VIRTUAL CODE

Javascript

In one Shot

Level 0

Introduction to JavaScript

What is JavaScript:

- Javascript is a programming language which helps us to intract with the computer.
- In web development, it is used to make dynamic or logical websites.
- Javascript can be executed in one’s browser on javascript console.
- We can use node.js to run javascript code.
- Another way to execute javascript is by inserting it inside <script> tag of an HTML document.

Level 1

Variables & Data Types

Variables :

- Variables are used to store some data.

Variables :

- Variables are used to store some data.

Declaration of Variables :

- let a = 20;
- var a = 20;
- const a = 20;

- Var is globally scoped while let & const are block scoped.
- var can be updated & re-declared within its scope.
- let can be updated but not re-declared.
- Const can neither be updated nor be declared.

Rules for choosing Variable's names :

- Letters,digit,underscores & $ sign allowed.
- Must begin with a $, _ or a letter.
- Javascript reserved words can not be used as a variable name.
- It is case sensitive . AYUSH is not equal to ayush.

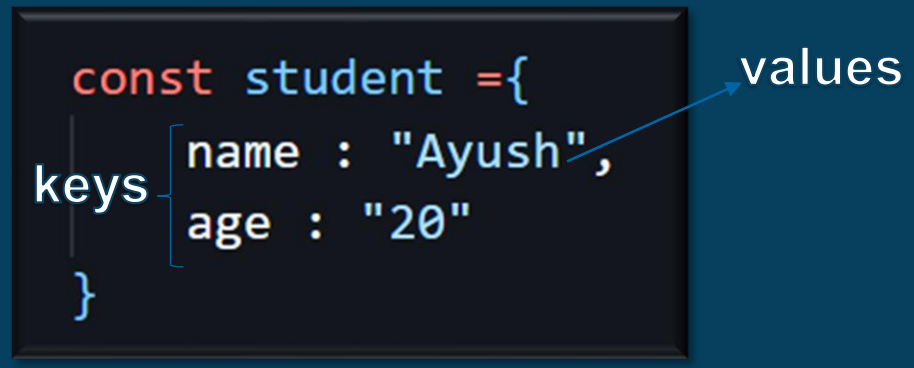
Primitive data types & objects :

- There are 7 primitive data types:

Primitive data types
Null
Number
String
Symbol
Undefined
Boolean
BigInt

objects :

- An object in javascript can be created as follows:



Level-1 practice sheet

- Create a variable of type string and try to add a number to it.
- Use typeof operator to find the datatype of the string in last question.
- Create a const object in javascript can you change it to hold a number later.
- Try to add a new key to the const object in problem 3 were you able to do it.
- Write a javascript program to store name , age , marks of a student using objects.

Level 2

Operators & Conditional statements

Operators In javascript

Arithmetic operators	
+	Addition
-	Subtraction
*	multiplication
**	exponentiation
/	Division
%	Modulus
++	Increment
--	Decrement

Operators In javascript

Assignment operators	
=	a=b
+=	a=a+b
- =	a=a-b
* =	a=a*b
/=	a=a/b
%=	a=a%b
**=	a=a**b

Operators In javascript

Comparison operators	
==	Equal to
!=	Not equal to
===	Equal value and type
!==	Not equal value or not equal type
?	Ternary operator
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Operators In javascript

logical operators	
&&	Logical and
||	Logical or
!	Logical not

Conditional statements:

- if statement

```
if(condition){
    //statement
}
```

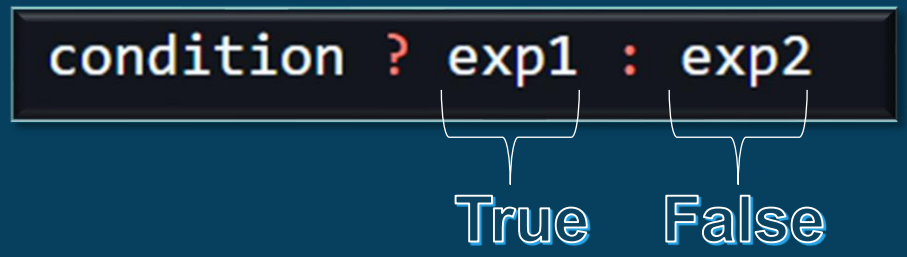
- If-else statement

```
if(condition){
    //statement if condition is true
}
else{
    //statement if condition is false
}
```

- If-else if statement

```
if(condition){
    //execute if condition is true
}
else if(condition){
 //execute if above condition is false and this condition is true
}
else{
    //execute if both conditions are false
}
```

ternary operator



Level-2 practice sheet

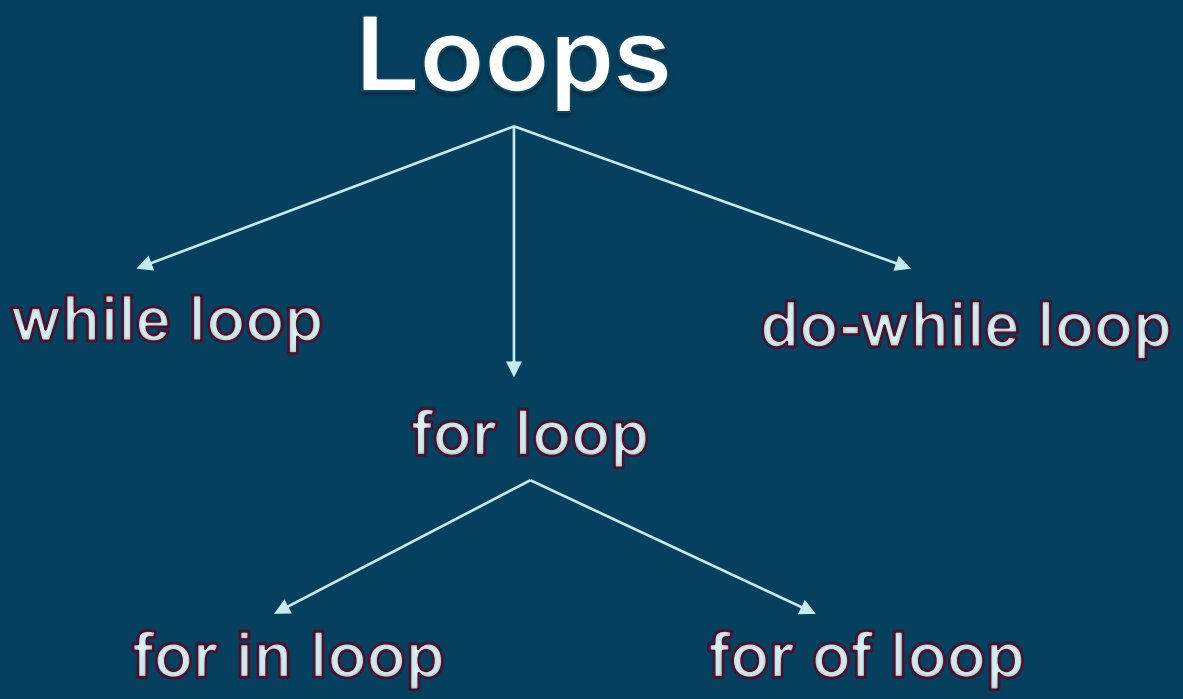
- Use logical operators to find whether the age of a person lies between 10 and 20
- Demonstrate the use of switch case statements in javascript.
- Write a javascript program to find whether a number is divisible by 2 and 3
- Print “rejected” or “accepted” according to proposal answer

Level 3

Loops and Functions

Loops:

- Use to perform repeated actions.



for Loop:

initialization condition updation

- for loop

```
for(statement 1; statement 2; statement3){
    //code to be executed
}
```

- for in loop

```
for(key in object){
   //code to be executed
}
```

- for of loop

```
for(variable of iterable){
    //code to be executed
}
```

while Loop:

```
while(condition){
    //code to be executed
}
```

do-while Loop:

```
do{
    //code to be executed
}
while(condition)
```

Functions in javascript :

- Function is a block of code designed to perform particular task.

Declaration of function:

```
function name(parameter1,parameter2){
    //code to be executed
}
```

```
const add =(a,b)=>{
    let c=a+b;
    return c;
}
```

Function calling:

```
let x=add(5,6);
```

Level-3 practice sheet

- Write a program to to print the marks of a student in an object using for in loop

  obj={ ayush:95 , Hemant:92 , Dev:33 }

- Write the program to print table of given number.
- Write a program to print “try again” until the user enters the correct number.
- Write a function to print the average of 4 numbers.

Level 4

Strings

Strings :

- String is the collection of the characters

```
let name="ayush"
```

- Template literals

```
let statement=`${name} is a good boy`
```

name is variable

- Escape sequence characters

```
let name='nobisuke is nobita\'s son'
```

\n	New line
\t	tab

Strings method

Properties & methods
name.length
name.toLowerCase( )
name.toUpperCase( )
name.slice(a,b) or name.slice(a)
name.replace(“abc”,”def”)
name1.concat(name2)
name.trim( )

Level-4 practice sheet

- Find the output:

```
console.log("ayush\"".length)
```

- Explore includes,startsWith & endsWith function of a string
- Write a program to convert a given string to upper case
- Extract score's number out of this string "virat scored 100 in last match "
- Try to change 3rd character of a given string ,were you able to do it ?

Level 5

Arrays

Arrays :

- Arrays are variables which can hold more than one value.

```
const arr=[27,"ayush",true,"dev"]
```

- For accessing the values in array:

```
let numbers=[1,2,3,4,5,6]
numbers[0] // print 1
numbers[1] // print 2
```

Methods of Arrays :

Methods of Arrays	
toString( )	Convert an array to a string of comma separated values
join( )	Joins all the arrays elements using a separator
pop( )	Removes last element from the array
push( )	Adds a new element at the end of the array
shift( )	Removes first element and returns it
unshift( )	Adds element to the beginning returns new array length
delete	Array elements can be deleted using delete operator
Concat( )	Used to join arrays to the given array
Sort( )	It is used to sort an array alphabetically
splice( )	Splice can be used to add new items to an array
slice( )	Slices out a piece from an array , it creates a new array
reverse( )	Reverse the elements in source array

looping through Arrays :

- forEach loop:

```
const arr=[1,3,5,7,9]
arr.forEach((value,index,array)=>{
    //function logic
});
```

- map( ) :

```
const arr1=[1,3,5,7,9]
let new_arr1=arr1.map((value,index,array)=>{
    return value*value;
});
```

- filter( ) :

```
const arr2=[1,3,5,7,9]
let new_arr2=arr2.filter((x)=>{
  return x>2
});
```

looping through Arrays :

- Reduce method:

```
const arr3=[1,3,5,7,9]
let new_arr3=arr3.reduce((a,b)=>{
  return a+b;
});

Array.from("ayush")
```

- Array.from :

```
Array.from("ayush")
```

Level-5 practice sheet

- Create an array of numbers and take input from the user to add numbers to the beginning of this array.
- Keep adding numbers to the array in 1 until 27 is added to the array.
- Filter for even numbers from a given array
- Create an array of cube of given number.
- Use reduce to calculate factorial of a given number from an array of first n natural numbers

Level 6

DOM :Document Object Model

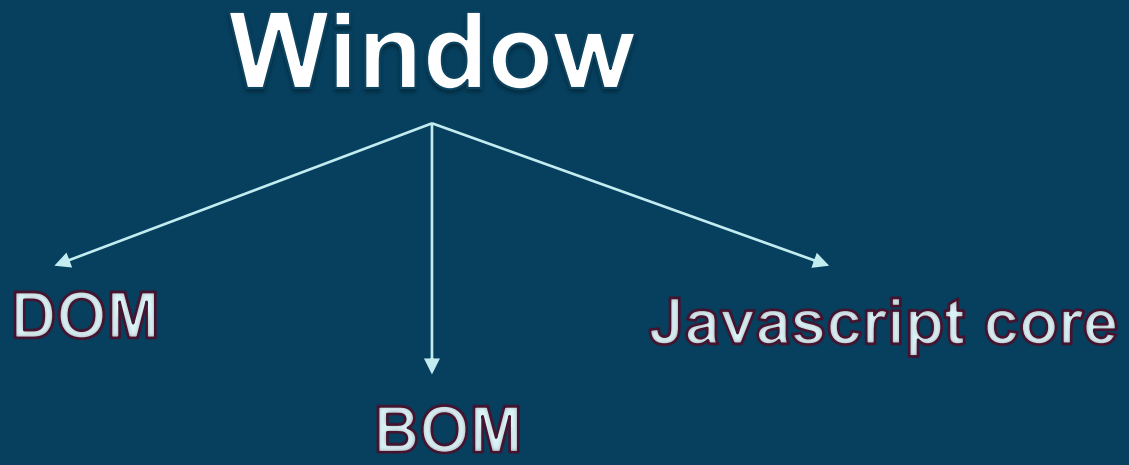
Console object methods

Console object method	
assert( )	Used to assert the condition
clear( )	Clears the console
log( )	Outputs a message to a console
table( )	Displays a tabular data
warn( )	Used for warnings
error( )	Used for errors
info( )	Used for special information

alert,prompt and confirm

alert,prompt and confirm	
alert	Used to invoke a mini window with a msg
prompt	Used to take input as string
confirm	Shows a message and waits for the user to press ok or cancel,return true for ok and false for cancel

Window object, BOM & DOM

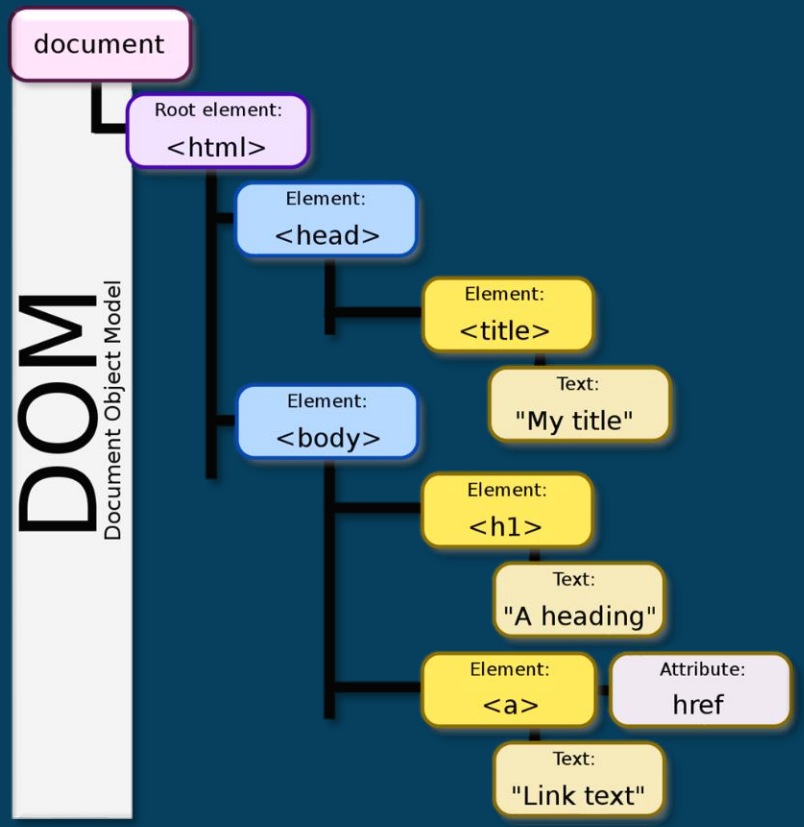


BOM: Browser Object Model

- The browser object model(BOM) represents additional objects provided by the browser for working with everything except document
- The functions alert/confirm/prompt are also a part of BOM

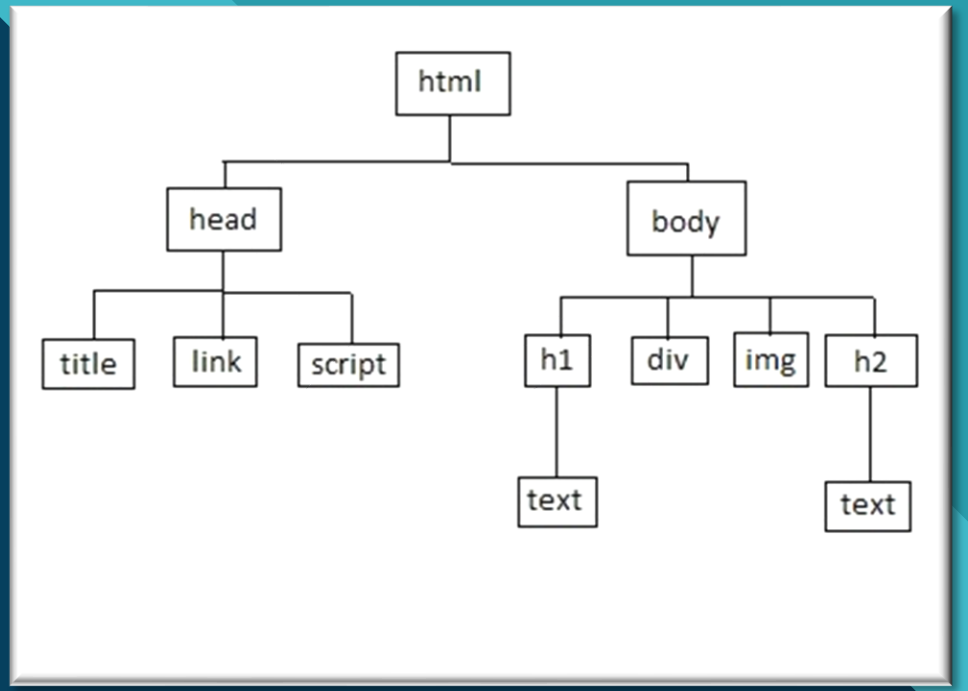
DOM: Document Object Model

- DOM represents the page content as HTML
- DOM tree refers to the HTML page where all the nodes are object , there can be three main types of node in the DOM Tree:
- Text node
- Element node
- Comment nodes



Walking the DOM

Access the elements	
element.firstChild	First child element
element.lastChild	last child element
element.childNodes	All child nodes
element.parentNode	Parent nodes
element.parentElement	Parent elements
document.previousElementSibling	Previous element sibling
document.nextElementSibling	next element sibling
document.firstElementChild	First element child
document.lastElementChild	Last element child



Searching the DOM

Searching the DOM	
document.getElementById( )	Get a element by id
document.querySelectorAll( )	Return all the elements inside an element matching the given CSS selectors
document.querySelector( )	Return a element inside an element matching the given CSS selectors
document.getElementsByTagName( )	Return elements with given tag name
document.getElementsByClassName( )	Returns elements that have the given CSS class
document.getElementsByName( )	Searches elements by the name attribute

Matches,closest & contains methods

- A.matches(CSS) : to check if A matches the given CSS selector

- B.closest(CSS) : to look for the nearest ancestor that matches the given CSS selector , B itself is also checked
- A.contains(B) : return true if B is inside A or when A==B

Level-6 practice sheet

- Write a program using prompt function to take input of age as a value from the user and use alert to tell him if he can derive , use confirm to ask the user if he wants to see prompt again
- Change the background of the page to any other color based on user's input
- Create a nav bar and change the color of its first element to brown.
- Write a javascript code to change background of all <li> tags to purple

Level 7

Events & other DOM properties

Attribute Methods

Access the elements	
a.has Attribute(name)	Method to check for existence of an attribute
a.getAttribute(name)	Method used to get the value of an attribute
a.setAttribute(name,value)	Method used to set the value of an attribute
a.removeAttribute(name)	Method to remove the attribute from a
a.attributes	Method to get the collection of all atributes

data-x Attribute

innerHTML and outerHTML

- The innerHTML property allows to get the HTML inside the element as a string
- The outerHTML property contains the full HTML innerHTML + the element itself

Insertion Methods

```
let box=document.createElement('div') //create
box.className="ayush" //set class
box.innerHTML="<h1>Virtual code</h1>"
document.body.append(box)
```

Some insertion Method	
name.append(box)	Append at the end of node
name.prepend(box)	Insert at the beginning of node
name.before(box)	Insert before node
name.after(box)	Insert after node
name.replaceWith(box)	Replaces name with the the box

Insert Adjacent Html/text/element	
“beforebegin”	Insert HTML immediately before element
“afterbegin”	Insert HTML into element at the beginning
“beforeend”	Insert HTML into element at the end
“afterend”	insert HTML immediately after element

```
name.insertadjacentHTML("beforebegin","<h1>hello</h1>")
name.insertadjacentHTML("afterbegin","<h1>hello</h1>")
name.insertadjacentHTML("beforeend","<h1>hello</h1>")
name.insertadjacentHTML("afterend","<h1>hello</h1>")
```

classNameand classList	
Name.classList.add(“class”)	Add a class.
Name.classList.remove(“class”)	Remove a class
Name.classList.toggle(“class)	Add the class if does not exist,otherwise removes it
Name.classList.contains(“class”)	Checks for the given class,returns true/false

SetTimeout and SetInterval

- SetTimeout allows us to run a function once after the interval of time
- `Let timerID=setTimeout(function,<delay>,<arg1>,<arg2>)`
- `clearTimeout(timerID)`

- `setInterval runs the function not only once,but regularly after the given interval of time`
- `Let timerID=setTimeInterval(function,<delay>,<arg1>,<arg2>)`
- `clearTimeInterval(timerID)`

Browser Events

- An event is a signal that something has happened all the DOM nodes generates such signal

Some important DOM Events	
Mouse Events	Click.contextmenu(right click), mouseover/mouseout,mousedown/mouseup,mousemove
Keyboard events	Keydown and keyup
Form elements event	Submit,focus etc
Document event	DomContentLoaded

Handling Events

```
//events can be handled through HTML attribute
<button onclick="alert('hello')"></button>
```

```
//events can also be handled through the onclock property
button.onclick=function(){
    alert("hello")
}
```

addEventListener and removeEventListener :

```
button.addEventListener(event_name,handler)
button.removeEventListener(event_name,handler)
```

Level-7 practice sheet

- Write a program to change different background color on body by clicking different buttons
- create a nightmode feature in your website

Project-1 Digital Clock

- Create a digital clock using setInterval and date object in javascript with good UI using HTML & CSS

Level 8

Callbacks, promises,async & await

Synchronous & Asynchronous actions

- Synchronous: means the code runs in a particular sequence of instructions given in the program.
- Each instruction waits for the previous instruction to complete its execution

- Aynchronous: it allows execute next instructions immediately and does not block the flow

Callbacks

- A callback is a function passed as an argument to another function.

Callback Hell

- Nested callbacks stacked below one another forming a pyramid structure.
- Pyramid Of Doom
- This style of programming becomes difficult to understand & manage

Promises

- Promises is for “eventual” completion of task.it is an object in js .
- It is a solution to callback hell.

```
let promise= new Promise((resolve,reject)=>{
  //...
})
```

- .then( ) & .catch( )

Async - Await

- Async function always returns a promise .
- async function myfunc( ){…..}
- Await pauses the execution of its surrounding async function until the promise is settled

Fetch API (Application Programming Interface)

- The fetch API provides an interface for fetching (sending/receiving) resources .
- It uses request and response objects
- The fetch() method is used to fetch a resource or data.

- let promise = fetch(url , [option])

Fetch API (Application Programming Interface)

- The fetch API provides an interface for fetching (sending/receiving) resources .
- It uses request and response objects
- The fetch() method is used to fetch a resource or data.

- let promise = fetch(url , [option])
- AJAX is asynchronous JS & XML
- JSON is Javascript Object Notation
- json( ) method : returns a second promise that resolves with the result of parsing the response body text as JSON.(input is JSON, output is JS object)

Project-2 Weather app

- Create a Weather app using api in javascript with good UI using HTML & CSS

VIRTUAL CODE

Thank You