**JAVASCRIPT FUNCTIONS**

Functions are one of the fundamental reusable building blocks in JavaScript. A function in JavaScript is similar to a procedure—a set of statements that performs a task or calculates a value, but for a procedure to qualify as a function, it should take some input and return an output where there is some obvious relationship between the input and the output. To use a function, you must define it somewhere in the scope from which you wish to call it.

**Function Syntax and Working:**

A function definition is sometimes also termed a function declaration or function statement. Below are the rules for creating a function in JavaScript:

* Begin with the keyword function followed by,
* A user-defined function name (In the above example, name is sum)
* A list of parameters enclosed within parentheses and separated by commas (In the above example, parameters are x and y)
* A list of statements composing the body of the function enclosed within curly braces {} (In the above example, statement is “return x + y”).

**Return Statement:**

There are some situations when we want to return some values from a function after performing some operations. In such cases, we make use of the return. This is an optional statement. In the above function, “sum()” returns the sum of two as result.

**Function Parameters:**

Parameters are input passed to a function. In the above example, sum() takes two parameters, x and y.

**Calling Functions:**

After defining a function, the next step is to call them to make use of the function. We can call a function by using the function name separated by the value of parameters enclosed between the parenthesis.

**Types of Functions in JavaScript:**

1. Named Functions
2. **Anonymous Functions**
3. **Arrow Functions**:
4. **Immediately Invoked Function Expressions (IIFE)**:
5. **Constructor Functions**:
6. **Generator Functions**:

**I) Named Functions:**

* These have a specific name and are declared using the function keyword.
* Example:

function greet() {

console.log("Hello, World!");

}greet(); // Outputs: Hello, World!

**II) Anonymous Functions:**

* Functions without a name, often used as expressions or callbacks.
* Example:

const greet = function() {

console.log("Hello, World!");

};greet(); // Outputs: Hello, World!

**III) Arrow Functions:**

* A shorter syntax for writing functions introduced in ES6. They do not have their own this binding.
* Example:

const greet = () => {

console.log("Hello, World!");

};greet(); // Outputs: Hello, World!

**IV) Immediately Invoked Function Expressions (IIFE):**

* Functions that are executed as soon as they are defined.
* Example:

(function() {

console.log("IIFE executed!");

})();

**V) Constructor Functions:**

* Used to create objects, typically with the new keyword.
* Example:

function Person(name) {

this.name = name;

}const john = new Person("John");console.log(john.name); // Outputs: John

**VI) Generator Functions:**

* Introduced in ES6, these functions use the function\* syntax and can pause execution using the yield keyword.
* Example:

function\* generator() {

yield 1;

yield 2;

yield 3;

}const gen = generator();console.log(gen.next().value); // Outputs: 1