What is JavaScript?

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

$$\begin{tikzcd}\text{Input (code)} \arrow[r] & \boxed{\text{Computer}} \arrow[r] & \text{Output}\end{tikzcd}$$

Setting up VS Code </>

It is a free & popular code editor by Microsoft

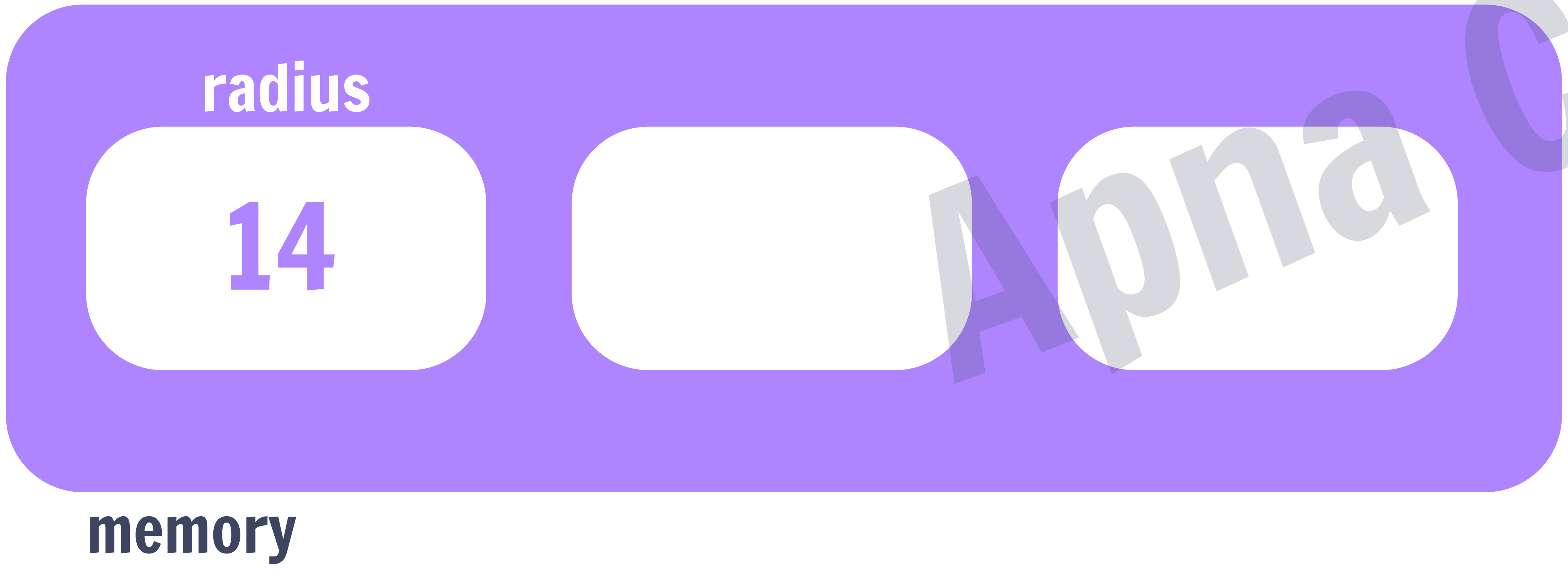
Our 1st JS Code

Console.log is used to log (print) a message to the console

```
console.log("Apna College");
```

Variables in JS

Variables are containers for data



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.

let, const & var

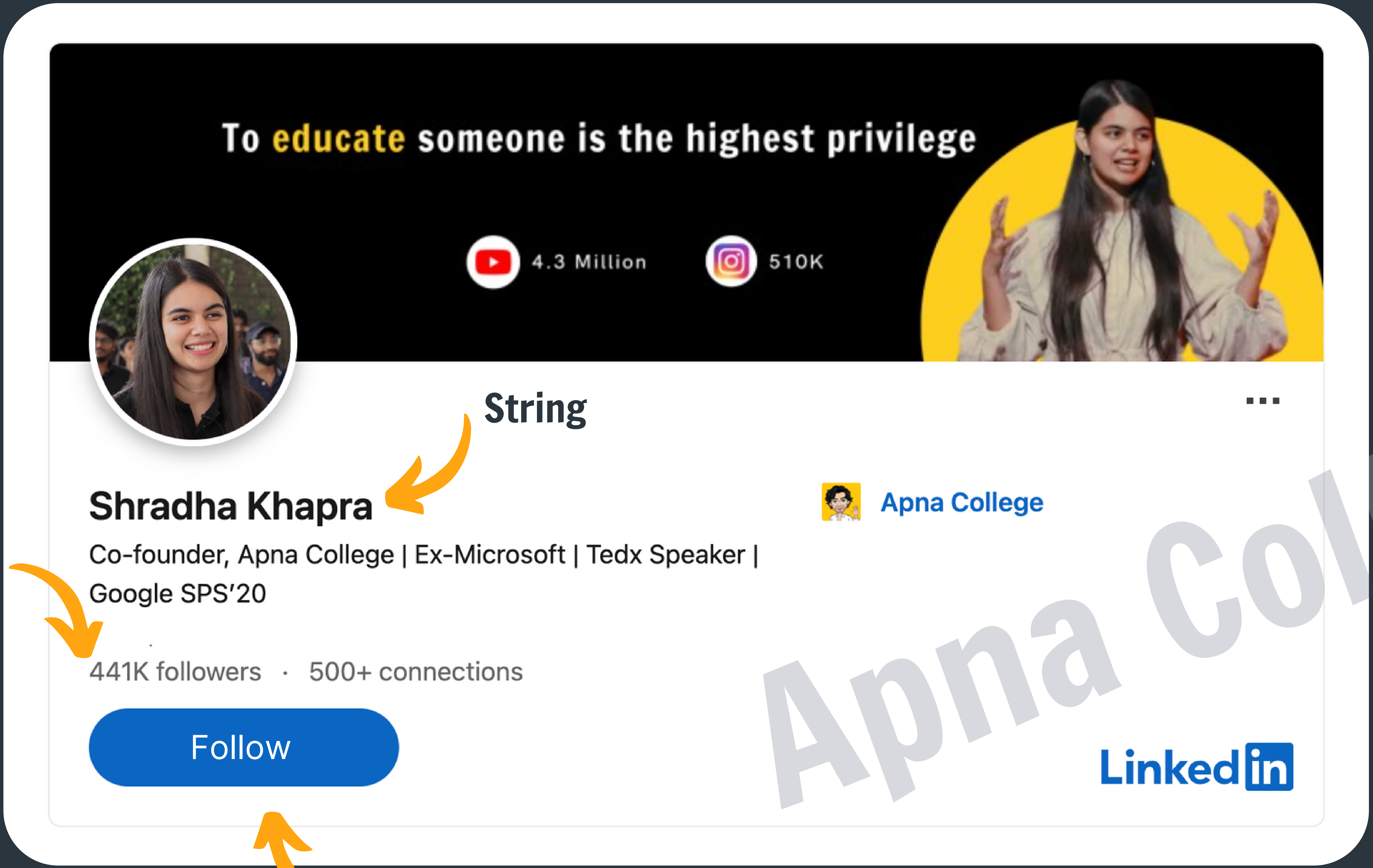
var : Variable can be re-declared & updated. A global scope variable.

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

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

Data Types in JS

Primitive Types : Number, String, Boolean, Undefined, Null, BigInt, Symbol



Shradha Khapra

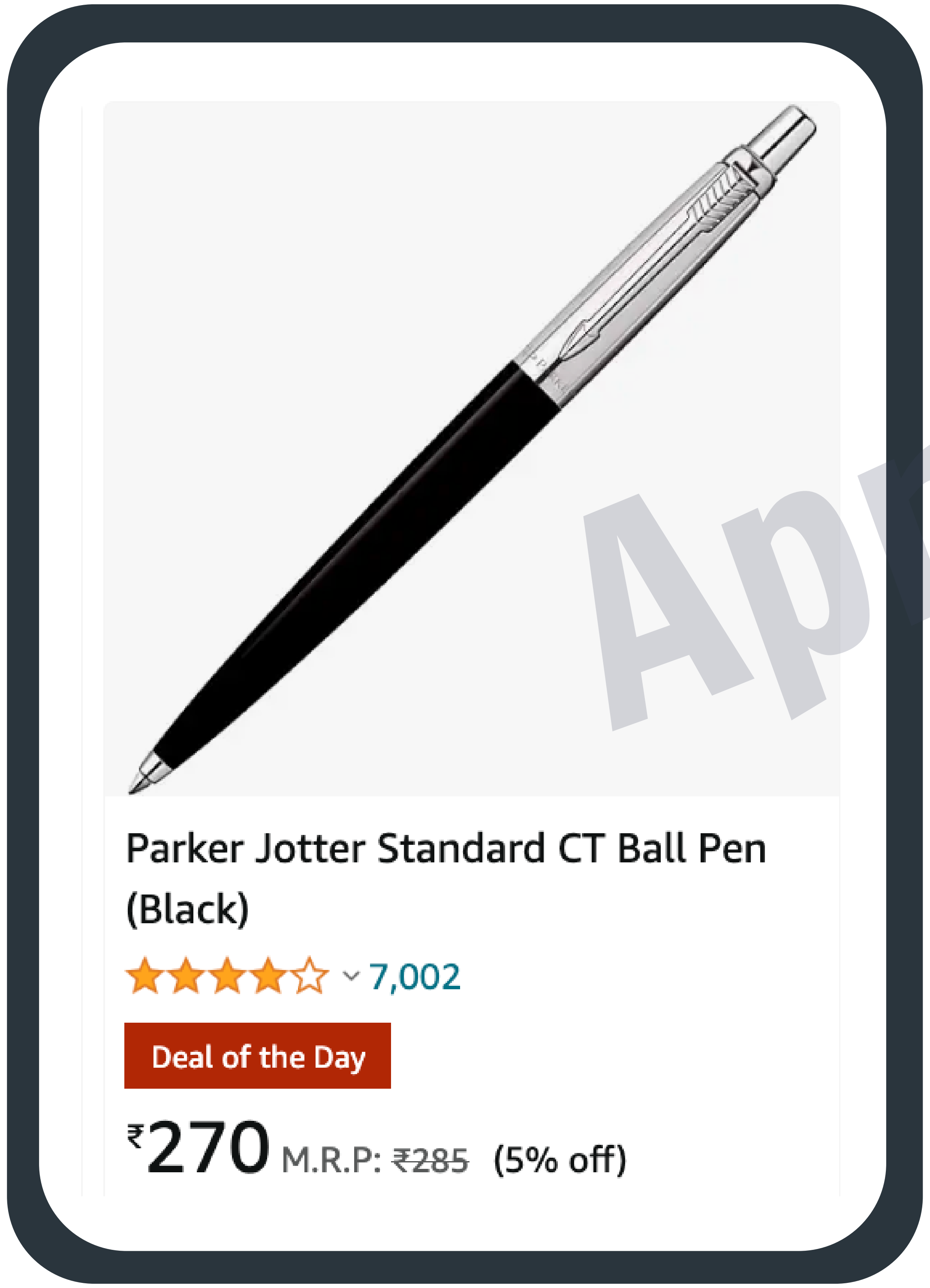
Co-founder, Apna College | Ex-Microsoft | Tedx Speaker | Google SPS'20

441K followers · 500+ connections

Follow

Let's Practice

Qs1. Create a const object called "product" to store information shown in the picture.



Parker Jotter Standard CT Ball Pen (Black)

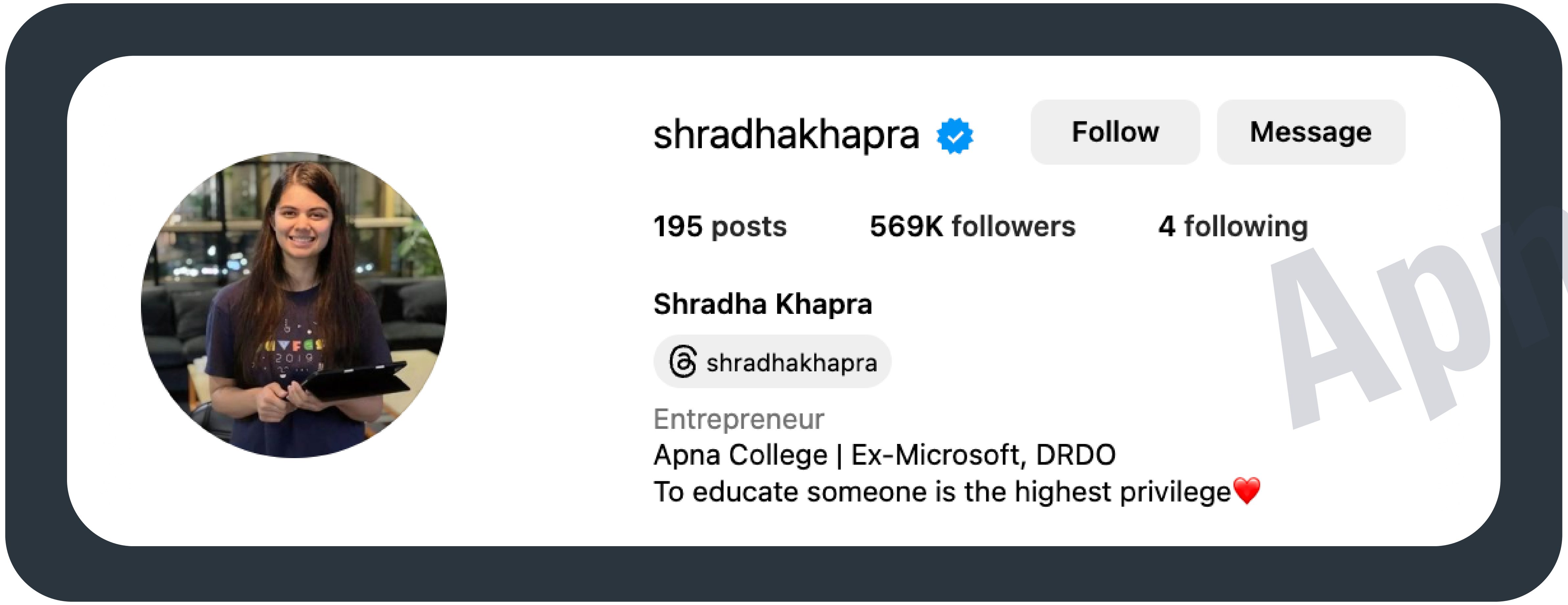
★★★★☆ 7,002

Deal of the Day

₹270 M.R.P: ~~₹285~~ (5% off)

Let‘s Practice

Qs2. Create a const object called “profile” to store information shown in the picture.



shradhakhapra

195 posts 569K followers 4 following

Shradha Khapra

shradhakhapra

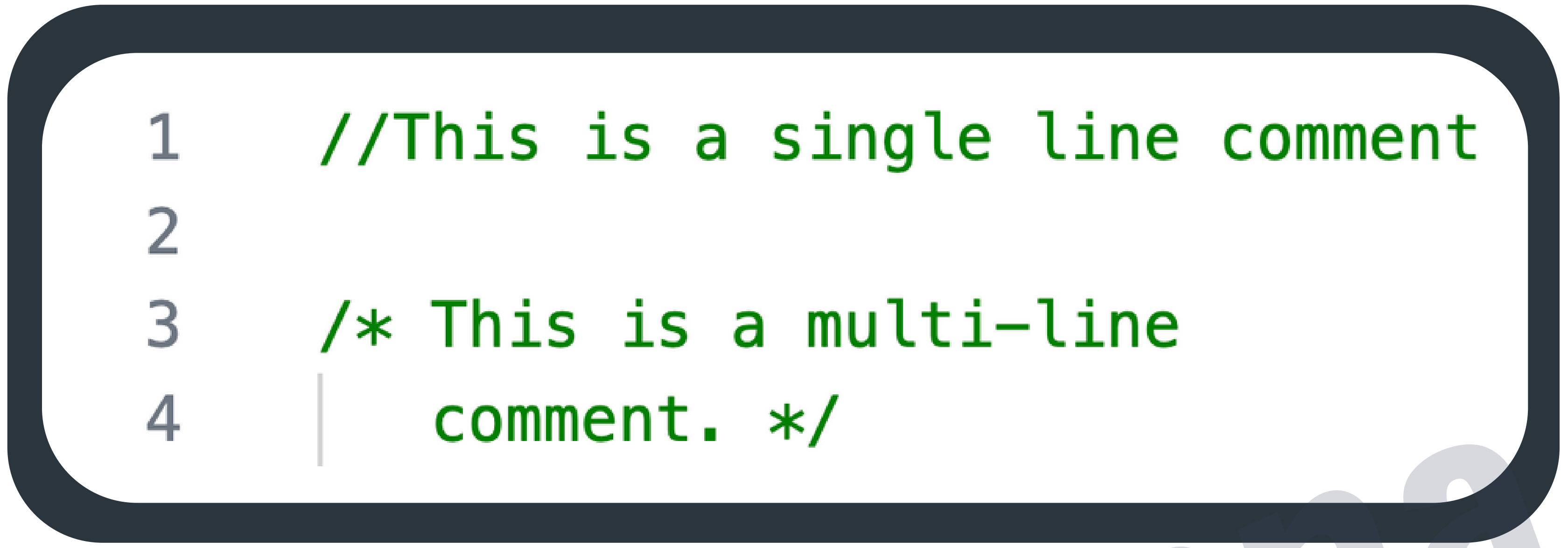
Entrepreneur

Apna College | Ex-Microsoft, DRDO

To educate someone is the highest privilege❤️

Comments in JS

Part of Code which is not executed



```
//This is a single line comment

/* This is a multi-line
   comment. */
```

Operators in JS

Used to perform some operation on data

Arithmetic Operators

+, -, *, /

- Modulus
- Exponentiation
- Increment
- Decrement

Operators in JS

Assignment Operators

= += -= *= %= **=

Operators in JS

Comparison Operators

Equal to	==	Equal to & type	===
Not equal to	!=	Not equal to & type	!==

>, >=, <, <=

Operators in JS

Logical Operators

Logical AND	&&
Logical OR	||
Logical NOT	!

Conditional Statements

To implement some condition in the code

if Statement

```
let color;
if(mode === "dark-mode") {
    color = "black";
}
```

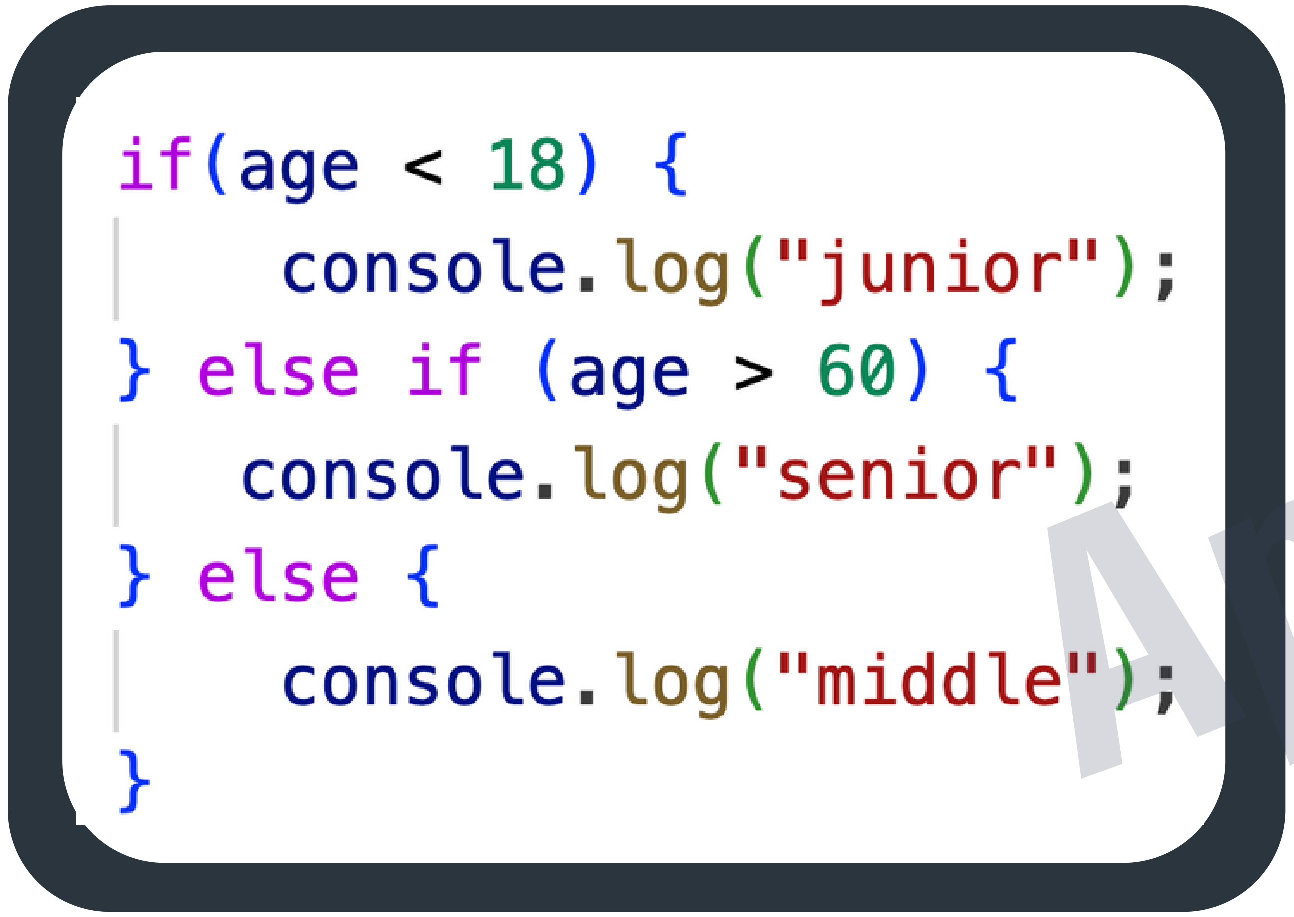
Conditional Statements

if-else Statement

```
let color;
if(mode === "dark-mode") {
    color = "black";
} else {
   color = "white";
}
```

Conditional Statements

else-if Statement



```
if(age < 18) {
    console.log("junior");
} else if (age > 60) {
   console.log("senior");
} else {
    console.log("middle");
}
```

Operators in JS

Ternary Operators

condition ? true output : false output

```
age > 18 ? "adult" : "not adult";
```

MDN Docs

Let‘s Practice

Qs1. Get user to input a number using prompt(“Enter a number:”). Check if the number is a multiple of 5 or not.

Let‘s Practice

Qs2. Write a code which can give grades to students according to their scores:

- 80-100, A
- 70-89, B
- 60-69, C
- 50-59, D
- 0-49, F

Loops in JS

Loops are used to execute a piece of code again & again

for Loop

```
for (let i = 1; i <= 5; i++) {
    console.log("apna college");
}
```

Loops in JS

Infinite Loop : A Loop that never ends

Loops in JS

while Loop

```
while (condition) {

    // do some work

}
```

Loops in JS

do-while Loop

```
do {

    // do some work

} while (condition);
```

Loops in JS

for-of Loop

```
for (let val of strVar) {

    //do some work

}
```

Loops in JS

for-in Loop

```
for (let key in objVar) {

    //do some work

}
```

Let‘s Practice

Qs1. Print all even numbers from 0 to 100.

Let‘s Practice

Qs2.

Create a game where you start with any random game number. Ask the user to keep guessing the game number until the user enters correct value.

Strings in JS

String is a sequence of characters used to represent text

Create String

```
let str = “Apna College“;
```

String Length

```
str.length
```

String Indices

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

Template Literals in JS

A way to have embedded expressions in strings

```
`this is a template literal`
```

String Interpolation

To create strings by doing substitution of placeholders

```
`string text ${expression} string text`
```

String Methods in JS

These are built-in functions to manipulate a string

- str.toUpperCase( )
- str.toLowerCase( )
- str.trim( ) // removes whitespaces

String Methods in JS

- str.slice(start, end?) // returns part of string
- str1.concat(str2) // joins str2 with str1
- str.replace(searchVal, newVal)
- str.charAt(idx)

Let‘s Practice

Qs1. 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 fullname length.

eg: user name = “shradhakhapra” , username should be “@shradhakhapra13”

Arrays in JS

Collections of items

Create Array

let heroes = [“ironman”, “hulk”, “thor”, “batman”];

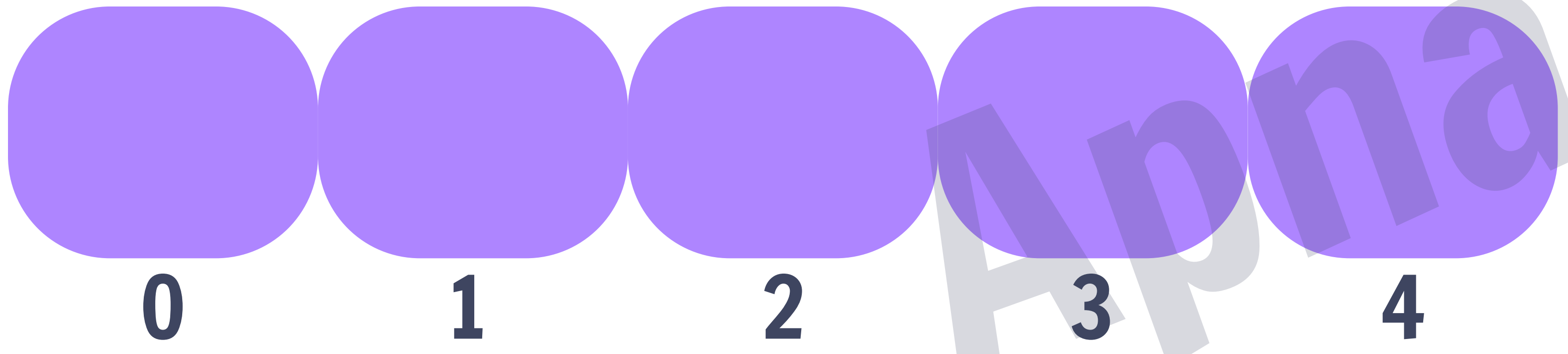
let marks = [96, 75, 48, 83, 66];

let info = [“rahul”, 86, “Delhi”];

Arrays in JS

Array Indices

arr[O], arr[1], arr[2] ....



Looping over an Array

Print all elements of an array

Let‘s Practice

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

Let‘s Practice

Qs. For a given array with prices of 5 items -> [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.

Arrays in JS

Array Methods

Push( ) : add to end

Pop( ) : delete from end & return

toString( ) : converts array to string

Arrays in JS

Array Methods

Concat( ) : joins multiple arrays & returns result

Unshift( ) : add to start

shift( ) : delete from start & return

Arrays in JS

Array Methods

Slice( ) : returns a piece of the array

```
slice( startIdx, endIdx )
```

Splice( ) : change original array (add, remove, replace)

```
splice( startIdx, delCount, newEl1... )
```

Let‘s Practice

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

a. Remove the first company from the array

b. Remove Uber & Add Ola in its place

c. Add Amazon at the end

Functions in JS

Block of code that performs a specific task, can be invoked whenever needed

Functions in JS

Function Definition

```
function functionName( ) {

    //do some work

}


function functionName( param1, param2 ...) {

    //do some work

}
```

Function Call

```
functionName( );
```

Arrow Functions

Compact way of writing a function

```
const functionName = ( param1, param2 ...) => {
    //do some work
}
```

```
const sum = ( a, b ) => {
    return a + b;
}
```

Let‘s Practice

Qs. Create a function using the “function” keyword that takes a String as an argument & returns the number of vowels in the string.

Qs. Create an arrow function to perform the same task.

forEach Loop in Arrays

arr.forEach(callBackFunction)

CallbackFunction : Here, it is a function to execute for each element in the array

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

```
arr.forEach( ( val ) => {

    console.log(val);

} )
```

Let‘s Practice

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

Some More Array Methods

Map

Creates 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))

```
let newArr = arr.map( ( val ) => {

    return val * 2;

} )
```

Some More Array Methods

Filter

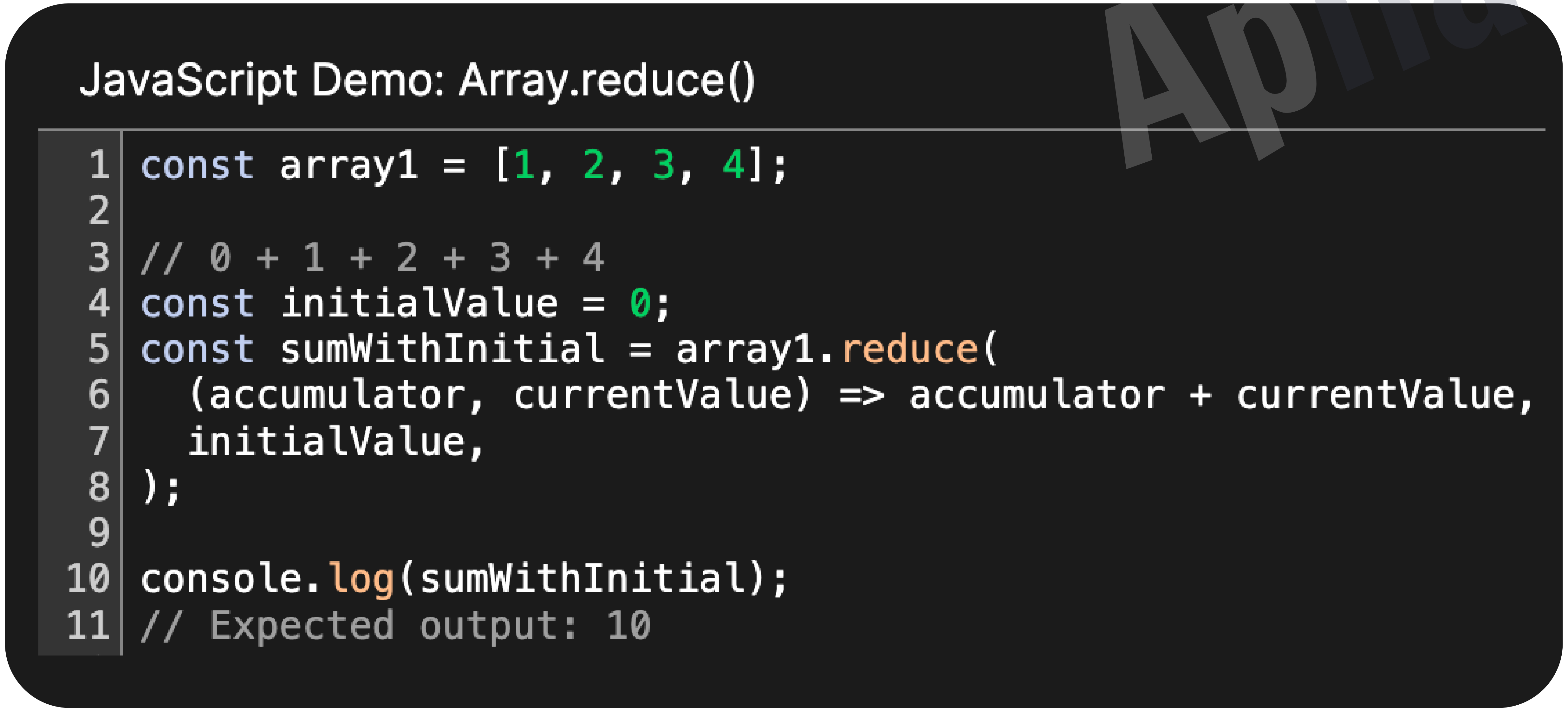
Creates a new array of elements that give true for a condition/filter.
Eg: all even elements

```
let newArr = arr.filter( ( ( val ) => {

    return val % 2 === 0;

} )
```

Some More Array Methods

Reduce

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



JavaScript Demo: Array.reduce()

```
const array1 = [1, 2, 3, 4];

// 0 + 1 + 2 + 3 + 4
const initialValue = 0;
const sumWithInitial = array1.reduce(
  (accumulator, currentValue) => accumulator + currentValue,
  initialValue,
);

console.log(sumWithInitial);
// Expected output: 10
```

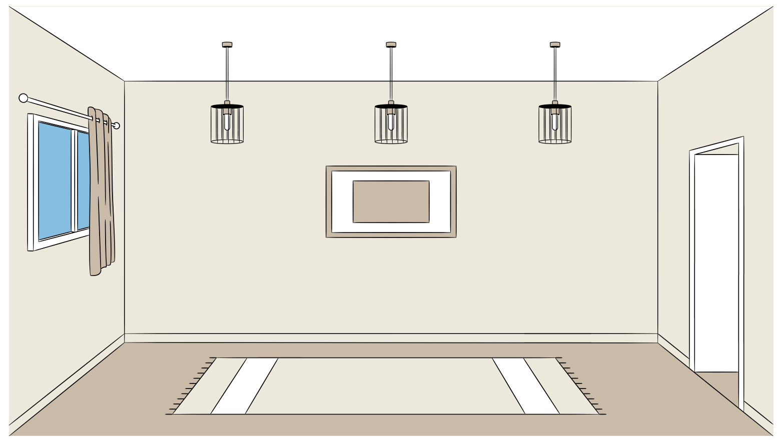
Let's Practice

Qs. We are given array of marks of students. Filter our of the marks of students that scored 90+.

Qs. Take a number n as input from user. Create an array of numbers 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.

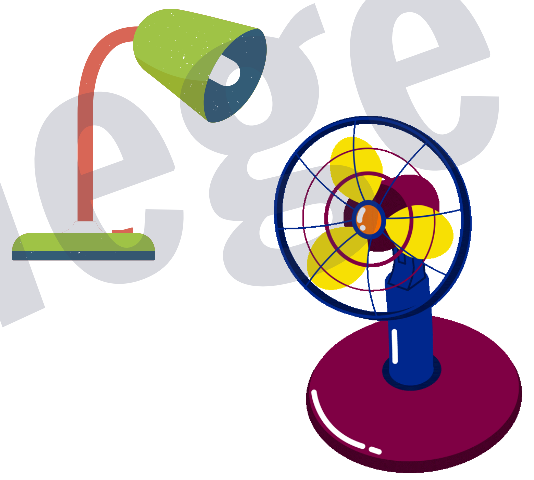
The 3 Musketeers of Web Dev

HTML
(structure)



CSS
(style)

JS
(logic)



Starter Code

<style> tag connects HTML with CSS

<script> tag connects HTML with JS

```
<html>
        <head>
                <title> Website Name </title>
        </head>
        <body>
                <!-- Content Tags -->
        </body>
</html>
```

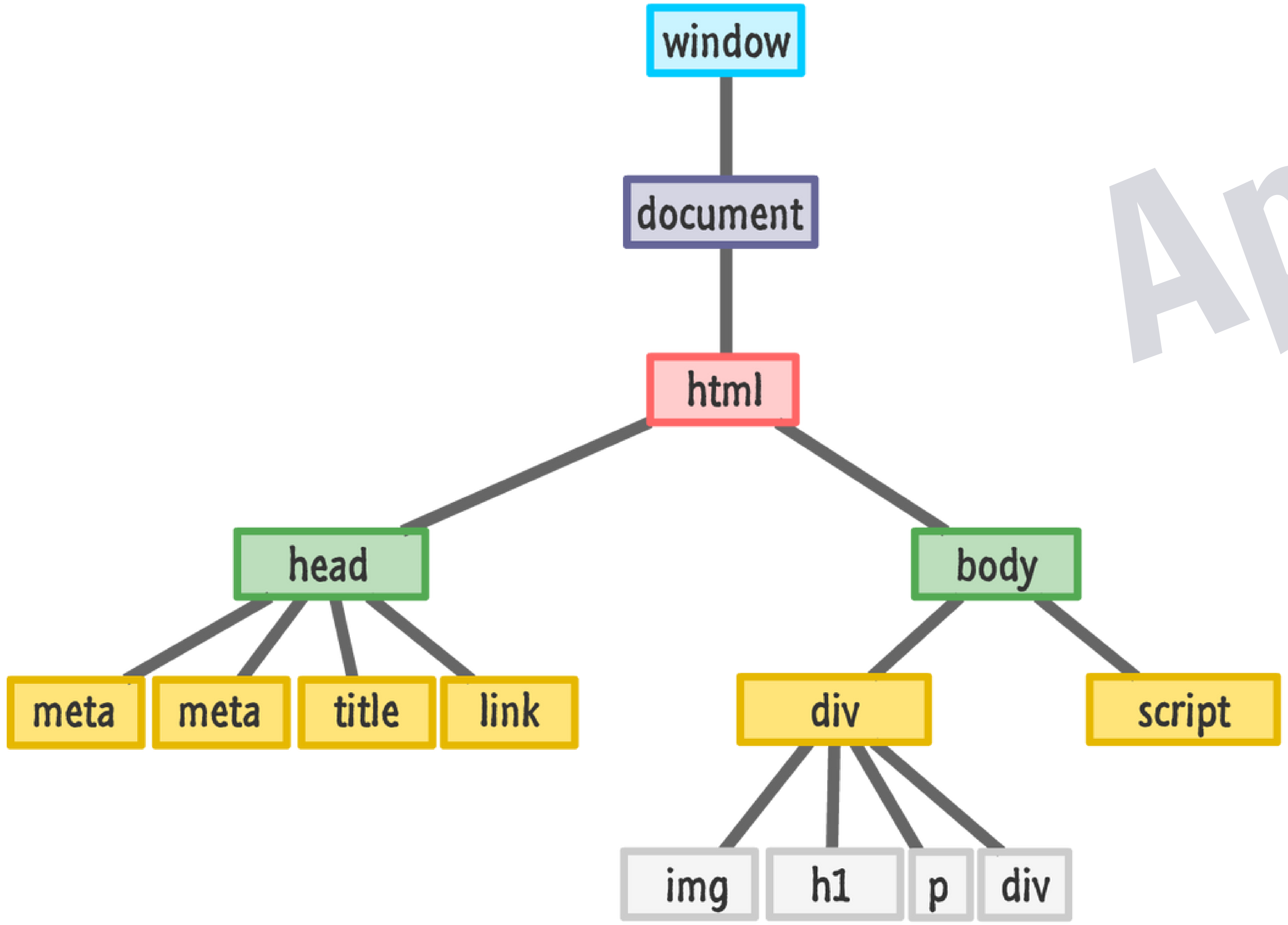
Window Object

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.

What is DOM?

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



DOM Manipulation

Selecting with id

```
document.getElementById("myId")
```

Selecting with class

```
document.getElementsByClassName("myClass")
```

Selecting with tag

```
document.getElementsByTagName("p")
```

DOM Manipulation

Query Selector

```
document.querySelector("#myId / .myClass / tag")
//returns first element
```

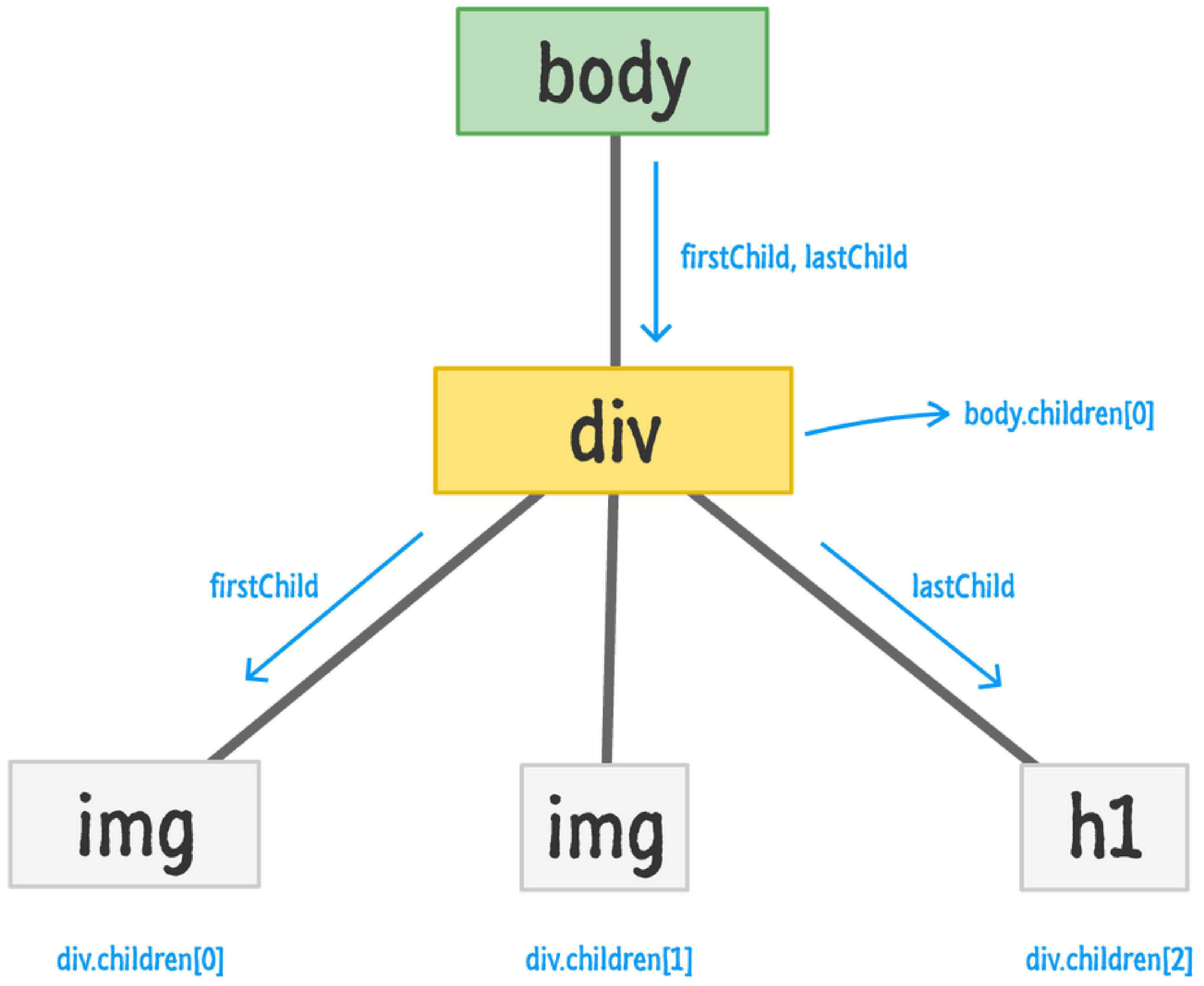
```
document.querySelectorAll("#myId / .myClass / tag")
//returns a NodeList
```

DOM Manipulation

Properties

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

Homework



body
firstChild, lastChild
div
body.children[0]
firstChild
lastChild
img
img
h1
div.children[0]
div.children[1]
div.children[2]

Let‘s Practice

Qs. Create a H2 heading element with text - “Hello JavaScript”. Append “from Apna College students” to this text using JS.

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

DOM Manipulation

Attributes

- getAttribute(attr) //to get the attribute value
- setAttribute(attr, value) //to set the attribute value

Style

- node.style

DOM Manipulation

Insert Elements

let el = document.createElement("div")

- node.append(el) //adds at the end of node (inside)
- node.prepend(el) //adds at the start of node (inside)
- node.before(el) //adds before the node (outside)
- node.after(el) //adds after the node (outside)

Delete Element

- node.remove( ) //removes the node

Let‘s Practice

Qs. Create a new button element. Give it a text “click me”, background color of red & text color of white.

Insert the button as the first element inside the body tag.

Qs. Create a <p> tag in html, give it a class & some styling.

Now create a new class in CSS and try to append this class to the <p> element.

Did you notice, how you overwrite the class name when you add a new one?
Solve this problem using classList.

Events in JS

The change in the state of an object is known as an Event

Events are fired to notify code of "interesting changes" that may affect code execution.

- Mouse events (click, double click etc.)
- Keyboard events (keypress, keyup, keydown)
- Form events (submit etc.)
- Print event & many more

Event Handling in JS

```
node.event = ( ) => {
    //handle here
}
```

example

```
btn.onclick = ( ) => {
    console.log("btn was clicked");
}
```

Event Object

It is a special object that has details about the event.

All event handlers have access to the Event Object's properties and methods.

```
node.event = (e) => {
    //handle here
}
```

e.target, e.type, e.clientX, e.clientY

Event Listeners

node.addEventListener(event, callback)

node.removeEventListener(event, callback)

*Note : the callback reference should be same to remove

Let's Practice

Qs. Create a toggle button that changes the screen to dark-mode when clicked & light-mode when clicked again.

Classes & Objects

Prototypes in JS

A javaScript object is an entity having state and behavior (properties and method).

JS objects have a special property called prototype.

We can set prototype using _ _ proto _ _

*If object & prototype have same method,
object's method will be used.

Classes in JS

Class is a program-code template for creating objects.

Those objects will have some state (variables) & some behaviour (functions) inside it.

```
class MyClass {
    constructor( ) { ... }
    myMethod( ) { ... }
}

let myObj = new MyClass( ) ;
```

Classes in JS

Constructor( ) method is :

- automatically invoked by new
- initializes object

```
class MyClass {
    constructor( ) { ... }
    myMethod( ) { ... }
}
```

Inheritance in JS

inheritance is passing down properties & methods from parent class to child class.

```
class Parent {

}

class Child extends Parent {

}
```

*If Child & Parent have same method, child's method will be used. [Method Overriding]

super Keyword

The super keyword is used to call the constructor of its parent class to access the parent's properties and methods.

```
super( args )  // calls Parent‘s constructor

super.parentMethod( args )
```

Let‘s Practice

Qs. You are creating a website for your college. Create a class User with 2 properties, name & email. It also has a method called viewData( ) that allows user to view website data.

Qs. Create a new class called Admin which inherits from User. Add a new method called editData to Admin that allows it to edit website data.

Error Handling

try-catch

```
try {
    ... normal code
} catch ( err ) { //err is error object
    ... handling error
}
```

What this chapter is about?

async await >> promise chains >> callback hell

Sync in JS

Synchronous

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.

Asynchronous

Due to synchronous programming, sometimes imp instructions get
blocked due to some previous instructions, which causes a delay in the UI.
Asynchronous code execution allows to execute next instructions
immediately and doesn't block the flow.

Callbacks

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

Callback Hell

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

Promise is for "eventual" completion of task. It is an object in JS.

It is a solution to callback hell.

$$\texttt{let promise = new Promise(}\ \underbrace{\texttt{(resolve, reject) => \{ .... \}}}_{\text{Function with 2 handlers}}\ \texttt{)}$$

*resolve & reject are callbacks provided by JS

Promises

A JavaScript Promise object can be:

- Pending : the result is undefined
- Resolved : the result is a value (fulfilled) resolve(result)
- Rejected : the result is an error object reject(error)

*Promise has state (pending, fulfilled) & some result (result for resolve & error for reject).

Promises

.then( ) & .catch( )

```
promise.then( ( res ) => { .... } )

promise.catch( ( err ) ) => { .... } )
```

Async-Await

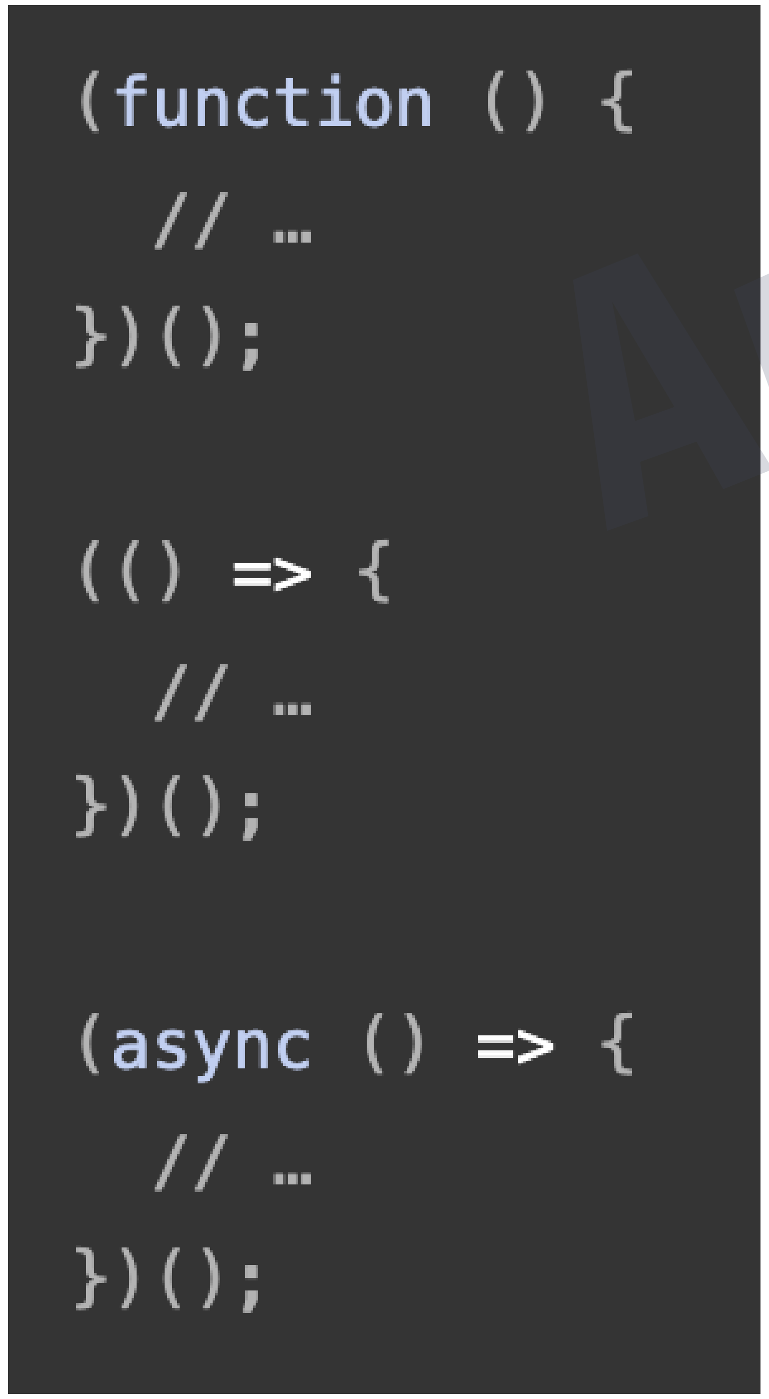
async function always returns a promise.

```
async function myFunc( ) { .... }
```

await pauses the execution of its surrounding async function until the promise is settled.

IIFE : Immediately Invoked Function Expression

IIFE is a function that is called immediately as soon as it is defined.



```
(function () {
  // …
})();

(() => {
  // …
})();

(async () => {
  // …
})();
```