**Javascript functions**

**Declare a function**

function functionName(){

// function body

}

The function name must be a valid javascript identifier. By convention, the function name are in camel case with verbs like getData(), fetchContents(), and isValid().

A function can accept zero, one or multiple parameters. In the case of multiple parameters, we need to use a comma to separate two parameters.

function say () {}, function square (params1) {}, function add (params1, params2) {}

**calling a function**

to use a function, we need to call it. Calling a function is also known as invoking a function. To call a function use its name with parentheses.

functionName(arguments)

Returning a value

Every function in javascript implicitly returns undefined unless we explicitly specify a return value. The functions immediately stops executing immediately when it reaches the return statement. Therefore, we can use the return statement without a value to exit the function prematurely.

Argument object

Inside a function, we can access a object called arguments that represents the named arguments of the function. The arguments object behave like array though it is not a instance of the array type.

If we can use square brackets [] to access the arguments: arguments [0] returns the first argument, arguments [1] returns the second one. Also we can access the length property of the arguments object to determine the number of arguments.

Function hoisting

Function hoisting is a mechanism that the javascript engine physically moves function declaration to the top of the code before executing them.

1. Arrow function

Arrow function is a concise way to write functions in javascript. It was introduced in ECMAScript 6 (ES6). Arrow functions are defined using the => and have a shorter syntax compared to traditional function expression.

Arrow function have some differences in behavior, especially with regard to this keyword. Arrow function have lexical scoping for the this keyword which means they inherit the value of this.

Arrow function with no argument

If a function doesn’t take any argument then we should use empty parentheses.

const showMessage = () =>

  console.log("I am a arrow function with zero argument");

Arrow function with one argument

If a arrow function has only one argument, we can omit the parentheses.

const showMessageOneArgument = message => console.log(message);

showMessageOneArgument("I am a arrow function with one argument");

Arrow function as expression

We can also dynamically create a function and use it as expression.

let age = 5;

let welcome = age(age < 18)

  ? () => console.log("baby")

  : () => console.log("adult")

Multiline arrow function

If a function body has multiple statements, we need to put them inside curly brackets {}.

Anonymous function

Anonymous function is a function in javascript which dont have any name. to define function we can simple write function without name

function () {}

Above function will show error. To resolve this error we have to wrap this anonymous function into (). This function denotes as function as expression and it return function object.

(function (){ console.log(“this is anonymous function”)})

If we have to call anonymous function, write () at the end of anonymous function.

(function (){console.log(“this is immediate invoked function”)})()

**Requirements for using arrow function**

1. **Function expression or arrow function syntax**

**Function expression**

**Arrow function**

1. **Assigning anonymous function as variable**

We can assign anonymous function as variable. We can reuse this function anywhere we want. We can also pass it as argument to other function.

Usecase for anonymous function

1. **Callbacks**

Anonymous functions are often used as callback functions, which are functions passed as arguments to other functions to be executed later, often in response to some event or asynchronous operation.

1. **Immediate invoked function expression**

Anonymous functions can be used to create IIFE, which are functions that are executed immediately after they are defined.

(function () {

})();

1. **Event handling**

Anonymous function are commonly used for event handling.

1. **Asynchronous operations**

Anonymous functions can be used as callbacks for asynchronous operations such as setTimout , fetch or promise.

1. **Iterating arrays**

Anonymous function can be used in array methods like map, filter and reduce for data transformations.

1. **dynamic function generation**

Immediately invoked function­

Immediate invoked function expression is a javascript function that runs as soons as it is defined. It is a design pattern which is also known as self-executing anonymous function and contain two major part

The first is the anonymous function with lexical scope enclosed within the Grouping operator ().

IIFE is often used to create a private scope for variables and functions to prevent polluting the global scope and to encapsulate functionality.

The second part creates the immediately invoked function () through which the javascript engine will directly interpret the function.