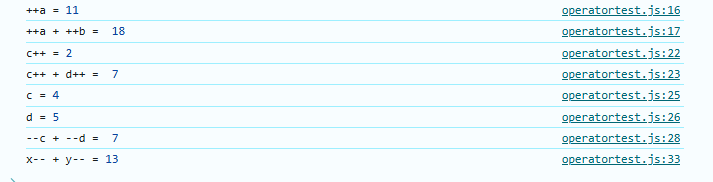
//post decrement operator

let x=6;

let y=7;

console.log("x-- + y-- =",x-- + y--);//6+7=op 13

**OutPUT**



1. **Assignment Operators = , +=, -=, \*=, /= , %=, \*\*=**
   * + = equals to
     + += means if k= 4 then k+= is k=k+any value like 5 so k=4+5=9
     + -= means k=k-4
     + \*= means k=k\*4
     + /= means k= k/4
     + %= means k= k%4
     + \*\*= means k= k\*\*4

**Same logic for all remaining assignment operators**

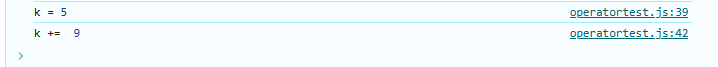
let k = 5;// = equal to operator

console.log("k =", k);

k += 4; //k=k+4=5+4=9  //+= Oprator

console.log("k += ", k);

**Output**

****

k -= 4; //k=k-4=5-4=1  //-= Oprator

console.log("k -= ", k);

**Output**



k \*= 4; //k=k\*4=5\*4=20  //\*= Oprator

console.log("k \*= ", k);

**Output k \*= 20**

k /= 4; //k=k/4=5/4=1.25  ///= Oprator

console.log("k /= ", k);

**Output k /= 1.25**

k %= 4; //k=k%4=5%4=1 //%= Oprator

console.log("k %= ", k);

**Output k %= 1**

k \*\*= 4; //k=k\*\*4=5\*\*4= 5\*5\*5\*5=625 //\*\*= Oprator

console.log("k \*\*= ", k);

**Output k \*\*= 625**

**4)Comparison Operators : output of comparison operator is always Boolean values**

* **Equal to ==** If both variables have different “datatypes”but “same values” then output is "true"
* **Not equal to !=**
* **Equal to & type ===** If both variables have differen“datatypes”but “same values” then output is "false"
* **Not Equal to & type !==**
* **>, >= , < , <=**

//4) Comparison Operators == ,!=, === , !== , - >, >= , < , <=

let p = 5;

let q = "5";

//equal to operator ==

//p is "number" q is "string" but both have same value so output is true

//both variables have different datatypes but same values then output is "true"

console.log(" p == q ",p == q); //true

//Not equal to operator !=

console.log(" p != q ",p != q); //false

//Equal to & type === (Strictly equal to)

//p has data type number and q has data type string so op is false

//both variables have different datatypes but same values then output is "false"

console.log(" p === q ",p === q); // false

//Not Equal to & type !== (Strictly not equal to)

console.log(" p !== q ",p !== q); //true

//> greter than

console.log(" p > q ",p > q); // false

//>= greter than equal to

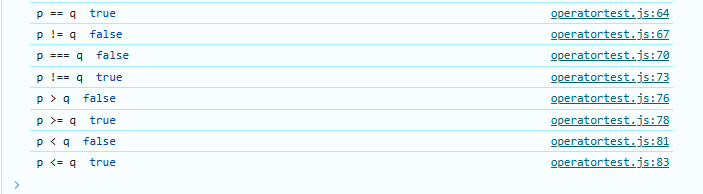
console.log(" p >= q ",p >= q); // true

// < less than equal to

console.log(" p < q ",p < q); // false

// <= less than eual to

console.log(" p <= q ",p <= q); // true

****

**5)Logical Operators**

* Logical AND &&
* Logical OR ||
* Logical NOT !

|  |  |  |  |
| --- | --- | --- | --- |
| Condition 1 | Condition 2 | && Logical AND | ||Logical OR |
| T | T | T | T |
| F | T | T | T |
| T | F | T | T |
| F | F | T | F |

//5) Logical Operator

//LOGICAL AND  && , OR || and NOT !

let m = 6;

let n = 5;

console.log("cond1 && cond2 ", m>n && m===6); //true

console.log("cond1 || cond2 ", m<=n || n!=5); //F || T op false

console.log("(6 > 8)", !(m==8)); //true

**6)Ternary Operators (Conditional Operator)** This is the only JS operator which takes three operands

**Syntax :** condition ? “True Output” : “False Output”;

//6)Ternary Operators

let num1 = 30;

let num2 = 50;

num = num1>num2 ? "30 is smaller number" : "50 is Larger Number";

console.log(num);

Output: 50 is Larger Number

**Conditional Statements**

To implement some condition in the code.

1. **If Statements : :**

* If condition is true then only we get output other wise it doesn’t show any output.
* We can use multiple if statement in single code.
* If first ifstatement is false it will check second if condition,if second false then it will go too third

**Syntax :**

if (Condition)

{

printing statement }

//Conditional Statements

//If Statement

let age = 21;

if (age > 18)

{

    console.log("You can Vote")

} // OP You can vote

let x = "pink";

if(x === "black")

{

    console.log("Test1 Passed");

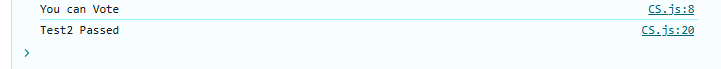
} //this test doesn't giveoutput becausecondition is failed , because both values have same data types but differnt values

if (x === "pink")

{

    console.log("Test2 Passed")

}

****

1. **If Else Statements**

* If block condition is false then only it will execute else condition

**Syntax**

If (condition)

{

Printing statement 1

}

else{

printing statement 2

}

//If Else Statements

//write a code to print even and odd numbers

 let num = 5;

 if(num%2 == 0)

 {

    console.log(num,"is Even Number");

 }

 else

 {

    console.log(num, "is Odd Number");

 }//OP 5 is odd number

//one more example for if else

    let mode = "bright";

    let color;

    if(mode==="White")

    {

        color = "Black";

    }

    else{

        color = "White";

    }

    console.log(color); //Output : White

**3)Else if Statements**

**- If first if condition is false then itwill go to 2nd 2nd is false it will go to third 3rd also false then it will execute else block**

**Syntax :**

If (condition)

{

Printing statement 1

}

else if(condition)

{ Printing statement 2}

else if(condition)

{ Printing statement 3 }

else {

Printing statement 4}

 //3) Else-if

 let ages = 58;

 if(ages < 20)

 {

    console.log("They are students");

 }

 else if(ages<40)

 {

    console.log("They are Emplyoee");

 }

 else if(ages<=60)

 {

    console.log("They are retired Persons");

 }

 else{

    console.log("They are aged persons");

 }

**Output : They are retired Persons**

**4)Switch Statement**

//4)Switch

//Mon-7am

//Tue-Thurs-4m

//Fri-9am

//Sat-Sun-8am write a code for this given datausing switch statement

let day="Saturday";

switch(day)

{

   case 'Monday':

      console.log("7am");

      break;

   case 'Tuesday':

   case 'Wednesday':

   case 'Thursday':

      console.log("4am");

      break;

   case 'Friday':

      console.log("9am");

      break;

   case 'Saturday':

   case 'Sunday':

      console.log("8am ");

      break;

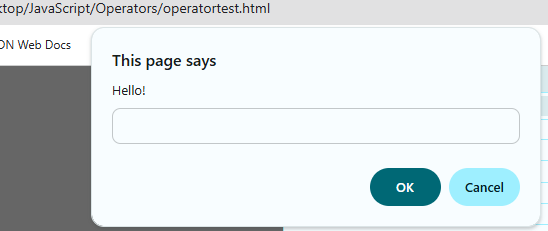
   default:

      console.log("7am- Happy weekend!!");

}

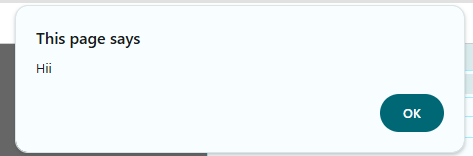
**Prompt : Used to take input from user**

Example : prompt(“Hello”);

****

**Alert : Used to show the Pop up**

Example : alert("Hii");



**Practice Test 2**

1. Get user to input a number using prompt(“Enter a number:”).Check if the number is a multiple of 5 or not.

let user\_num = prompt("Enter a number :");

console.log(user\_num);

if(user\_num % 5 == 0)

{

    console.log(user\_num,"is a multiple of 5");

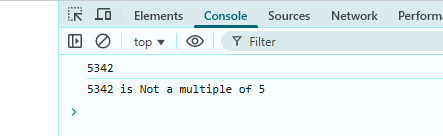
}

else

{

    console.log(user\_num, "is Not a multiple of 5");

}



Q2. Write a code which gives grades to students according to their scores.

* 90-100 , A
* 70-89,B
* 60-69,C
* 50-59,D
* 0-49,F

let user\_score = prompt("Enter a  Student Score :");

console.log(user\_score);

if(user\_score >=90 && user\_score<=100)

{

    console.log(user\_score, "Grade A");

}

else if(user\_score >=70 && user\_score<=89)

{

    console.log(user\_score, "Grade B");

}

else if(user\_score >=60 && user\_score<=69)

{

    console.log(user\_score, "Grade C");

}

else if(user\_score >=50 && user\_score<=59)

{

    console.log(user\_score, "Grade D");

}

else //if(user\_score >=0 && user\_score<=49)

{

    console.log(user\_score, "Grade F");

}

**Loops in JS**

**Loops are used to execute a pieceofcode again and again.**

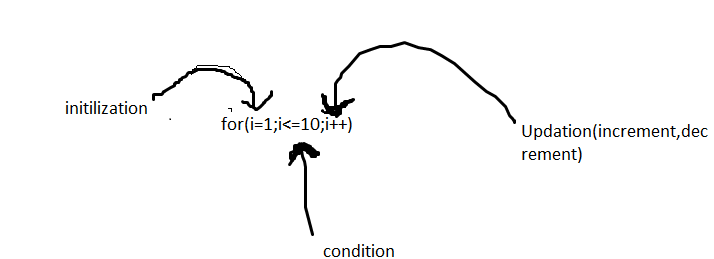
**Types of Loops in JS**

1)For Loop

2)While loop

3)Do While Loop

**1)For Loop**

****

Syntax : for(i=1;i<=10;i++)

{

console.log(“Hiii”);

}

**Example**

//for loop

//Print 1 to 5

for(let i =1 ; i<=5;i++)//5time executes

{

    console.log("Hello JS");

}

Output: 

//Calculate sum of 1 to 100

let sum=0;

let n = 100;

for(let i=1;i<=n;i++)

{

    sum=sum+i;

}

console.log("sum",sum);

Output : sum 5050

//Calculate sum of 1 to 100

let sum=0;

let n = 100;

for(let i=1;i<=n;i++)

{

    sum=sum+i;

}

console.log("sum",sum);

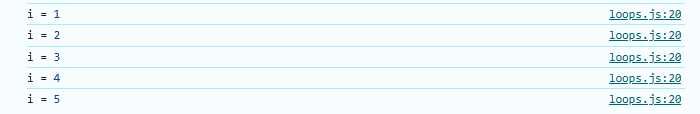
//Print 1 to 5

for(let i=1;i<=5;i++)

{

    console.log("i =", i);

}

**Output : **

**Infinite Loop : a loop that never ends**

* infinite loop will execute in case of false condition
* Never ever run the infinite loop in real time it will doesn't stop execution it will crash your code.

Example :

for(let i=1;i>=0;i++)

{

    console.log("i =", i);

}

Output : loop will execute continueously.

**2) While Loop – while loop doesn’t work for false condition**

**Syntax :**

initilization

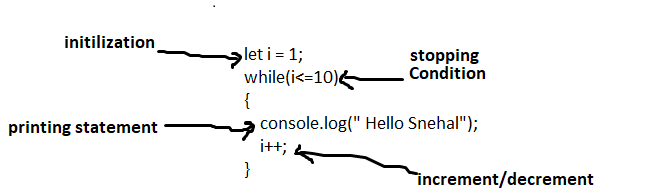
while(condition)

{

Printing statement ;

increment or decrement;

}



**Example:**

//2) While loop

let i = 1;

while(i<=10)

{

    console.log(" Hello Snehal");

    i++;

}

**Output :** 

1. **do While loop – do while loop run at least once for false condition.**

|  |  |
| --- | --- |
| **While loop** | **Do while loop** |
| 1. while loop doesn’t run for false condition**.** | 1. do while loop run at least once for false condition**.** |
| **Example**  let i = 1;  while(i>=10)  {      console.log(" Hello Snehal");      i++;  }  **Output :** it doesn’t print anything because condition is false.  Because 1 is not greater than or eualto 10 | **Example**  //3) do while loop  let m =1;  do  {      console.log("Apna College");      m++;  }  while(m>=10);  Output : Apna College  It will ru |

**Syntax**

initilization

do

{

Printing statement

increment/decrement;

}

while(condition);

**Example**

//3) do while loop

let m =1;

do

{

    console.log("Apna College");

    m++;

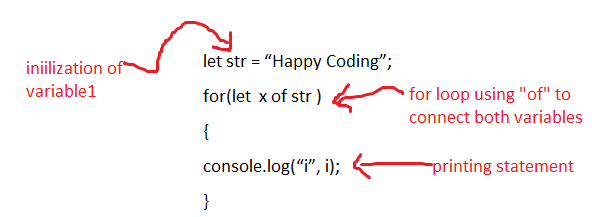
}

while(m<=10);



**4)for of loop**

* **used on “string” and “arrays” to print each character of that string**
* it will print space as well if string have space.
* if ther is “double character ”in a single string it will print single char but it will print count of that char Ex 

****

**Syntax**

let str = “Happy Coding”;

for(let x of str )

{

console.log(“x”, x);

**Example**

//4) for-of loop

let str = "Happy Coding";

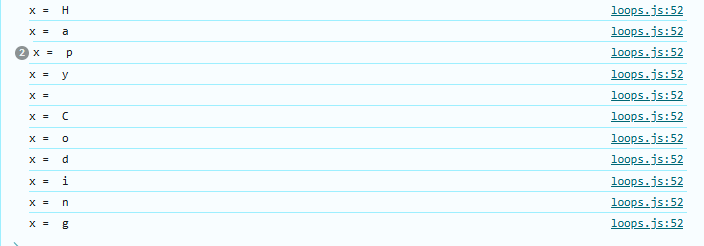
for(let x of str)

{

    console.log("x = ", x);

}

**Output**



**Write a code to print size and charater of String “HelloJS”**

//for of loop to print size

let size=0;

let str1="Hello JS"

for(let val of str1)

{

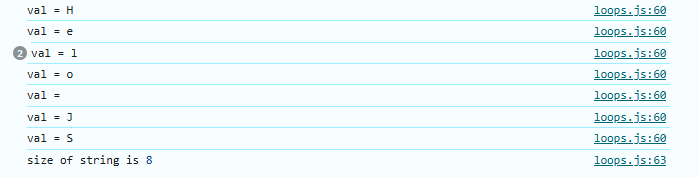
    console.log("val =",val);

    size++;

}

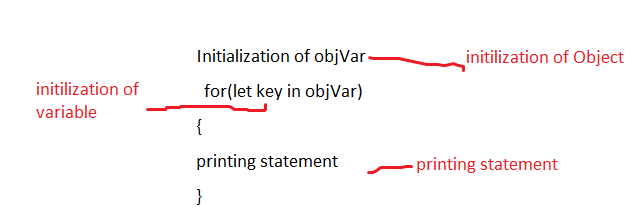
console.log("size of string is", size);

Output :



**5)for in loop**

**“for in loop” used for “objects” and arrays**

****

**Syntax**

Initialization of objVar

for(let key in objVar)

{

printing statement

}

**Example**

//for in loop used for objects and arrays

//object iscollection of "key" and "values"

 let student =

 {

    name : "Snehal Kakade",

    age :26,

    place : "Pune",

    cgpa : 8.48

 };

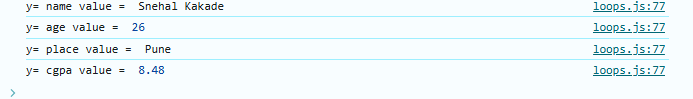
 for(let y in student)

 {

    console.log("y=",y,"value = ",student[y]);

 }

**Output**

****

**Practice Test 3**

//Q1 Print all \numbers from 0 to 100

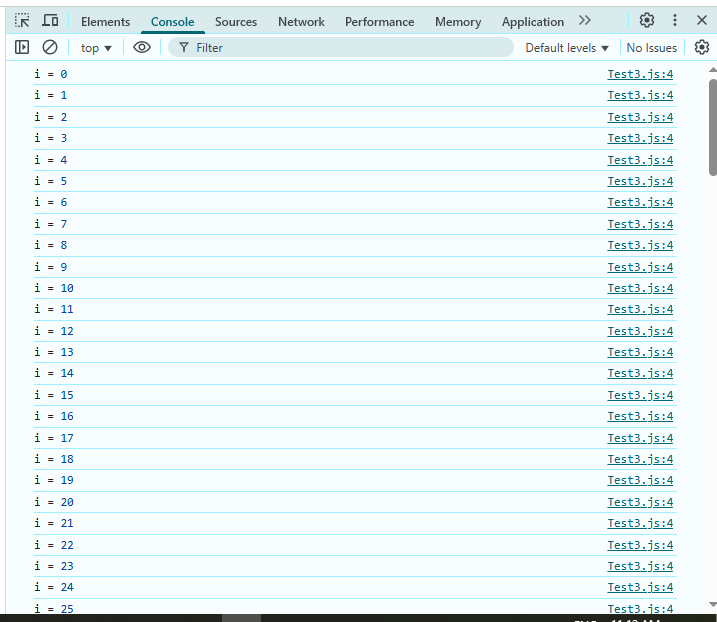
for(let i=0;i<=100;i++)

{

    console.log("i =",i);

}

**Output : it will print 1 to 100 number like this**

****

**Q2.**

//Q2 Print all the even numbers from 0 to 100

for(let num=0;num<=100;num++)

{

    if(num%2===0)

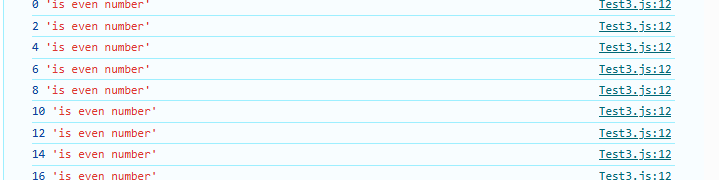
    {

        console.log(num,"is even number");

    }

}

**Output : it will print even numbers from 0 to 100 like this**

****

**Q3.**

//Q3 Create a game where you start withany random game number.Ask the

//user to keep guessing the game number until the user enters correct value.

let gameNum=25;

let userNum = prompt("Guess the game Number");

while(userNum  != gameNum)

{

   userNum=prompt("You entered the wrong the number, Guess again");

}

 console.log(userNum,"You entered the right number");

 //Output while loopwill run untill user is nott entering correct value

**Output : it will until user is not enetering correct value once user enter correct value it will show op as “**You entered the right number**”**

**String**

String is a sequence of characters used to represent text

Strings are immutable(we cant change).

String index will start from 0

# **Create String**

let str = “Happy Weekend!!”;

# **String Length**

Str.length

# **String Indices**

str[0],str[1],str[2]

//Strings

let str = "Apana College";

console.log(str[6]) //Op C //to print indices

console.log(str.length) //to print lenght// OP 13

**1) Template Literals in JS -** A way to have embedded expressions in strings

**`This is a template literal`** ======= create a string using back tick(first button from keyboard below escape)

//Template literals in JS

//using backtick symbol `This is a template literal`

let specialString = `This is a template literal`;

console.log(specialString); //op This is a template literal

console.log(typeof specialString ); //to print type ofstring //op String

template literals for object

//using Template literals for objects

let obj = {

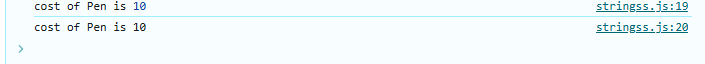
    item : "Pen",

    price : 10,

};

console.log("cost of",obj.item ,"is" ,obj.price);//without template literals

console.log(`cost of ${obj.item} is ${obj.price} `);//using template lietrals

**Output**

**Using template literals write like this ,**

* No need to use any comma ,double inverted just inside back tick write like this

console.log(`cost of **${obj.item}** is **${obj.price}** `);//using template litrals

**2)String Interpolation**

To create strings by doing substitution of placeholders

* Create a string using backtick ``
* `string text **${expression}** string text`

//String Interpolation

let specialString1 = `This is a template literal ${1+2+3}`;

console.log(specialString1); //op This is a template literal 6

Output : This is a template literal 6

**Escape Characters**

* **\n** used for new line
* **\t** used for tab space (to print space)

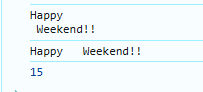
//Escape Characters

console.log("Happy \n Weekend!!"); //for new line

let str1 =("Happy\tWeekend!!"); //for tab space

console.log(str1);//for tab space

console.log(str1.length);



**String Methods in JS**

These are built in functions to manipulate a string

* str.toUpperCase()
* str.toLowerCase()
* str.trim() //remove whitespaces from start and end of string.
* str.slice(start,end?) //returns part of string excluding start and end
* str1.concat(str2) //joins str2 with str1
* str.replace(searchVal,, newVal)
* str.charAT(idx)

1. Why Strings are immutable in JS

Ans : If we take this example , Here it should not give you the output as upper case **because any method from string should not change their original string** means it should not change “Happy Testing**”, it will writtern new value of that string like str3=str2.toUUpperCase**, it will written new value inside str3.

let str2 = "Happy Testing";

str2.toUpperCase();

console.log(str2); //op Happy Testing

**Examples of String Methods**

//String Methods

let str2 = "      Happy Testing     ";

str3= str2.toUpperCase(); //toUpperCase();

console.log(str3);//toUpperCase(); Method

console.log(str2.toLowerCase());//toLowerCase() Method

console.log(str2.trim()); //trim() Method

let str4 = "123456789"

//it will print values in between mentioned index or numbers

console.log(str4.slice(1,5)); //slice(start,end?) Method //op 234

To join two string we used **concat** method we can also use + symbol

let str5 = "Hello";

let str6 = "World!!";

console.log(str5.concat(str6)); //concat() method used to add to strings

//op HelloWorld!!

**replace Method : replace single time only**

//replace method

let str7 = "HelloHello";

console.log(str7.replace("He","Ma"));//op MalloHello

**replace All Method : replace all the mentioned values from string**

let str7 = "HelloHello";

console.log(str7.replace("He","Ma"));//op MalloHello

console.log(str7.replaceAll("lo","p"));//op HelpHelp

**charAt Method**

//charAt()

let str8 ="ILoveJS";

console.log(str8.charAt(5));//J

**Practice Test 4**

Q1 prompt the user to enter their full name.Generate a username for them based on the input.Start username with @,followed by their full name and ending with the full length.

Eg: user name=”snehalkakade”, username should be “@snehalkakade12”

let str = prompt("Enter Your FullName");

console.log(str);

let str2="@" + str + str.length;

console.log(str2);

Output : @snehalkakade12

**Arrays**

* **Arrays : Collection of item of similar datatypes**
* Type of **array** is **object**
* **Method** is used to perform some task replace,charAt etc
* **Property** is used to show values example length etc

**Example**

let heros = [“Snehal”, “Pooja”, “Monika”, “Nikhil”, “Aditya”];

let marks = [97, 76, 87, 89,90];

//array of marks

let marks = [98, 97, 94, 89, 85];

console.log(marks);//To print array

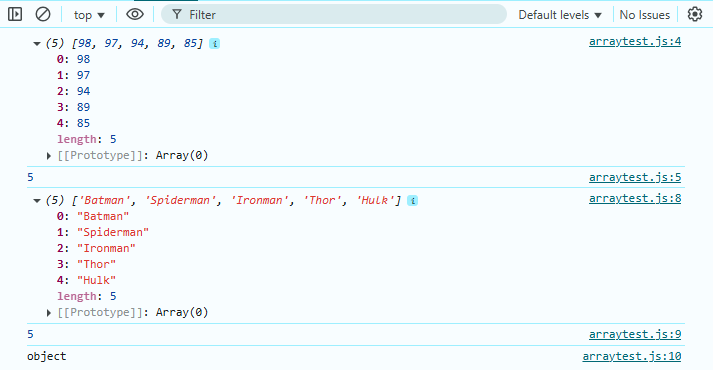
console.log(marks.length);//length is a property

let heroes = ["Batman", "Spiderman", "Ironman", "Thor", "Hulk"];

console.log(heroes);

console.log(heroes.length);//length is a property

console.log(typeof(heroes));;//to print typeof array



**Array Indices**

* If we are try to print the value out of index it will show output as “undefined”.
* Array Index start from 0

Example



//array to print index

let price = [97, 95, 82, 91, 90, 76];

console.log(price[4]); //op 90

console.log(price[8]);//op undefined



**Replace values from array using index**

//To change the values from array replace 91 with 52 in price array

let price = [97, 95, 82, 91, 90, 76];

price[3]=52;

console.log(price);

output  [97, 95, 82, 52, 90, 76]

Strings are Immutable but arrays are mutabl

|  |  |
| --- | --- |
| Arrays | Strings |
| 1. Arrays are mutable, we can change the values of array   let price = [97, 95, 82, 91, 90, 76];  price[3]=52;  console.log(price);  output  [97, 95, 82, 52, 90, 76] | 1.**strings are immutable**,it can’t change the original string, it will create new value of string.  let str2 = "Happy Testing";  str2.toUpperCase();  console.log(str2); //op Happy Testing  Output : Happy Testing |

**Looping Over an Array used to** Print all the elements of an array

**Array Length = array index + 1**

**Example** [97, 95, 82, 52, 90, 76] = 5+1=6 arrany length is 6

//looping over array to print array

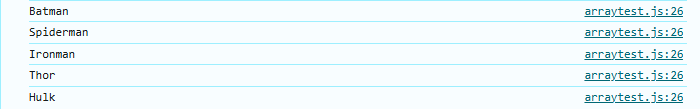
let heros = ["Batman", "Spiderman", "Ironman", "Thor", "Hulk"];

for(let idx=0; idx<heros.length; idx++)

{

    console.log(heros[idx]);

}



//print the array using for of loop

let students = ["POOJa", "Snehal", "Monika", "Aditya", "Nikhil", "Manish"];

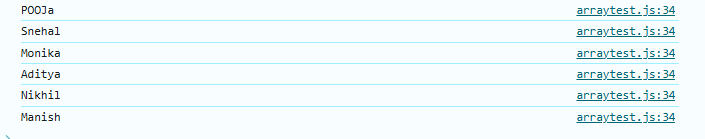
for(let x of students)

{

    console.log(x);

}

**Output**

****

**Practice Test 5**

Q. For a given array with marks of students -.[85, 97, 44, 37, 76, 60] Find the average marks of the entire class.

let marks = [85, 97, 44, 37, 76, 60];

console.log(marks);

let sum = 0;

for(let val of marks)

    {

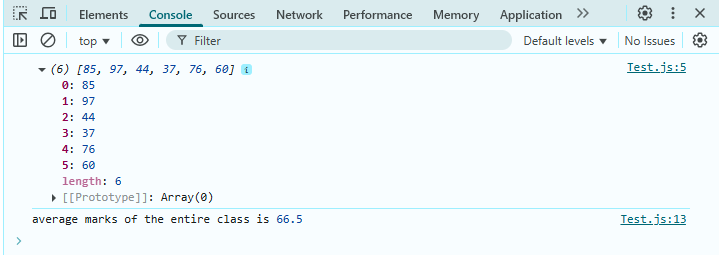
    sum = sum + val;

   }

let avg = sum/marks.length;

 console.log("average marks of the entire class is", avg);

Output :



Q2 For a given array with pricesof5items-> [250, 645, 300, 900, 50]. All items have an offer of 10% OFF on them.Change the array to store final price after applying offer.

let price= [250, 645, 300, 900, 50];

for(let i =0; i<price.length; i++)

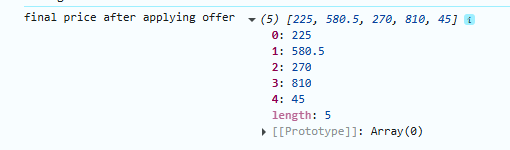
{

   let offer =price[i]/10;

   price[i]= price[i]-offer;

}

  console.log("final price after applying offer", price);



**Array methods in JS**

1. **push( ):** add to end
2. **pop( ) :** delete from end & return
3. **toString( ) :** converts array to string
4. **concat ():** join multiple arrays and return the result.
5. **unshift():** add to start
6. **shift()** : delete from start and return
7. **slice()** :returns a piece of the array
8. **splice()**: change the original array( add , remove, replace)

Example Push() andPop()

//Array methods in JS push() , pop(), toString

//push() method will add items at end of array

let fruits = ["Mango", "orange", "Banana"];

fruits.push("Apple", "Jackfruit");//push method add item at end of array

console.log(fruits); // OP all 5 fruits

//pop() method will delete item from end of array

let flowers = ["Lotus", "Rose", "Mogra", "Lily"];

let new\_flowers= flowers.pop();

console.log(flowers);

console.log("delete items", new\_flowers);



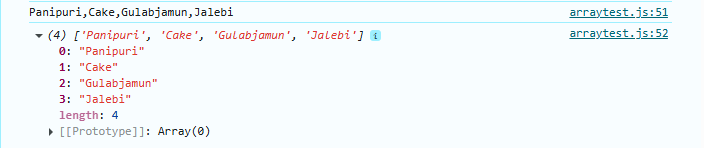
1. **toString()** : it does not change the original array, it will return the new String

//toString converts array to string

let foodItem = ["Panipuri", "Cake", "Gulabjamun", "Jalebi"];

console.log(foodItem.toString());//it will return new string, doesn't change existing

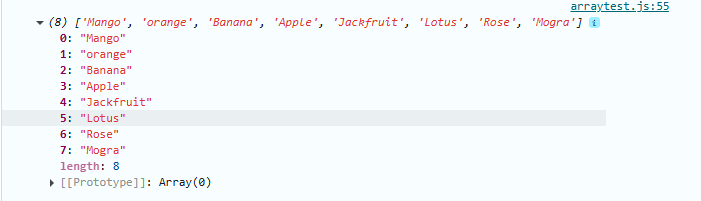
console.log(foodItem);//it will show same fooditem array



1. **concat ():** join multiple arrays and return the result.

//concat() method joins multiple arrays

console.log(fruits.concat(flowers)); //join fruits and flowers array



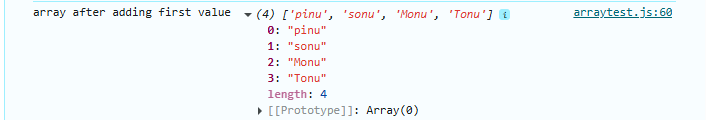
1. **unshift():** add to start

//unshift() method - add items at start of array

let names = ["sonu", "Monu", "Tonu"];

let updated\_Names = names.unshift("pinu");

console.log("array after adding first value",names);



1. **shift() :** delete from start and return

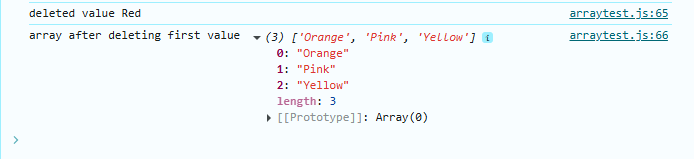
//shift() : delete from start and return

let colors = ["Red", "Orange", "Pink", "Yellow"];

let up\_colors = colors.shift();

console.log("deleted value", up\_colors);

console.log("array after deleting first value",colors);



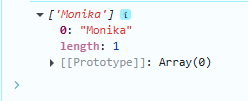
1. **slice()** :returns a piece of the array

//slice() :returns a piece of the array

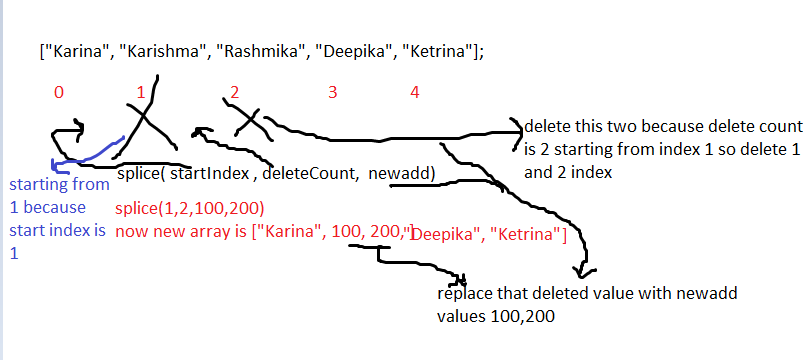
let student = ["Sayli", "Sanchita", "Monika", "Manisha"];

let upStudents = student.slice(2,3);//it will return only first mentioned index

console.log(upStudents); // op "Monika"



1. **splice()**: change the original array( add , remove, replace)



startIndex = starting index

deleteCount = delete value stating from start index

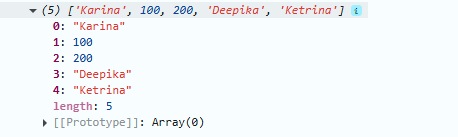
newadd= add new values where we have deleted deleted count

//splice(): change the original array( add , remove, replace)

let heroines = ["Karina", "Karishma", "Rashmika", "Deepika", "Ketrina"];

let values = heroines.splice(1,2,100,200);

console.log(heroines); // op  ['Karina', 100, 200, 'Deepika', 'Ketrina']



**Practice test on array Methods**

Q3.Create an array to store campanies -> “Bloomberg”, “Microsoft”, “Uber”, “Google”, “IBM”, “Netflix”

1. Remove the first company from the array
2. Remove Uber and add ola in its place
3. Add amazon at the end

let campany = ["Bloomberg","Microsoft", "Uber", "Google", "IBM", "Netflix"];

// a. Remove the first company from the array

let upCampany= campany.shift(0);

console.log(campany); // output of a is ['Microsoft', 'Uber', 'Google', 'IBM', 'Netflix']

// b. Remove Uber and add ola in its place

//updated array is ['Microsoft', 'Uber', 'Google', 'IBM', 'Netflix'] after a

let nvalue = campany.splice(1,1,"ola");

console.log(campany);

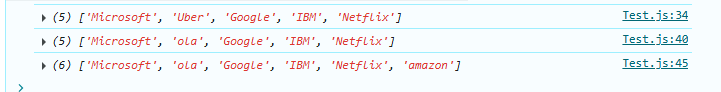
//c.  Add amazon at the end

//updated array after b ['Microsoft', 'ola', 'Google', 'IBM', 'Netflix']

let latest\_array = campany.push("amazon");

console.log(campany);

//op  ['Microsoft', 'ola', 'Google', 'IBM', 'Netflix', 'amazon']



**Functions and Methods**

**Functions :** block of code to performs a specific task,can be invoked or call whenever needed.

**Function Definition**

**Type 1 Type 2**

function functionName( ) function functionName(param1, param2…. )

{ {

//do some work //do some work

} }

=========================

**Function Call**

**functionName();**

==============================

function myTest()

{

   console.log("Welcome to Mphasis!!!");

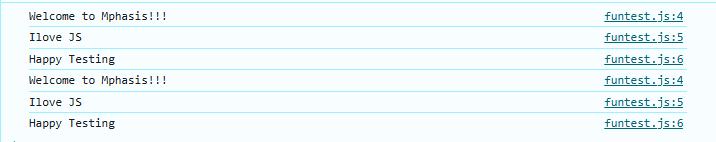
   console.log("Ilove JS");

   console.log("Happy Testing");

}

myTest();//we can call functions multiple time

myTest(); //op all console will print twice



* **Functions by passing parameters**

//Functions using parameters and arguments.

function myTesting(msg,msg1)

{

    console.log(msg,msg1)

}

myTesting("I Love JS",100); //op I Love JS 100

**Output**

****

* **Functions using “return ” keyword**

//Function to calculate sum of 2 numbers

function sum(x,y)

{

    let s = x+y;

    return s;

}

let val = sum(4,5);

console.log("addition of two number is ",val);

**Output : addition of two number is 9**

**Function parameters works as local variables of functions scope of that variable are within function block {} only .**

**Arrow Functions :** Compact way of writing a function

Syntax :

const functionName= (param1, param2…) =>

{

//do some work }

//Arrow Functions

//addition of arrow function

const arrowSum = (a,b) =>

{

    console.log(a+b);

};

arrowSum(6,5);//op 11

//multiplication of arrow function

const arrowMulti = (i,j) =>

{

    console.log(i\*j);

}

arrowMulti(10,20);//op 200

**Practice Test 6**

Q1. Create a function using the “function” keyword that takes a String as an argument & returns the number of vowels in the String(example String “Hello”)

function countVowels(str)

{

    let count = 0;

    for(let char of str)

    {

       if( char === "a" ||

           char === "e" ||

           char === "i" ||

           char === "o" ||

           char === "u")

        {

           count++;

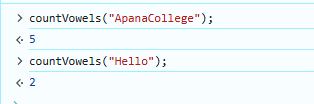
        }

    }

     return count;

}

**Output** :enter any string like countVowels("ApanaCollege");it will give you a output



**-Create an arrow function to perform the same above task.**

//Create an arrow function to perform the same above task.

const arrowCountVow = (str)=>

{

let count = 0;

    for(let char of str)

    {

       if( char === "a" ||

           char === "e" ||

           char === "i" ||

           char === "o" ||

           char === "u")

        {

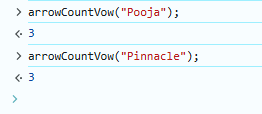
           count++;

        }

    }

     return count;

} //at console call like this arrowCountVow("Pooja");

****

1. **For Each Method in Arrays**

**forEach is method to perform operation on each index of array)**

Arr.forEach(callBackFunction)

CallbackFunction: **Here, it isa function to execute for each element in the array**

**A callback is a function passed as an argument to another function**

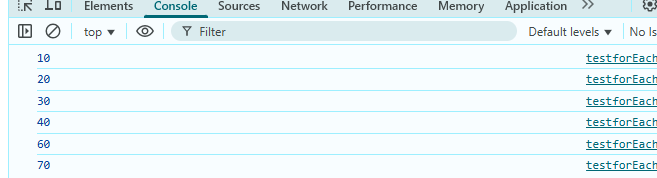
let arr = [10,20,30,40,60,70];

arr.forEach(function printVal(val)//value at each index

{

  console.log(val);

});//print whole array

**Output **

**ForEach loop to print value, index and whole array**

//forEach loop toprint value, index and whole array

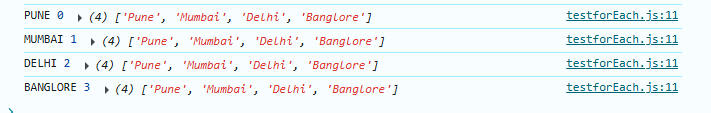
let city = ["Pune", "Mumbai", "Delhi", "Banglore"];

city.forEach((vals,indx,city)=>

{

  console.log(vals.toUpperCase(),indx,city);

});

****

**Higher Order Function/Methods**

**ForEach is one of the Higher Order Function/Methods**

**HOF or HOM** It will take other function as parameter or it will return another function as their output

**Practice Test on For Each Loop**

Q1 For a given array of numbers, print the square of each using the forEach loop.

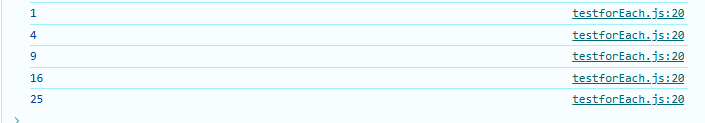
let num = [1,2,3,4,5];

num.forEach(function squareNum(sq)

{

  console.log(sq\*sq);

});



1. **Map Method in array**

Create a new array with the results of some operation.The value its callback returns are used to form new array

**arr.map(callbackFnx(value,index,array))**

|  |  |
| --- | --- |
| **forEach Method in array** | **Map Method in array** |
| forEach method is used to perform some operation on each index of array and return the result with each index **it does not return new array** | forEach method is used to perform some operation on each index of array and return the result with each index **it returns new array** |
|  |  |

//map method in array

let nums =[5,10,15,25];

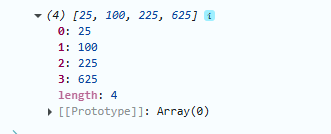
let newArr = nums.map((val) =>

{

       return val\*val;

});

console.log(newArr);

****

1. **Filter Method in array**

Create a new array of elements that give true for a condition/filter

Eg: All even elements

//filter out even values from array

let arr = [1,2,3,4,5,6,7,8,9,10];

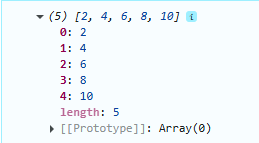
let evenArray=arr.filter((val)  =>

{

   return val % 2 === 0;

});

console.log(evenArray);



1. **Reduce Method in array**

Performs some operations and reduces the array to a single value.It returns that single value.

let arr = [1,2,3,4,5,6];

const output = arr.reduce((res,cur) =>

    {

        return res+cur;

    });

console.log(output); //op 21

//1+2 = 3 , 3+3=6, 6+4=10,10+5=15, 15+6=21

//output of result + current is 21

//Q.1 Find the largest number using reduce method

let num =[200, 600, 1000, 300,400];

let largeNum = num.reduce((pre,curr) =>

{

   return pre>curr?pre:curr;

});

console.log(largeNum); //op largest number is 1000

//Explanation 200>600 = false, 600>1000=false, 1000>300=True so op 1000

//1000>300=true , 1000>400=true so op is 1000

**Practice Test 6 on Array Methods(forEach, map,filter,reduce)**

Q1. We are given array of marks of students.Filter out of the marks of students that scored 90+ .given array [87,93,64,99]

 let arr = [87,93,64,99,86];

 let output = arr.filter((val,indx,arr) =>

 {

    return val>90;

 });

 console.log(output);//op 93,99

Output



Q2 Take a number n as input from user.Create an array ofnumbers from 1 to n.

Use the reduce method to calculate sum of all numbers in the array.

Use the reduce method to calculate product of all numbers in the array.

let n = prompt("enter a number: ");

let arrr =[];

for(let i=1; i<=n; i++)

   {

         arrr[i - 1] = i;

   }

   console.log(arrr);

   //Use the reduce method to calculate sum of all numbers in the array.

   let sum = arrr.reduce((res, curr)=>

   {

        return res + curr;

   });

   console.log("sum of all the numbers in the array is",sum);

   //Use the reduce method to calculate product of all numbers in the array.

   let prod = arrr.reduce((prev,currr) =>

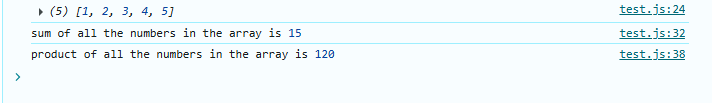
{

   return prev\*currr;

});

console.log("product of all the numbers in the array is",prod);

Output

****

* **Document Object Model(DOM)**

**HTML** (Structure) **CSS**(style) **JavaScript**(Logic)

**Structure of HTML**

**<html>**

**<head>**

**<title> Website Name </title>**

**</head>**

**<body> Content tags </body>**

**</html>**

Example :

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>JS File</title>

</head>

<body>

    <h1> Welcome to JavaScript Test</h1>

    <h4>Apna College Test</h4>

    <p>This are the basic questions related to javascript</p>

    <button>click</button>

</body>

</html>

Output



**Note: VIMP : Tags To connect JS and CSS to HTML**

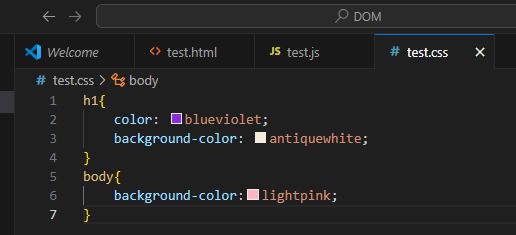
<style>tag connects HTML with CSS

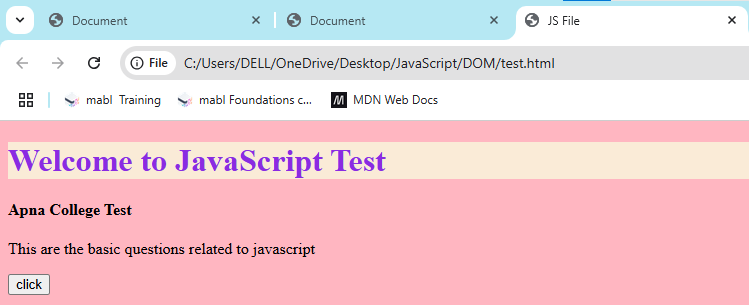
<link rel="stylesheet" href="test.css"/>

Write this inside head tag and before title tag

<script>tag connects HTML with JS

**How to add style inside CSS**

****

****

**Window Object**

The window object represents an open window in a browser.It is browser’s object(Not JavaScript’s )& is automatically created by browser.

**It is a global object with lots of properties & Methods.**

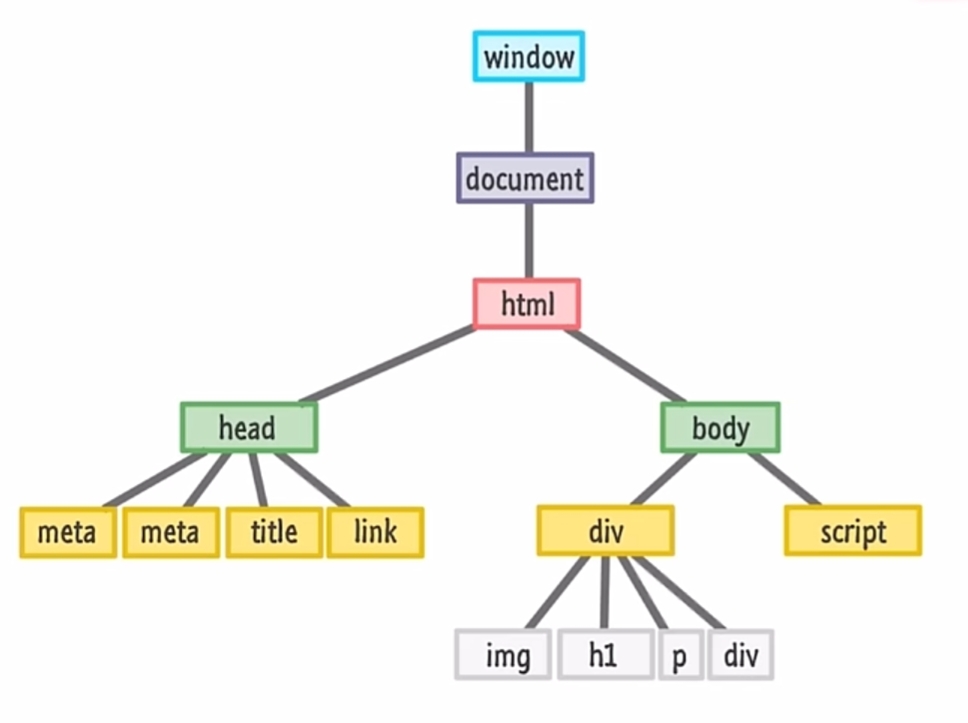
* Open console and type window it will show you output

example



**What is DOM?**

When a web page is loaded, the browser creates a Document Object Model(DOM) of the page.



|  |  |
| --- | --- |
| **console.log** | **console.dir** |
| Print the object  console.log(“hello”); | Print the properties and methods of documents  console.dir(window.document); |

**DOM Manipulation**

**If we write the script.js means jsfile before body tag means inside title tag output will be null.**

1. **Selecting with id (#idName)**

document.getElementById(“myId”)

***if id doesn’t Exist it will print null***

1. **Selecting with class (.className)**

document.getElementByClassName(“myClass”)

it will return html collection

***if class doesn’t Exist it will print empty collection with 0 length***

1. **Selecting with tag**

document.getElementByTagName(“p”)

**Example for id :**

<h1 id="Hello"> DOM Demo by Snehal Kakade</h1> //write inside html file

Access id using jS

let x = document.getElementById("Hello"); //write inside JS file

console.dir(x);

Output :



***if id doesn’t Exist it will print null***

//using id

let x = document.getElementById("Hello00000"); //inside JS file

console.dir(x);

**Output**

**Example for id using #**

Inside css file just write #idName Example #Hello

#Hello{

    color: blueviolet;

    background-color: antiquewhite;

}

body{

    background-color:lightpink;

}

Write same h1 tag inside html

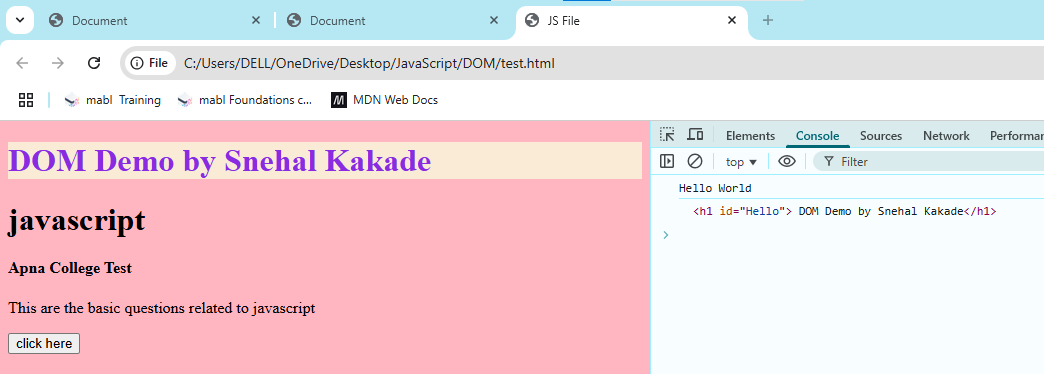
<h1 id="Hello"> DOM Demo by Snehal Kakade</h1>

<h1 id="Hii">javascript</h1>

But styling is applicable only for id Hello because we have used #id only for #Hello id

So styling only show for first h1 tag not for h1 second Hii id

Output is here



**Example for Class (represent using .className inside css file)**

.Hello1

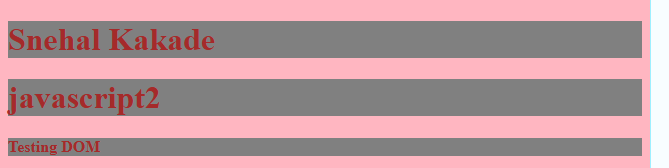
{

    color: brown;

    background-color:gray;

}

Output it will show all 3 heading of class in same style



Access class inside JS using

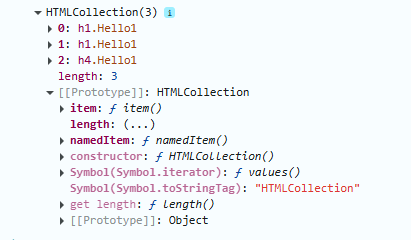
//using class

 let y = document.getElementsByClassName("Hello1"); //inside js file

 console.dir(y);

 console.log(y);

Output



***if class doesn’t Exist it will print empty collection with 0 length***

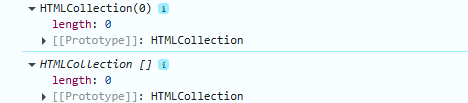
//using class

 let y = document.getElementsByClassName("Hello888"); //inside js file

 console.dir(y);

 console.log(y);

Output



**3)Using Tag name**

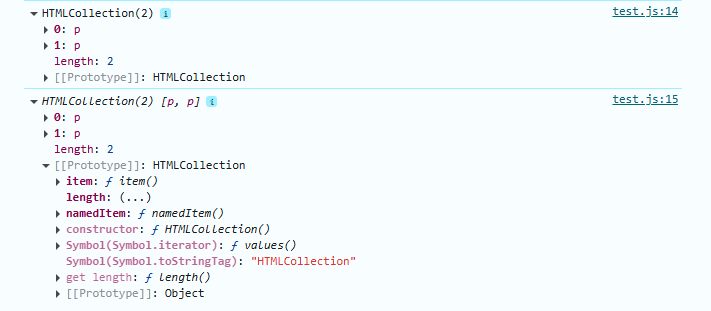
 //using TagName

 let z = document.getElementsByTagName("p");

 console.dir(z);

 console.log(z);

output



**Query Selector**

document.querySelector(“myId/myClass/tag”) //returns first element

document.querySelectorAll(“myId/myClass/tag”) //returns a NodeList all

1. **Example Query Selector using tag**

 //Using QuerySelector using tag

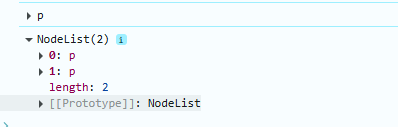
 let a = document.querySelector("p"); //query selector using tag

 console.dir(a); //returns first element

 //Using querySelectorAll using tag

  let b = document.querySelectorAll("p"); //query selector using tag

 console.dir(b); //returns all elements



1. **Example Query Selector using class**

 //Using QuerySelector using className

 let a = document.querySelector(".Hello1"); //query selector using className

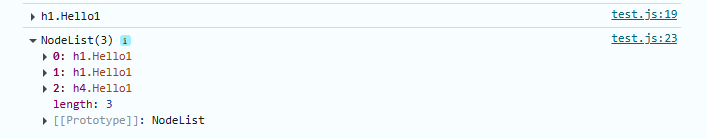
 console.dir(a); //returns first element

 //Using querySelectorAll using className

  let b = document.querySelectorAll(".Hello1"); //query selector using className

 console.dir(b); //returns all elements

**Output**

****

1. **Example Query Selector using id**

**Using querySelectorAll doesn’t make sense because id has unique value**

 //Using QuerySelector using id

 let a = document.querySelector("#Hello"); //query selector using

 console.dir(a); //returns first element

****

**DOM Manipulation Properties**

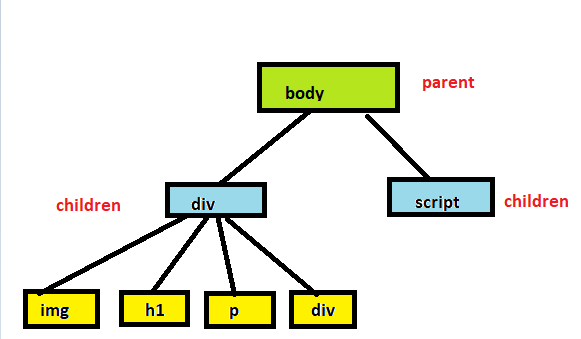
* tagName : returns tag for element nodes
* innerText : returns the text content of the element and all its children
* innerHTML : returnsthe plain text or HTML contents in the element
* textContent: returns textual content even for hidden elements

**Example of tagName**

let a = document.querySelector(".Hello1");

console.dir(a.tagName);//op H1

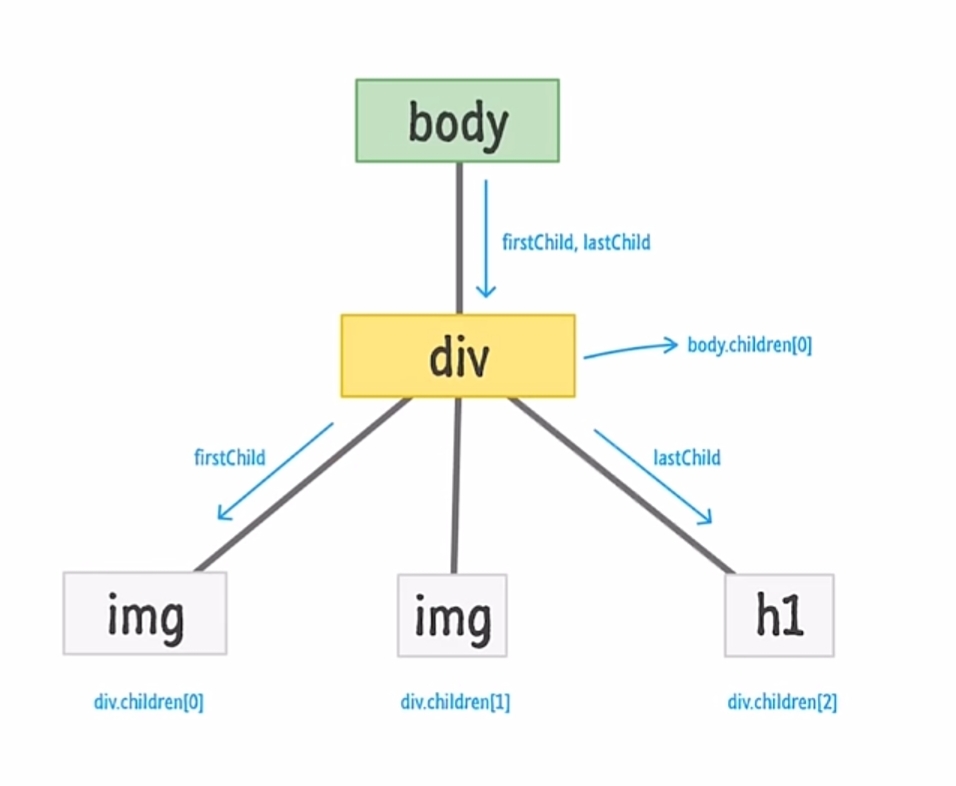
**Example of innerText : DOM Tree**

****

**Parents :** Body

**Children:** div, script

**Siblings :** div-script , img-h1-p-div //homework using MDN doc below tree



**There are 3types of nodes inside DOM tree**

1)text Nodes 2) Comment Nodes 3) Element Nodes

* **Example innerText :**returns the text content of the element and all its children

**Example**

<body>

    <div>

        <h3>Fruits</h3>

        <ul>

            <li>Mango</li>

            <li>Orange</li>

            <li>litchi</li>

        </ul>

    </div>

    <script src="domtest.js"></script>

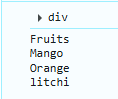
</body>

**JS code**

let div = document.querySelector("div");

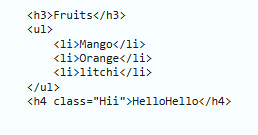
console.dir(div);

console.log(div.innerText);

****

**3)innerHTML : returnsthe plain text or HTML contents in the element**

console.dir(div.innerHTML);//using innerHTML

****

1. **textContent: returns textual content even for hidden elements**

<body>

    <h1 style="visibility: hidden">old visibility</h1>

    <div>

        <h3>Fruits</h3>

        <ul>

            <li>Mango</li>

            <li>Orange</li>

            <li>litchi</li>

        </ul>

        <h4 class="Hii">HelloHello</h4>

    </div>

    <script src="domtest.js"></script>

</body>

//using textContent

let heading = document.querySelector("h1");

console.dir(heading.textContent);

****

**Examples of DOM Manipulation**

//DOM Manipulation Properties

//tagName : returns tag for element nodes

//innerText : returns the text content of the element and all its children

//innerHTML : returnsthe plain text or HTML contents in the element

//textContent: returns textual content even for hidden elements

//using tagName

let a = document.querySelector(".Hii");

console.dir(a.tagName);

//using innerText

let div = document.querySelector("div");

console.dir(div.innerText); //get innerText

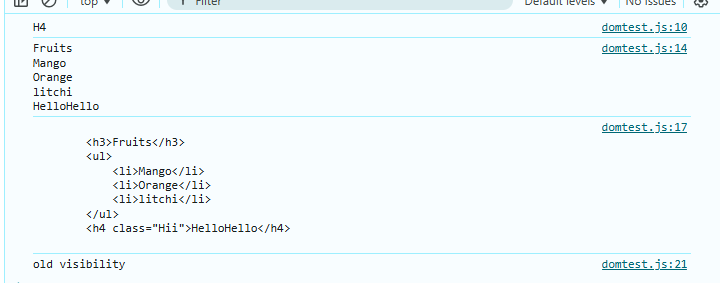
//using innerHTML

console.dir(div.innerHTML); //get innerHTML

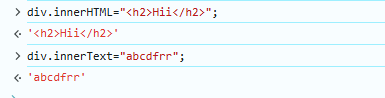
//using textContent

let heading = document.querySelector("h1");

console.dir(heading.textContent);

****

**Code To set innerText and innerHTML**

****

**Practice Test on DOM**

Q1. Create a H2 element with text –“Hello JavaScript”.Append “from apana Collee Students”to this text using JS.

<body>

    <h2>Hello JavaScript</h2>

    <script src="practicetest.js"></script>

</body>

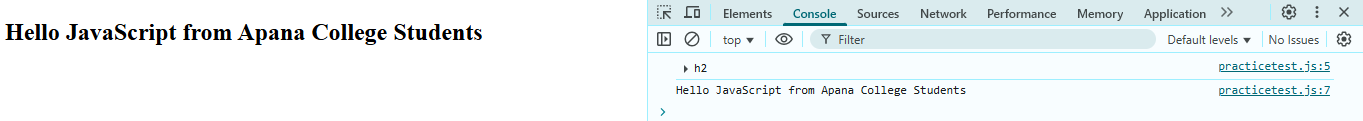
let x = document.querySelector("h2");

console.dir(x);

x.innerText= x.innerText + " from Apana College Students";

console.dir(x.innerText);

Output



Q2.Create 3 divs with common class name – “box”.Access them & add unique text to each of them.

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="stylesheet" href="Test.css"/>

</head>

<body>

    <div class="box">first div</div>

    <div class="box">second div</div>

    <div class="box">third div</div>

    <script src="Test2.js"></script>

</body>

.box{

    height: 100px;

    width: 100px;

    border: 1px solid black;

   background-color: aquamarine;

}

let divs = document.querySelectorAll(".box");

divs[0].innerText="new unique value 1";

divs[1].innerText="new unique value 2";

divs[2].innerText="new unique value 3";

console.dir(divs);

