(OR)

6

math.floor(math.random() * 10) ; // Random no. b/w 1 to 10

* Functions :-

(function definition (telling us))

function funcName() {
    // do something
}
// for calling
funcName()

function with arguments

function printInfo(name, age){
    console.log(`${name}'s age is ${age}`);
}

printInfo("Shraddha", 23);
printInfo("Karan");

Karan age is undefined

return ;
{
return val;
}

Scope : determines accessibility of variables, objects etc.
→ fun^n → Block → lexical
→ Global scope

- **fun^n :** fun^n defined inside fun^n are not accessible from outside. (more specific than global (if you will) when fun^n considered) (if no thing declared inside fun^n then it can take from global.)

- **block :** variable declared inside {} block can't be accessed outside.
(var ↓ block applicable nhi)

```
{
  var a = 25
}
console.log(a)   → Error
```

- **Lexical :** Variable defined outside a fun^n can be accessible inside another fun^n defined after the variable declaration. But opposite is not true.

→
```
function outer(){
  let x = 5
  let y = 6
  function inner(){
    let a = 10;
    console.log(x)   → (5)
  }
  inner();
  console.log(a); //error
}
```

```
> outer()
// 5
> inner()
// error inner() is not defined
as inner() is scope fun^n scope bar
```

* **fun^n Expression :** //Nameless fun^n

```
const sum = function(a,b){
    return(a+b);
}
sum(2,3);
sum; // vala fun^n pura print hoga.
```

* **Higher order function.**
→ takes one or multiple fun^n as arguments
→ return a function

- function multigreet (func, count){
  for (let i = 1; i <= count; i++){
    func();
  }
}

let greet = function(){
  console.log("Hello");
};

multigreet (greet, 1000)

{ function greet(){ console.log("hello"); } }

Higher order function (Returns)

function oddevenfactory (request){
  if (request == "odd"){
    let odd = function(n){
      console.log(!(n%2 == 0));
    }
    return odd;
  } else if (request == "even"){
    let even = function(n){
      console.log(n%2 == 0);
    }
    return even
  } else {
    console.log("wrong request");
  }
}

let request = "odd";

let func = oddevenfactory (request)

func; // f(n){ console.log(!(n%2 == 0)) }

func(3); // True

// we can also change value of request & accordingly results changes.