Page No.:
Date:

DAY 14

Java Script

including External JS file

console.log

info

warn

error

user input

Alert

conform

variables, keywords (var, let, const)

JS staments and semicolons

comments

diffrence between Expression & Statement

Arithmetic operators

mcriment

Decrement

DAY 15

primitive Data types in JS

1. Numbers
2. String
3. Boolean
4. Null
5. Undefined
6. Symbol
7. BigInt

Reference Data types

1. object
2. Array

Some important values in Java Script

1. undefined
2. null
3. NaN
4. Infinity

Page No.:
Date:

Basic operators in JS

1. Arithmetic operators (+, -, *, /, %, ++, --)
2. Assigment operators (=, +=, -=, *=, /=, %=)
3. comparison operators (==, ===, !=, !==, >, <, <=, >=)
4. logical operators (&&, ||, !)

variable Hosting (IMP)

IF-else Statement

Ternary operator

loops

1. for
2. while
3. Do while

DAY 16

Revision & function in JS

DAY 17

1. working with strings in JS
    1. slice()
    2. Template String
    3. split()
    4. replace()
    5. includes()
2. conditions
    IF-else, (sab par+)
    switch case
3. loops
    three main loops
    4. For Each loop (for Array)
    5. For...in loop (for object)
    6. for...of loop (for iterables)

Page No.:
Date:
YOUVA

4. Functions in Java Script

1. Regular Function
2. Arrow Function
3. Immediately Invoked Function Expression (IIFE)
4. Higher-order Function

In Lecture

1) Fnc statement

```
function abcd()
{
    // Fnc statement
}
```

2) Fnc Expression

```
var abcd = function()
{
    // Fnc Expression
}
```

3) Anonymous function

```
function()
{
    // Fnc anonymous
}
```

4) Fat Arrow function

```
() =>
{
    // Fat arrow fnc
}
```

5) Fat arrow with one Parameter

```
let abcd = (a) =>
{
}
abcd(12);
```

6) Fat arrow with implicit return

```
var abcd = () => "meet";
var ans = abcd();
```

Page No.:
Date:

DAY 18

Rest parameters

```
function abcd(a, b, c, ....chacha)
{

}
abcd(1,2,3,4,5);
```

output: a:1
b:2
c:3
chacha = [4,5]

variable Hosting

IIFE :- immediately invoked function expression

```
(function abcd())
{
    console.log("Hey");
}
)();
```

output :- Hey

→ So, what the difference between IIFE and Regular function ?

→ But in this IIFE function created variable are local behind the brecets of fnc we can't use this variables can not expose this variables and can not update on console.

HOFS - Higher order function returns (both)

→ such a function that ~~returns~~ takes onether function or function accept the onether function in perameters or may be both.

```
function abcd()
{
    return function()
    {
        console.log("HOFS");
    }
}
abcd()();
```

we can, var ans = abcd();
ans();

Ex-2
```
function abcd(fnc)
{
    fnc();
}
abcd(function())
    {
        console.log("HOFS");
    })
```

CB fnc - call back function

→ when we call any function and in argument we sent another function this is called cb fnc.

```
function abcd(val)
{                    (→ And this HOF fnc
}
abcd(function())
    {
      // call back function
    });
```

first class fnc

→ first class fnc is an credit in JS and JS give this credit to fnc. and this credit says, treate a function as a value. and every function can called first class function.

pure function

A function without side effects mean does not modify external state. (Global values) it's called pure function.

```
function add(a,b)
{    return a+b;
}
console.log(add(2,3));
```

Page No.:
Date:

impore function

→ This fnc totaly diffrent from pore fnc.

```
let total = 0;
function addTotal(a)
{
    total += a;
}
addToTotal(5);
console.log(total); 115
```

closer - closer order fnc

This COF is consept that says when function retarn onether fnction and retarned fnc can use their parente fnc's data, variables, members it's called closer.

```
function abcd()
{
    var a = 12;
    retarn function()
    {
        console.log(a);
    }
}
var ans = abcd();
ans();
```

Scoping & closares in javascript

Page No.:
Date:
YOUVA

DAY 20

Arrays and objects

Arrays :-

A collection of matiple values with similore data type.

Ex :- var A = [12, 13, 14];

But in Java script we can store maltiple data types of value and maltiple values. it not homogeneous

Ex :- var arr = ["a", 12, function() {}, {}, [], NaN, undefind];

object :-

object is one type of way to hold key value pairs.

And when you want to store the info of one entity.

DAY 21

practice session -1

DAY 22

map

→ when we need to Ram loop at any Array and add element in new array thum we use map.

Ex :- map give an blanke array.

```
var arr = [1, 2, 3, 4];
arr.map(function(value)
{
    return 1;
})
```

output :- 1 1 1 1

→ map can return

Page No.:
Date:
YOUVA

ForEach

→ when we need to oprration in any array. than we can use of foreach

→ Foreach can not return.

Ex:-
```
var arr = [1, 2, 3, 4];
arr.forEach(function(value){
    console.log(value);
});
```

Filter

→ Filter work as it as map but filter return only two values true and false (Boolean) and filter also create blank array.

Ex:-
```
var arr = [1, 2, 3, 4]
arr.filter(function(value){
    return true;
});
```

false → array
Blank

value > 2
↳ 3, 4

output = 1, 2, 3, 4

Redase

→ map return multiple value, filter can return two Boolean values and Redase can return only one value.

Ex:-
```
var arr = [1, 2, 3, 4];

arr.redase(function(accumulator, key){
    return accumulator + key;
}, 0);
```

key ↑ array value

↑ accumulator starting value.

output :- 10