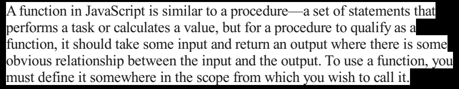


A function in JavaScript is similar to a procedure—a set of statements that performs a task or calculates a value, but for a procedure to qualify as a function, it should take some input and return an output where there is some obvious relationship between the input and the output. To use a function,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

FUNCTIONS

1. Normal functions

```
functions Test(arg1, agr2, ...){
    :
}
```

2. Arrow function

```
Const test2 = () => {
    :
}
```

thus $f(0)\ f(1)\ f(2).$

$0 * 1 * 2,\ 3$

$\text{fact}(n) = n! = n \times n-1 \times n-2 \ldots$

$f(0) = 1 \quad f(1) = 1 \quad f(2)$

$f(3) = f(0) * f(1) * f(2) * f(3$

$= 1 \times 1 \times 2 \times 3$

$= 2 \times 3$

$= 6$

$f(4) \quad f(3) \times f(n-1)$

$= 4 \times 6 = 24$

$f(4) = 4 \times f(3)$

$f(5) = 5 \times f(4)$

```
function fact(n) {
    return ~~1 or 0 n = =~~ 1
}
```

fibonacci

$(i-1)+(i-2)$

fib = 0 , 1 , 1 , 2 , 3 , 5 , 8 , 13 ......

fib(0) = 0

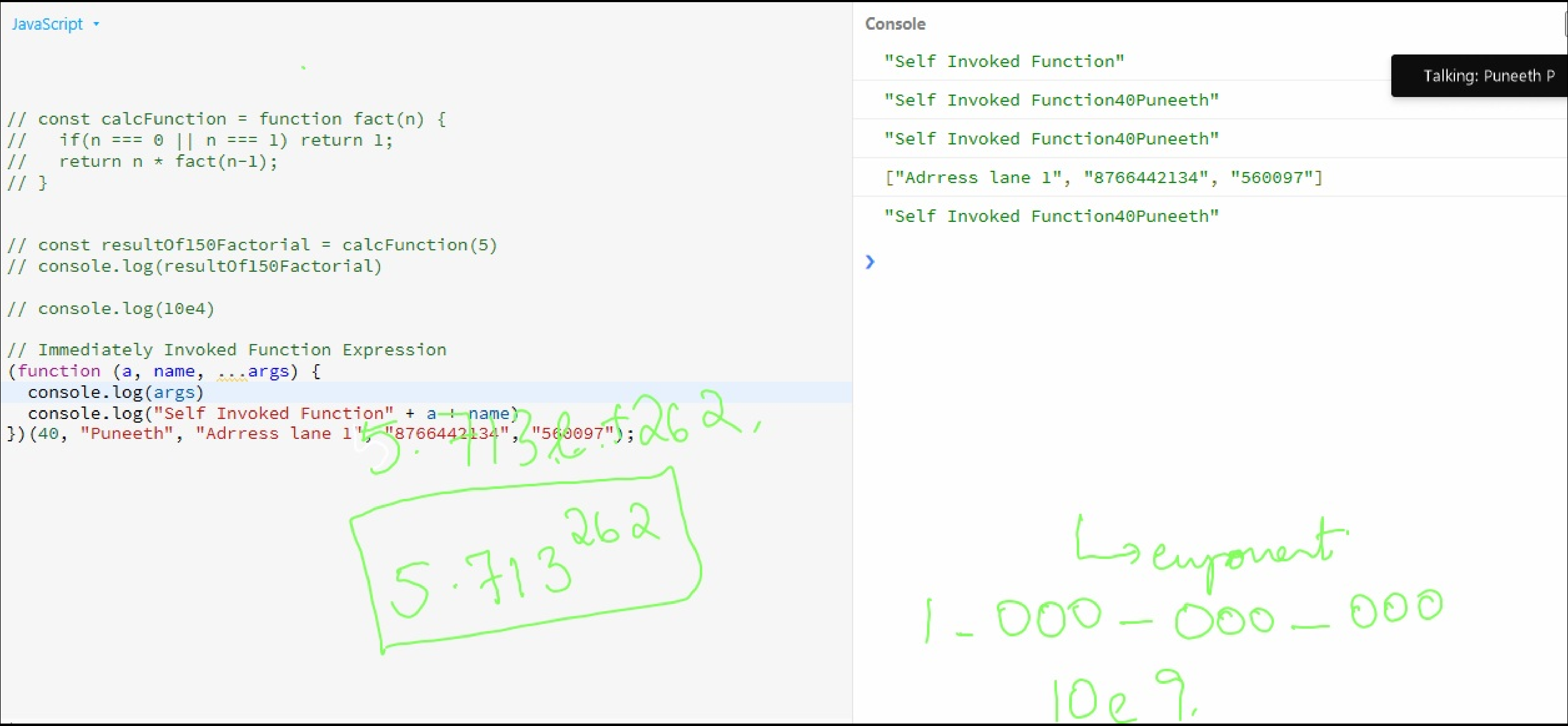
fib(1) = 1

fib(2) = 1

fib(3) = 2

fib(4) = 3

fib(5) = 5



```javascript
// const calcFunction = function fact(n) {
//   if(n === 0 || n === 1) return 1;
//   return n * fact(n-1);
// }


// const resultOf150Factorial = calcFunction(5)
// console.log(resultOf150Factorial)

// console.log(10e4)

// Immediately Invoked Function Expression
(function (a, name, ...args) {
  console.log(args)
  console.log("Self Invoked Function" + a + name)
})(40, "Puneeth", "Adrress lane 1", "8766442134", "560097");
```

```
"Self Invoked Function"
"Self Invoked Function40Puneeth"
"Self Invoked Function40Puneeth"
["Adrress lane 1", "8766442134", "560097"]
"Self Invoked Function40Puneeth"
›
```

5.713e+262,

$\boxed{5.713^{262}}$

↳ exponent

1_000_000_000

10e9.

Classes

OOP → Inheritance, Polymorism, Abstraction, Encapsulation.

```
Class Person {
  Constructor (nam, age) {
      this.name = nam
      this.age = age.
  }
  sayhello () {
  console.("this. name + ' age' + this.age)
  }
}
```

Abstraction →
(method whose implementation are not seen)

Inheritance:

```
Class Student extends Person {
  construt (name, age, school) {
      super (name, age)
      this.school = schoo
  }
  studiesIn ( ) { console.log("In school")
  }
}

Const brad = new Student ("Brad", 12, St.Joseph)

brad. studyIn() → Study in St. Joseph
brad. ~~age~~ sayHello() →
```

Polymorphism: method (arg1 ....).

↳

Polymorph. :

[1, 2, 3, 4]   (arg1, arg2...)
[1, 2]

```
+, *, -, /   Calculate (operation, ...args) {
      if (+)
      for (args :)
}
```

Encapsulations. → get & setters for class object values.

Encapsulation: