HOF : Function is calling a function
Are functions that take other functions as parameters or returns a function as a value. The function passed as a parameter is called callback.

```
const callback = (n) => {
    return n * * 2;
};
function cube (callback, n){
    return callback(n) * n;
}
console.log (cube ( callback, n ));
```

callback

HOF

```
const arr = ["hey", "hi", "hello", "hola", "Namaste"]
arr.forEach (myfunc)
function myfunc (val) {
    console.log (val);
}
```

output: hey
hi
hello
hola
Namaste

* callback : Function as an argument

```
arr.forEach ( (val) => {
    console.log (val);
});
```

* forEach() is HOF, the () => { } inside it is callback

Set Timeout

```
setTimeout (() => {console.log ("Hello");
}, 3000);
```

```
function Mayur (){
  console.log ("Hello");
};
setTimeout (Mayur, 1000);
```

or

```
setTimeout (() => {
  console.log ("Hello");
});
```

Set Interval

```
setInterval (() => {
  console.log ("Hi");
}, 1000);
```

* map
* filter
* reduce
* find
* every
* some
* sort

```
const numbers = [1, 2, 3, 4, 5, 6];
const numsquare = numbers.map((num) => num * num);
console.log (numsquare);
```

output : [1, 4, 9, 16, 25, 36]

*map returns an array

Filter

```
const count = ["Hi", "Hello", "Hola", "Finland"];
const store = count.filter((val) => val.includes("land"));
console.log(store);
```

output : [Finland] returns an array

Reduce

```
const num = [1,2,3,4,5,6,7]
const sum = num.reduce((accumulator, current) =>
  accumulator + current , 0);
console.log(sum);
```

accumulator → duty of monitor to add all the wrappers : duty
current → monitor will be moving in array from 1-7
0 → monitors wrapper

output : 28

if acc - current ⇒ output = -28

```
// Reduce
// Reduce takes a callback function. The call back function takes accumulator, current, and optional initial value as a parameter and returns a single value. It is a good practice to define an initial value for the accumulator value. If we do not specify this parameter, by default accumulator will get array first value. If our array is an empty array, then Javascript will throw an error.
```

Sort

```
const names = ["Anurag", "Anirudh", "Hitesh", "Vyom"];
console.log(names.sort);
```

names.reserve

output :

```
[
  'Anirudh', 'Anurag',
  'Bishal',  'Hitesh sir ',
  'Mayur',   'Surya',
  'Vyom',    'momin',
  'snehal'
]
```

Destructing , Spread and Rest

* const sci = [2.71, 3.14, 9.81, 37, 100];
  let [e, pi, gra, kodyten, boiltemp] = sci;
  console.log(e, pi, gra, kodyten, boiltemp);
  output : 2.71 3.14 9.81 37 100

destructuring

* const arr1 = [1,2,3]
  let [var1,,var2] = arr1;
  console.log(var1,,var2);
  output : 1 3

* const arr2 = [1,2,3,4,5,6,7,8];
  let [num1, num2, ... rest] = arr2
  console.log(num1, num2);
  console.log(rest);
  output : 1 2
  [3, 4, 5, 6, 7, 8]