FUNCTION:

==>CALLBACK FUNCTION:

function show(a)

{

console.log(a);

---

}

show(10);

==>we can pass a entire function as a argument to the function and the passed function is called as

callback function.

ex:

function show()

{

console.log("hello world");

}

function demo(arg)

{

console.log(arg);

arg();

}

//we have to pass entire function as a argument.

demo(show);//pass reference of function show(); called as callback function // show() is callback function.

===================================================================================

<script>

function show()

{

console.log("hello world");

}

function demo(arg)

{

console.log(arg);//it will print the entire function definition.

}

demo(show);

</script>

==============================================================

//now we have to call the callback function which we have passed.

<script>

function show()

{

console.log("hello world");

}

function demo(arg)//receive entire function

{

console.log(arg);

arg();//it will call the passed function.

}

demo(show);

</script>

================================================================

function increment(num)

{

return num=5;

}

var x=3;

increment(x);//we can pass either 3 or x(as a reference)

function show()

{

console.log("hello world");

}

function demo(arg)//receive entire function

{

console.log(arg);

arg();//it will call the passed function.

}

demo(show);

//it can return as------------------------------------------------------------------------------------------

function demo(arg)//receive entire function

{

console.log(arg);

arg();//it will call the passed function.

}

demo(function show(){

console.log("hello world");}

);

================================================================================================

if show() is defined outside then the scope becomes global,if its global ,we can call function from anywhere.

but if we pass show() as a function argument then this function is defined as local scope.i.e show() will be accessible by demo function only.

whenever we want our function to be reuse everytime at many place then declare function globally.

=====================================================================================================================

==>why to give name to the function ,if we cant access this function outside anywhere?

Note:When a function is directly defined in the argument then there is no need to give its name because the passed function cannot be referred in the entire program. it can only be referred by the argument in the container function.

function demo(arg)//receive entire function

{

console.log(arg);

arg();//it will call the passed function.

}

demo(function ()

{

console.log("hello world");

});

//A function without name is called as anonymous function.

=================================================================================

In JS everything is an object, so a function is also an object. we can assign reference to an object(function)

var obj=function show()

{

}

function show()

{

console.log("hello world");

}

var obj=show;

console.log(obj);//it will print entire function

obj();//call the function

show();//

=========================================================================================<script>

var obj=function show()

{

console.log("hello world");

console.log(obj);

}

obj();

</script>

==============================================================

var obj=function show()

{

console .log("hello word");

}

console .log(obj);

==========================================================================

create 1 function that will add 2 numbers

const sum=function(a,b)

{

return a+b;

}

var rs=sum(1,2);

console.log(rs);

=========================================

js has introduced better way of writing/defining anonymous function .that new syntax is known as fat arrow function.

syntax:

const ref=()=>

{

}

const show=()=>{

console.log("hello world");

show();

\*\*\*\*

//const show=()=>console.log("hello world");

==========================================================

const sum=(a,b)=>{

return a+b;

}

rs=sum(1,2);

console.log(rs);

======================================================

const increment=(n)=>{

return n+5;

}

increment(2);

===============================================

const increment=(n)=> n+5;

increment(2);

============================================================================================

filter says pass the call back().and filter() will call callback() for each and every element of an array and element will be collected in item argument.

var arr=[1,2,3,4];

var evenarray=

arr.filter( (item)=>{

if(item%2==0)

return true;

else

return false;

});

console.log(evenarray);

=======================================================================

var arr=[1,2,3,4];

arr.forEach((item)=>{

});

======