**Day-17**

**Variables, ID & Class Access**

1. let name = "Amitha";

2. document.getElementById("myId");  
  
3. document.getElementsByClassName("myClass");

4. getElementById() returns one element by ID. querySelector() returns the first matching element for any CSS selector

5. document.getElementById("myId").innerText = "New text";

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

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

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

**Operators**

9. true – because == does type coercion.

10. == compares value only. === compares value and type.

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

12. 1

13. 5

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

15. Not equal value or not equal type.

16. Returns 'object'

**Arrays**

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

18. students[students.length - 1];  
  
19. students.unshift("New Student");  
  
  
  
20. students.pop();  
  
21. for (let i = 0; i < students.length; i++) {  
 console.log(students[i]);  
}  
  
22. The number of elements in the array.

23. students.forEach(student => console.log(student));  
  
24. Array.isArray(students); // true

**Arrow Functions**

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

26. function add(a, b) {  
 return a + b;  
}  
  
const add = (a, b) => a + b;

27. const square = num => num \* num;  
  
28. const greet = name => `Hello, ${name}!`;

29. let numbers = [1, 2, 3];  
let doubled = numbers.map(num => num \* 2);

30. Arrow functions do not have their own this; they inherit this from their parent scope.  
Regular functions have their own this depending on how they are called.  
  
**Event Handling**

31.document.getElementById("myBtn").addEventListener("click", function() {  
 alert("Button clicked");  
});

32. document.getElementById("myBtn").addEventListener("click", function() {  
 document.getElementById("myPara").innerText = "Text changed!";  
});

33. document.getElementById("hoverMe").addEventListener("mouseover", function() {  
 console.log("Mouse over");  
});  
  
  
  
34. document.getElementById("myInput").addEventListener("input", function() {  
 console.log(this.value);  
});

35. Prevents the default behavior of an event .

**Forms**

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

37. if (document.getElementById("myInput").value === "") {  
 alert("Field cannot be empty");  
}

38. document.getElementById("myForm").addEventListener("submit", function(e) {  
 e.preventDefault();  
 });

39. Use e.preventDefault() in the submit event handler.

40. document.getElementById("myForm").reset();