Introduction to functions

A function is a named block of code. A function can be created using **function** keyword, followed by name of the function (**functionName**) and parameters (**param1**, **param2**,…). After that it also contains the statements which forms the body of the function inside curly braces({}).

Syntax:

1. function functionName(param1, param2,…){
2. *//statements*
3. }

Example:

1. function myFunction(num1,num2){
2. num3=num1\*num2;
3. return num3
4. }

In the above example we have a function named **myFunction**, which has parameters **num1**and **num2**. The body of the function is placed inside the curly braces.

A function is a named block of code. A function can be created using **function** keyword, followed by name of the function (**functionName**) and parameters (**param1**, **param2**,…). After that it also contains the statements which forms the body of the function inside curly braces({}).

Syntax:

1. function functionName(param1, param2,…){
2. *//statements*
3. }

Example:

1. function myFunction(num1,num2){
2. num3=num1\*num2;
3. return num3
4. }

In the above example we have a function named **myFunction**, which has parameters **num1**and **num2**. The body of the function is placed inside the curly braces.

 Since functions are treated as objects you can also pass them as a parameter to another function. For example, in the below code, we are passing the functions welcome() and goodbye() as parameters to the function greet()

1. function welcome(){console.log("Hello World");}
2. function goodbye(){console.log("See you later");}
3. function greet(choice){
4. choice();
5. }
6. greet(welcome);
7. greet(goodbye);

Since functions are treated as objects, you can also return them from a function. For example, in the below code, we are returning the function welcome() ( stored in variable hello ) from a function greet():

1. function greet(){
2. var hello=function welcome(){console.log("Hello World");}
3. return hello;
4. }
5. var retFunc=greet();
6. retFunc();

# ****Higher Order Functions****

Functions which can either accept other functions as parameters or return other functions as parameters are called as Higher Order Functions. Many in-built functions in JS are Higher Order Functions.

# First Class Citizen:

Any object which can be assigned, passed as a parameter and returned from a function is called a First Class Citizen in a programming language. Thus,**all** functions are First Class Citizens in JS.



We have seen that we can pass one function as a parameter to another function. Usually, for such purposes we can create a function without a name. Such functions without a name are called anonymous functions. For example:

1. function greet(choice){
2. choice();
3. }
4. greet(function(){ console.log("Hello World")});
5. *// Hello World*

Here we can see that the function which is being passed as a parameter does not have a name.

# Arrow functions

An arrow function is a concise way of writing a function. Arrow functions are anonymous functions as they don't have a name.

Syntax:

1. (parameter) => function body

For example:

1. function greet(choice){
2. choice();
3. }
4. greet(function(){ console.log("Hello World") }); *// Hello World*
5. greet(()=>{ console.log("Hello World") }); *// Hello World*

Let us look at take a deeper look at the syntax and different ways to write arrow functions in the coming pages.

elow are few scenario of arrow functions.

**Syntax 1**: Multi parameter, multi line code:

If code is in multiple lines, we need to have {}.

1. calculateCost = (ticketPrice, noOfPerson)=>{
2. noOfPerson= ticketPrice \* noOfPerson;
3. return noOfPerson;
4. }
5. console.log(calculateCost(500, 2));
6. *// 1000*

**Syntax 2**: No parameter, single line code:

If the code is single line, we don't need {}. The expression is evaluated and automatically returned.

1. trip = () => "Let's go to trip."
2. console.log(trip());
3. *// Let's go to trip.*

**Syntax 3**: One parameter, single line code:

If only one parameter, we don't need ().

1. trip = place => "Trip to " + place;
2. console.log(trip("Paris"));
3. *// Trip to Paris*

**Syntax 4:**One parameter, single line code:

if only one parameter, we can simply use '\_' and not use a variable name also.

1. trip = \_ => "Trip to " + \_;
2. console.log(trip("Paris"));
3. *// Trip to Paris*