Episode 18

Higher-order functions and functional programming

* What is Higher order Functions?

A function which takes another function as an argument or returns a function from it is known as Higher order function.

example :-

```
Function x(){
    console.log("Namaste");
}
function y(x){
    x();
}
```

callback function

Here ~~y is taking~~ function y is taking another function x as an argument, therefore y is higher order function.

* Introduction to functional programming

example :-

```
~~const radius = [3,2,~~
const radius = [3,1,2,4];
const calculateArea = function (radius){
    const output = [];
    for(let i=0; i<radius.length; i++){
        output.push(Math.PI * radius[i] * radius[i];
    }
```

```
    return output;
};
console.log(calculateArea(radius));

const calculateCircumference = function(radius) {
    const output = [];
    for(let i=0; i<radius.length; i++) {
        output.push(2*Math.PI*radius[i]);
    }
    return output;
};
console.log(calculateCircumference(radius));

const calculateDiameter = function(radius) {
    const output = [];
    for(let i=0; i<radius.length; i++) {
        output.push(2*radius[i]);
    }
    return output;
};
console.log(calculateDiameter(radius));
```

⇓ output

- ▶ (4) [28.2743, 3.1415, 12.5663, 50.2654]
- ▶ (4) [———]
- ▶ (4) [6, 2, 4, 8]

[this works good, but this is not good way]

this code is not modular, not reusable.

* DRY Principle - Don't Repeat Yourself

* Optimized Code (interview recommended)

for doing this, we will make use of functional programming

```
const radius = [3,1,2,4];
const area = function (radius) {
    return Math.PI * radius * radius;
};
const circumference = function (radius) {
    return 2 * Math.PI * radius;
};
const diameter = function (radius) {
    return 2 * radius;
};
const calculate = function (radius, logic) {
    const output = [];
    for (let i=0; i<radius.length; i++) {
        output.push (logic (radius[i]));
    }
    return output;
};
console.log (calculate (radius, area));
console.log (calculate (radius, circumference));
console.log (calculate (radius, diameter));
```

↓

same output as before

Page No.
Date

* Beauty of functional programming

Functional Programming is huge in itself but a small part of it says that think or make logic in your head according to functions.

* pollyfil for map function in javascript

- map is higher order function.

```
const radius = [3,1,2,4];
const area = function(radius){
    return Math.PI * radius * radius;
};
Array.prototype.calculate = function(logic){
    const output = [];
    for(let i=0; i<this.length; i++){
        output.push(logic(this[i]));
    }
    return output;
};
console.log(radius.map(area));
console.log(radius.calculate(area));
```

> same output

* Higher order functions ~~and functional programming~~ is only possible because, functions are ~~it~~ first class citizens.