```
/*
A function is a block of code that executes only when it is called.
var; const;let

*/
const name = "Mary Doe";
console.log(name);
const greet = function () {
 //console.log("Good morning " + name); // concatination
 let name = "Peter Doe";
 let lastName = "Doe";
 console.log(`Good Morning, ${lastName}`);
};
greet();
// console.log(lastName);
// Function with parameters: Parameters are variables used to define a function
// Arguements are values assigned to the parameters
const square = function (n) {
 // let r = n**2
 console.log(n ** 2);
};
square(5);
const rect = function (l, w) {
 let area = l * w;
 console.log(area);
};
```

```
rect(4, 3);
// A return statement
const areaCircle = function (r) {
  let pi = 3.142;
  let area = pi * r ** 2;
  return area;
};
console.log(areaCircle(5));
// A function can return a function. This is called a closure. A lower function has access to the parameters
// of the higher function
const higher = function (x) {
  let lower = function () {
    return x ** 3;
  };
  return lower();
};
console.log(higher(3));
// A function returning more than one value at a time
const area1 = function (l, w) {
  let area = l * w;
  let perimeter = 2 * (l + w);
  //   return `The area of the rectangle is ${area}, and the perimeter is ${perimeter}`;
  //return [area, perimeter];// array
  return { area: area, perimeter: perimeter }; // array
};
```

```
console.log(area1(4, 3));
// A function with a default parameter
var details = function (name, email, address = "Gwarinpa Estate Abuja") {
  return `My name is ${name}, email me on ${email}. I live at ${address}`;
};
console.log(details("Joel Doe", "joeldoe@gmail.com"));
// Recursive function
//  factorial function 3! = 3x2x1; 4! = 4x3x2x1; 5x4x3x2x1
const fact = function (n) {
  if (n <= 1) {
    return 1;
  }
  return n * fact(n - 1);
};
console.log(fact(5));
/*
-b+-squareroot(b**2-4*a*c)/2*a
*/
// Arrow function
const hello = () => "Good Morning";
console.log(hello());
const sq = (n) => n ** 2;
console.log(sq(9));
```