**Video-First Class Functions In JavaScript ->**

**// console.log(a);**

**// console.log(b);**

**// a();**

**// b();**

**// 1. Function Statement**

**function a(){**

**console.log("a is called");**

**}**

**//a();**

**// 2. Function Expression - When we assign function to varaible or we can say when fucntion act like a value.**

**// when we assign function as a value to a variable**

**var b = function(){**

**console.log("b is called");**

**}**

**//b();**

**// 3. Difference between function Statement and function Expression**

**// Here Major diff is we can do Hoisting in Function Statement but not in Function Expression**

**// if We console.log(a) // complete function a() will be printed**

**// console.log(b) // undefined will be printed because memory is allocated to var b which is undefined**

**// a(); // complete function will run;**

**// b(); // Error will pop up ( Uncaught TypeError: b is not a function).**

**// 4. Function Declaration**

**// function declaration is same as function statement**

**// 5. Anonymous Function**

**// Anonymous function are those functions which does not have name and used to assign function**

**// as a value to a variable.**

**// ex-**

**var c = function(){**

**console.log("Anonymous function is assign as a value to a variable c");**

**}**

**c();**

**// 6. Named Function Expression**

**// Named function expression are same as anonymous function having name.**

**// Ex-**

**var d = function xyz(){**

**console.log("Named Function Expression is called having name xyz");**

**}**

**d();**

**// corner case OF Named Function Expression is we cannot directlty call a named function expression**

**// xyz(); // Uncaught Error: xyz is not defined.**

**// 7. Difference B/W parameters and Arguments**

**// parameters ->**

**function area(para1,para2){ // here para1 and para2 are known as parameters**

**return para1\*para2;**

**}**

**// Arguments ->**

**area(2,4); // here 2 and 4 are known as arguments which we are passing to functions.**

**// 8. First Class Functions**

**// First Class Function is the ability of functions to be used as values and can be passed as an value to an**

**// another functions and can be returned as an value from functions.**

**function tp(){**

**console.log("tp");**

**}**

**function topper(para){**

**console.log("taking function as a para and also return it as value");**

**return function(){**

**console.log("....");**

**}**

**}**

**console.log(topper(tp));**

**// First Class functions and first class citizens are same**

**Video 14-CallBack Functions In JavaScript (Event Listner) ->**

**// 1. What is a callback functions in javascript ...?**

**// => Functions are First Class Citizens in JavaScript**

**// => Means you can take a function and pass it to another functions and when you do so the function you pass it to another function is known as callback function.**

**setTimeout(function y(){**

**console.log("I am callback function");**

**},5000);**

**// here function y() is callback function which will execute in atleast after 5 sec.**

**// callback function give us the power of asynchronous in javaScript as JavaScript Engine does not wait for 5 sec to finish ... it will proceed further all codes written below.**

**function x(para){**

**console.log("x...");**

**para();**

**}**

**x(function tp(){**

**console.log("tp..."); // tp() is known as callback function.**

**});**

**// JavaScript is a Single threded And Synchronous Language(means having single call stack/Main thread)**

**//**

**// 2. Deep About Event Listener**

**// let count =0;**

**// document.getElementById("clickme").addEventListener("click",function xyz(){**

**// console.log("Button Logged",++count);**

**// });**

**// Here using global variable is not a good choice here any function can access to count varible which is not good.**

**// Closure Demo along with Event listener**

**function eventListnerAboveUsingClosure(){**

**let count =0;**

**document.getElementById("clickme").addEventListener("click",function xyz(){**

**console.log("Button Logged",++count);**

**});**

**}**

**eventListnerAboveUsingClosure();**

**// NoW here count variable will be hidden and can not be accessed by any other function inside program (Data Privacy)**

**// 3. Why do we need to remove EventListner ?**

**// => EventListener are very heavy ,it takes memory ... it maintain references with the variable even when callstack is empty.**

**// When we use setTimeout function , function we give inside is also known as callback function and store at another place( web/ browser API ) and put back in Call Stack when time is over.**

**// In AddEventListener the function we attach to event is also known as callback function. When the event is held by user we put that function into callstack( mean while function is store at Web API with that event.**