# FRONT END – PART 2

## JAVASCRIPT : FUNCTIONS AND ARRAYS

### (Purpose of Return)

**1. What is the purpose of return statement in a function ?**

a) Returns the value and continues executing rest of the statements, if any

b) Returns the value and stops executing the function

c) Returns the value and stops the program

**d) Stops executing the function and returns the value**

**SOLUTION DESCRIPTION:**

The return statement stops the execution of the function and returns the specified value.

### (Function Definition)

**2. What is the correct way of defining a function in JavaScript?**

a) Identifier and Parentheses

b) Return type and Identifier

**c) Function keyword, Identifier and Parentheses**

d) Identifier and Return type

### (More Arguments Passed)

**3. What will the function 'test' return upon execution of these statements?**

```
function test(a, b) {
    console.log( a + b );
}
test( 2 , 3 , 4);
```

**a) 5**

b) Error (incorrect number of parameters )

c) 9

d) 7

### (Fewer Arguments)

**4. What will the function test return upon execution of these statements?**

```
function test(a, b, c) {
    console.log( a + b * c);
}
test(2,3);
```

a) 5

b) Error (incorrect no of parameters )

**c) NaN**

d) 0

### (Function Hoisting)

**5. In JavaScript function _____________ are hoisted**

a) Initializations

**b) Declarations**

### (What is the Output)

**6. What will be the output of following code?**

```
    x = 5;
    console.log(x);
    var x;
```

a) Uncaught ReferenceError: x is not defined

**b) 5**

c) NaN

d) 0

**(Find the Output)**

**7. What is the output of following code?**

```
    hoisted();

    function hoisted() {
        console.log('Hoisted');
    }
```

**a) "Hoisted"**

b) Uncaught ReferenceError: hoisted is not defined

c) ""

d) .*ƒ* hoisted() { console.log('Hoisted') }

**(Variable Hoisting)**

**8. What will be the output of following code?**

```
    function demo() {
        console.log(x);
        var x = 10;
    }

    demo();
```

a) NaN

b) 10

**c) undefined**

d) 0

**(Multiplication)**

**9. What will the following code print on console?**

```
    function multiply(a, b) {
        return a*b;
    };

    console.log(multiply);
```

a) multiply

b) a*b

**c) *ƒ* multiply(a, b) { return a*b; }**

d) undefined

e) None of the above

**SOLUTION DESCRIPTION:**

"multiply" is a variable that stores the address of a function which takes two input and return an output. When we have to execute the function we need to make a function call by adding () at the end of the variable name that stores the address of that function.

CODING NINJAS #skillforlife

If we simply console.log(function-name) then it will display the content that is stored at the address stored by variable.

In this case we are using console.log(multiply) so this will print function body that is stored at the address contained by variable "multiply".

If we use console.log(multiply(2,4)) then we are calling the function and output will be the data returned by the function and that is 8.

### (Global and Local Variable)

**10. What is the output of following code?**

```
var a = 10;

function test() {
    var a = 20;
}

test();
console.log(a);
```

**a) 10** b)20

### (Global Vs Local)

**11. What is the output of following code**

```
var a = 10;

function test() {
    var a = 20;
    console.log(a);
}

test();
```

a) 10 **b) 20**

### (Function Within Function)

**12. What is the expected output of following code**

```
function a() {
    console.log("Inside a");

    function b() {
        console.log("Inside b");
    }
}

a();
```

**a) Inside a**

b) Inside b

CODING NINJAS #skillforlife

c) *ƒ* a() { console.log("Inside a"); function b() { console.log("Inside b"); } }

d) a

**(Calling Inner Function)**

**13. What is the expected output of following code**

```
function a() {
    console.log("Inside a");

    function b() {
        console.log("Inside b");
    }
}

b();
```

a) Inside a

b) Inside a Inside b

c) *ƒ* a() { console.log("Inside a"); function b() { console.log("Inside b"); } }

d) b

**e) Uncaught ReferenceError: b is not defined**

CODING NINJAS #skillforlife

**(Function Call Inside Function)**

**14. What is the expected output of following code?**

```
var combinedString = "";
function a() {
    combinedString = "Inside a";

    function b() {
        combinedString = combinedString + " Inside b";
    }

    b();
}

a();
console.log(combinedString);
```

a) Inside a

**b) Inside a Inside b**

c) *ƒ* a() { console.log("Inside a"); function b() { console.log("Inside b"); } }

d) a

e) Uncaught ReferenceError: b is not defined

**(Function Expression)**

**15. What is the output of following code?**

```
var RectArea = function(width, height) {
    return width * height;
}

console.log(RectArea(5,4));
```

**a) 20** b) RectArea

c) ƒ (width, height) { return width * height; }

(Nested Variable Function)

**16. What is the expected output of following code?**

```
function a() {
    console.log("Inside a");

    function b() {
        console.log("Inside b");
    }
}

a()b();
```

a) Inside a

b) Inside a Inside b

c) ƒ a() { console.log("Inside a"); function b() { console.log("Inside b"); } }

d) ab

**e) Uncaught SyntaxError: Unexpected identifier**

(Nested Variable Function)

**17. Choose the correct syntax for a function expression.**

a) function function_Name(){ /*function body*/ }

**b) var variable_Name = function(){ /*function body*/ }**

**c) var variable_Name = function function_Name(){ /*function body*/ }**

d) None of the above.

**SOLUTION DESCRIPTION:**

The main difference between a function expression and a function declaration is the function name, which can be omitted in function expressions to create anonymous functions.

(Run Subtract)

**18. What is the expected output of following code?**

```
var add = function (a, b) {
    return a+b;
}
```

CODING NINJAS #skillforlife

```
var subtract = function (a, b) {
    return a-b;
}

var op = function (func) {
    var x = 2;
    var y = 3;
    return func(x, y);
}
console.log(op(subtract));
```

a) 5 **b) -1**

c) *f* (func) { var x = 2; var y = 3; return func(x, y); } d) *f* (a, b) { return a+b; }

**(Pass Function to Function)**

**19. What is the expected output of following code?**

```
var add = function(a, b) {
    return a+b;
}

var subtract = function(a, b) {
    return a-b;
}

var op = function (func) {
    var x = 2;
    var y = 3;
    return func(x, y);
}

console.log(op(add));
```

**a) 5** b) -1

c) *f* (func) { var x = 2; var y = 3; return func(x, y); } d) *f* (a, b) { return a+b; }

**(Callback Function)**

**20. Choose correct statements about the callback function.**

**a) If you want to execute a function right after the return of some other function, then callbacks can be used.**

**b) A function passed to another function as an argument is referred to as a callback function.**

c) A callback function can run before another function has finished.

d) None of these.

**(Length of Array)**

**21. How to find the length of an array?**

a) arr.length() b) arr.size()

**c) arr.length** d) arr.size

CODING NINJAS #skillforlife

**(Push in Array)**

**22. What is the output of the following code?**

```
var color= ["Orange", "Blue", "Green"];

color.push("Red");

console.log(color[0]+ " " +color[color.length-1]);
```

**a) Orange Red** b) Red Orange

c) Orange Green d) Green Red

**(Shift Array)**

**23. What is the output of the following code?**

```
var fruits = ["Red", "Orange", "Blue", "Green"];

console.log(fruits.shift());
```

**a) Red** b) Orange c) Blue d) Green

**(Heterogeneous Array)**

**24. Is the below given array allowed in Javascript?**

```
var myArray = [45, "Ninja"];
```

**a) Yes** b) No

**(Function of Splice)**

**25. What is the function of splice() in JavaScript?**

a) copies a given part of an array and returns that copied part as a new array.

b) it removes duplicate elements from the given array

c) it splits the string by separating it into two new array.

**d) slices a given part of an array and returns that sliced part as a new array.**

**(Array Splice)**

**26. What is the output of following code?**

```
var fruits = ['Apple', 'Orange', 'Kiwi', 'Strawberry'];

fruits.splice(4, 1, 'Banana');

console.log(fruits);
```

a) [ "Banana","Apple", "Orange", "Kiwi", "Strawberry"]

**b) ["Apple", "Orange", "Kiwi", "Strawberry", "Banana"]**

c) ["Apple", "Banana", "Orange", "Kiwi", "Strawberry"]

CODING NINJAS #skillforlife

d) ["Apple", "Orange", "Kiwi", "Banana", "Strawberry"]

**(Array Splice)**

**27. What is the output of following code?**

```
var fruits = ['Apple', 'Orange', 'Kiwi', 'Strawberry'];

fruits.splice(1, 0, 'Banana');

console.log(fruits);
```

a) [ "Banana","Apple", "Orange", "Kiwi", "Strawberry"]

b) ["Apple", "Orange", "Kiwi", "Strawberry", "Banana"]

**c) ["Apple", "Banana", "Orange", "Kiwi", "Strawberry"]**

d) ["Apple", "Orange", "Kiwi", "Banana", "Strawberry"]

**(Executing a function)**

**28. Which JavaScript method is used to call a function once for each array element?**

a) for() b) traverse() **c) forEach()** d) foreach()

**(Find the Output)**

**29. What will be the output of the following code?**

```
var sum = 0;
var arr = [1, 2, 3];
arr.forEach(getSum);
console.log(sum);

function getSum(ele) {
   sum += ele;
}
```

a) 3 **b) 6**

c) 2 d) None of the above.

**ASSIGNMENT**

**(Global and Local Variable)**

**30. What is the output of following code?**

```
var a = 10;

function test() {
    a = 20;
}

test();
console.log(a);
```

CODING NINJAS #skillforlife

a) 10

**b) 20**

**(Omitted Value in Array)**

**31. Consider the code given below -**

```
var arr = [ 1, , 3, 4 ];
```

**What is the observation made?**

**a) The omitted value takes “undefined”**

b) This results in an error

c) This results in an exception

d) The omitted value takes an integer value 0

**(Function in Function)**

**32. What will the following function return?**

```
function sqSum(a, b) {

    function square(x) {
        return x*x;
    }

    return square(a) + square(b);

}
```

a) Square of sum of a and b

**b) Sum of square of a and b**

c) Sum of a and b square

d) Error

**(Passing Fewer Arguments)**

**33. What will the function 'test' return upon execution of these statements?**

```
function test(a, b) {
    console.log( a + b );
}

test(2);
```

a) 2

b) Error (incorrect number of parameters )

**c) NaN**

d) 0

**(Hoisting in Javascript)**

**34. What will be the output of following code snippet?**

```
var a = 1;

function b() {
    a = 10;
    return;
    function a() { }
}
```

CODING NINJAS #skillforlife

```
b();
console.log(a);
```

a) 10

b) 1

c) ƒ b() { a = 10; return; function a() {} }

d) a

**SOLUTION DESCRIPTION:**

Expression function a() { } has created a local a that has a functional/local scope. This new a now gets hoisted to the top of its enclosing function b() with it's declaration and definition.

Therefore, the statement a = 10; is no longer changing the value of the global a which remains to be 1, but rather it is changing the local a from a function to an integer value of 10.

Since we are logging the global a, the output is 1.

**(Iterate Array)**

**35. What is the output of the following code?**

```
array = [ 60, 70, 20, 10, 40, 90 ];

const test = function(x) {
    return x > 5;
}

if(!array.every(test)) {
    console.log("statement 1");
}

else {
    console.log("statement 2");
}
```

CODING NINJAS #skillforlife

a) statement 1

b) statement 2

**(What is the output of following code?)**

**36. What is the output of following code?**

```
hoisted();
var hoisted = function() {
    console.log('bar');
};
```

a) bar

b) undefined

c) TypeError: Hoisted is not a function

**(For Each Word)**

**37. What is the output of following code?**

```
var words = ['one', 'two', 'three', 'four'];
```

```
words.forEach(function(word) {
    console.log(word);
    if (word === 'two') {
        words.shift();
    }
});
```

a) two three four

b) one three four

**c) one two four**

d) one two three

e) one two three four

**(Splice)**

**38. What is the output of following code snippet?**

```
var color = ['red', 'orange', 'blue', 'violet'];

var removed = color.splice(2);

console.log(color);
console.log(removed);
```

**a) ["red", "orange"] ["blue", "violet"]**

b) ["blue", "violet"] ["red", "orange"]

c) ["red", "violet"] ["orange", "blue"]

d) ["orange", "blue"] ["red", "violet"]

**(Find the Output – 2)**

**39. What will be the output of the following code:**

```
function a(){
    function b() {
        return 3;
    }

    return b();

    function b() {
        return 8;
   }
}

console.log(a());
```

a) 3

**b) 8**

c) Type Error: b is not a function

d) None of the above.

**SOLUTION DESCRIPTION:**

Both the b() functions are function declarations and will therefore be hoisted to the top of a() local scope.

However, the b() returning 8 will be hoisted after the one returning 3.

Therefore, the one returning 8 will be executed

CODING NINJAS #skillforlife

### (Rest Parameter)

**40. What will be the output of the following code?**

```
function fun(...input){
    var sum = 0;
    for(var i = 0; i < input.length; i++){
        sum += input[i];
    }
    return sum;
}
console.log(fun(1,2,3,4,5));
```

a) 3 b) 6 c) 10 **d) 15**

### (Default parameters)

**41. Consider the following code snippet:**

```
function f(a,b = 1) {
  console.log(a*b)
}
var x = // some hidden  value
f(5,x)
```

**What should be the value of x so that the output is 5?**

a) null **b) undefined** c) 0 d) ''

**SOLUTION DESCRIPTION:**

If we pass undefined to a default parameter, it will take the value of default parameter instead of undefined.

**********

CODING NINJAS #skillforlife