Page No.
Date

episode 13

first class functions / Anonymous Javascript -

* - what is a function statement in javascript

```
function a(){
        console.log("a called"); ⇒ a called
}
a();
```

* - What is a function expression.

```
var b = function (){
        console.log("b called"); ⇒ b called
}
b();
```

* - Difference between function statement and function expression
[difference is in hoisting]

✓ function statement
[Function statement are hoisted]

```
a();
function a(){
        console.log("a called"); ⇒ a called
}
```

✓ function expression
[Function expressions are not hoisted]

```
b();
var b = function(){
        console.log("b called"); ⇒ TypeError : b is not a function
}
```

[[this is all there is to be kept in mind for creating functions from a hoisting point of view]]

Page No.
Date

* - function declaration

function statement and function declaration is same.

* - Anonymous function in javascript

IMP

function without a name is called anonymous function.

example

```
function (){

}
```

[Anonymous functions does not have their own identity]

* - Syntax error and Anonymous functions

```
function (){

}
```

⇒ Syntax Error: function statement requires a function name

* - uses/advantages of anonymous functions :→

IMP

Anonymous functions are used in a place where functions are used as values.

in function expression example, it is an anonymous function i.e. function without name

[we can assign anonymous function to a variable]

example:→

```
var b = function () {
    console.log ("Hello");
};
```

[this is anonymous function expression]

Page No.
Date

* - What are Named function expression in javascript

```
var b = function xyz(){
    console.log("b called");
}
b();
```

$\Rightarrow$ b called

[here instead of using anonymous function, we are using function with a name]

Gotcha

corner case

error

```
var b = function xyz(){
    console.log("b called");
}
xyz();
```

$\Rightarrow$ ReferenceError: xyz is not defined

* - What is the difference between parameters and arguments

Parameters

```
var b = function (param1, param2) {
    console.log("b called");
}
b(1,2);
```

arguments

```
var b = function xyz(){
    console.log(xyz);
}
b();
```

$\Rightarrow$

```
f xyz() {
    console.log(xyz);
}
```

[outside we cannot access only inside we can access]

[the ability to used functions as values and can be passed as an argument to another function and can be returned from functions is called first class functions.]

* - First class functions in javascript

The ability to used functions as values is called as first class function.

example :

```
var b = function (param1){
      return function xyz(){          ⇒   f xyz(){

      }
}                                          }
console.log(b());
```

* - functions are first class citizens [First class functions] same as first class functions i.e. ability to be used like values.