# Day-17

# Variables id and Class Access

1. let greeting = "Hello, world!";

2. let element = document.getElementById("myId");

3. let elements = document.getElementsByClassName("myClass");

4. getElementById("id") returns an element with a specific ID.

querySelector("selector") returns the first element that matches a CSS selector like #id, .class, or a tag.  
Example:

document.querySelector("myId");  
document.querySelector("#myId");  
document.querySelector(".myClass");

5. document.getElementById("myId").textContent = "New Text";

6.document.querySelector(".myClass").style.backgroundColor = "lightblue";

7.document.getElementById("myId").classList.add("newClass");

8.document.getElementById("myId").classList.toggle("active");

# OPERATORS

9. 5 == '5' → true  
Because == allows type coercion ('5' becomes number 5).

10. == checks only value (with type conversion),  
 === checks both value and type (strict comparison).  
Example:  
5 == "5" → true  
5 === "5" → false

11. let avg = (a + b + c) / 3;

12. 10 % 3 → 1  
% gives the remainder when dividing 10 by 3.

13. 4 + true → 5  
true is treated as 1, so 4 + 1 = 5.

14. let result = (num > 0) ? "Positive" : "Not Positive";

15. !== means “not equal in value or type”.  
Example: 5 !== "5" → true

16. Type of null → "object" (this is a JavaScript bug/quirk)

# ARRAYS

17. let students = ["Alice", "Bob", "Charlie"];

18. let lastItem = array[array.length - 1];

19. array.unshift("NewItem"); (adds item to start)

20. array.pop(); (removes last item)

21. for (let i = 0; i < array.length; i++) {  
 console.log(array[i]);  
}

22. array.length returns number of elements in the array.

23. array.forEach(function(item) {  
 console.log(item);  
});

24. Array.isArray(variable)-- checks if the variable is an array.

# ARROW FUNCTIONS

25. const multiply = (a, b) => a \* b;

26. Traditional function:

function add(a, b) {

return a + b;

}  
Arrow function:

const add = (a, b) => a + b;

27. const square = num => num \* num;

28. const greet = name => ‘Hello, ${name}!’;

29. let numbers = [1, 2, 3, 4];

let doubled = numbers.map(num => num \* 2);

30. Arrow functions do not have their own this. They inherit this from the surrounding scope.25. const multiply = (a, b) => a \* b;

🡪Regular functions have their own this depending on how they are called.

# EVENT HANDLING

31. document.getElementById("myButton").addEventListener("click", function() {

//

});

32. document.getElementById("myButton").addEventListener("click", function() { document.getElementById("myPara").textContent = "Text changed!";

});

33. document.getElementById("myElement").addEventListener("mouseover", function() {

//your code here

});

34. document.getElementById("myInput").addEventListener("input", function(event) {

console.log(event.target.value);

});

35. preventDefault() stops the default action of an event e.g. prevents form from submitting or a link from opening. It’s used to handle behavior manually.

# FORMS

# 36.let value = document.getElementById("myInput").value;

37. let input = document.getElementById("myInput").value;

if (input === "") {

alert("Field cannot be empty");

}

38. document.getElementById("myForm").addEventListener("submit", function(event) {

event.preventDefault();

/\* handle form data \*/

});

39. Use event.preventDefault(); inside the submit event handler to stop the page from refreshing.  
Example: form.addEventListener("submit", function(event) {

event.preventDefault();

});

40.document.getElementById(“my-Form).reset();