

## JavaScript Function MCQs

### 1. Function Call with Simple Return

```
function getMessage() {  
    return "Hello, World!";  
}
```

```
console.log(getMessage());
```

What will be the output of the code above?

**a) Hello, World!**

b) undefined

c) Hello

d) Error

### 2. Function Call with Parameters

```
function addNumbers(a, b) {  
    return a + b;  
}
```

```
console.log(addNumbers(3, 5));
```

What will be the output of the code above?

**a) 8**

b) 35

c) 53

d) Error

### 3. Function Return with Boolean Check

```
function isEven(num) {  
    return num % 2 === 0;  
}
```

```
console.log(isEven(10));  
console.log(isEven(7));
```

What will be the output of the code above?

a) **true and false**

b) false and true

c) true and true

d) false and false

#### 4. **Function with Multiple Return Statements**

```
function checkSign(num) {  
  if (num > 0) {  
    return "Positive";  
  } else if (num < 0) {  
    return "Negative";  
  } else {  
    return "Zero";  
  }  
}
```

```
console.log(checkSign(0));  
console.log(checkSign(5));
```

What will be the output of the code above?

a) **Zero and Positive**

b) Zero and Negative

c) Positive and Zero

d) Negative and Positive

#### 5. **Function with Return Inside Loop**

```
function findFirstEven(numbers) {  
  for (let i = 0; i < numbers.length; i++) {  
    if (numbers[i] % 2 === 0) {  
      return numbers[i];  
    }  
  }  
  return null;  
}
```

```
console.log(findFirstEven([1, 3, 5, 8]));  
console.log(findFirstEven([1, 3, 5]));
```

What will be the output of the code above?

a) **8 and null**

b) 1 and null

c) null and 5

d) 8 and 5

6. **Function Returning an Array**

```
function getOddNumbers(n) {  
  let odds = [];  
  for (let i = 1; i <= n; i++) {  
    if (i % 2 !== 0) {  
      odds.push(i);  
    }  
  }  
  return odds;  
}
```

```
console.log(getOddNumbers(5));  
console.log(getOddNumbers(10));
```

What will be the output of the code above?

a) **[1, 3, 5] and [1, 3, 5, 7, 9]**

b) [1, 3] and [1, 3, 5, 7]

c) [5, 3, 1] and [9, 7, 5, 3]

d) null and null

7. **Calling Function in Another Function**

```
function square(num) {  
  return num * num;  
}  
  
function sumOfSquares(a, b) {  
  return square(a) + square(b);  
}
```

```
console.log(sumOfSquares(3, 4));
```

What will be the output of the code above?

a) 25

b) 49

c) 7

d) 25 and 49

## 8. Function with Nested Conditions

```
function numberCategory(num) {  
  if (num > 0) {  
    if (num % 2 === 0) {  
      return "Positive Even";  
    } else {  
      return "Positive Odd";  
    }  
  } else if (num < 0) {  
    return "Negative";  
  } else {  
    return "Zero";  
  }  
}
```

```
console.log(numberCategory(2));  
console.log(numberCategory(-5));
```

What will be the output of the code above?

a) Positive Even and Negative

b) Positive Odd and Zero

c) Negative and Positive Even

d) Zero and Positive Even

## 9. Function with Ternary Operator

```
function isAdult(age) {  
  return age >= 18 ? "Adult" : "Minor";  
}
```

```
console.log(isAdult(20));  
console.log(isAdult(15));
```

What will be the output of the code above?

a) **Adult and Minor**

b) Minor and Adult

c) 20 and 15

d) true and false

10. **Calling a Function Twice**

```
function double(x) {  
  return x * 2;  
}
```

```
console.log(double(double(2)));
```

What will be the output of the code above?

a) **8**

b) 4

c) 16

d) Error

11. **Return Object from Function**

```
function createPerson(name, age) {  
  return {name: name, age: age};  
}
```

```
console.log(createPerson("Alice", 25));
```

What will be the output of the code above?

a) **{name: "Alice", age: 25}**

b) ["Alice", 25]

c) {"name": "Alice", "age": 25}

d) 25, Alice

12. **Calling Function with Default Parameters**

```
function greet(name = "Guest") {  
  return "Hello, " + name;  
}
```

```
console.log(greet());  
console.log(greet("John"));
```

What will be the output of the code above?

**a) Hello, Guest and Hello, John**

b) Hello, John and Hello, Guest

c) undefined and "John"

d) null and "Guest"

### 13. Returning Undefined Explicitly

```
function doNothing() {  
    return;  
}  
  
console.log(doNothing());
```

What will be the output of the code above?

**a) undefined**

b) null

c) 0

d) Error

### 14. Loop with Continue Statement

```
function getEvenNumbers(n) {  
    let evens = [];  
    for (let i = 1; i <= n; i++) {  
        if (i % 2 !== 0) {  
            continue;  
        }  
        evens.push(i);  
    }  
    return evens;  
}  
  
console.log(getEvenNumbers(6));
```

What will be the output of the code above?

a) **[2, 4, 6]**

b) [1, 3, 5]

c) [6, 4, 2]

d) [6]

#### 15. Function Inside an Object

```
let calculator = {  
  add: function(a, b) {  
    return a + b;  
  }  
};  
  
console.log(calculator.add(4, 6));
```

What will be the output of the code above?

a) **10**

b) 46

c) Error

d) undefined

#### 16. Return in For Loop

```
function getFirstPositive(numbers) {  
  for (let i = 0; i < numbers.length; i++) {  
    if (numbers[i] > 0) {  
      return numbers[i];  
    }  
  }  
  return -1;  
}
```

```
console.log(getFirstPositive([-3, -2, 1, 2]));
```

What will be the output of the code above?

a) **1**

b) -1

c) 2

d) 0

#### 17. Return Value from Arrow Function

```
let multiply = (a, b) => a * b;  
  
console.log(multiply(3, 4));
```

What will be the output of the code above?

a) 12

b) 34

c) Error

d) 7

#### 18. Function with No Return

```
function logMessage() {  
    console.log("Hello!");  
}  
  
logMessage();
```

What will be the output of the code above?

a) Hello!

b) undefined

c) null

d) Error

#### 19. Return in While Loop

```
function findFirstDivisibleBy3(numbers) {  
    let i = 0;  
    while (i < numbers.length) {  
        if (numbers[i] % 3 === 0) {  
            return numbers[i];  
        }  
        i++;  
    }  
    return -1;  
}
```



```
console.log(findFirstDivisibleBy3([1, 2, 9, 4]));
```

What will be the output of the code above?

a) **9**

b) -1

c) 4

d) 1

## 20. Recursive Function Call

```
function factorial(n) {  
  if (n === 0) {  
    return 1;  
  }  
  return n * factorial(n - 1);  
}
```

```
console.log(factorial(5));
```

What will be the output of the code above?

a) **120**

b) 25

c) 1

d) 5

## 21. Using Function Expression

```
let divide = function(a, b) {  
  return a / b;  
};
```

```
console.log(divide(10, 2));
```

What will be the output of the code above?

a) **5**

b) 10

c) 2

d) undefined

22. **Return Boolean Value**

```
function isNegative(num) {  
    return num < 0;  
}  
  
console.log(isNegative(-5));
```

What will be the output of the code above?

a) **true**

b) false

c) null

d) undefined

23. **Function Call Inside an Array**

```
let array = [function() { return 1 }, function() { return 2 }];  
  
console.log(array[1]());
```

What will be the output of the code above?

a) **2**

b) 1

c) undefined

d) Error

24. **Return Type of Function**

```
function square(num) {  
    return num * num;  
}  
  
console.log(typeof square(4));
```

What will be the output of the code above?

a) **number**

b) string

c) undefined

d) object

25. **Passing Function as Argument**

```
function runOperation(operation, x, y) {  
    return operation(x, y);  
}  
  
console.log(runOperation((a, b) => a - b, 7, 2));
```

What will be the output of the code above?

a) **5**

b) 9

c) 2

d) Error

26. **Function with Return Type and Conditional Statement**

```
function checkEvenOrOdd(number) {  
    if (number % 2 === 0) {  
        return "Even";  
    } else {  
        return "Odd";  
    }  
}  
  
console.log(checkEvenOrOdd(8));  
console.log(checkEvenOrOdd(7));
```

What will be the output of the code above?

a) **Even and Odd**

b) Odd and Even

c) Odd and Odd

d) Even and Even

## 27. Function with Loops

```
function sumNumbers(n) {  
  let sum = 0;  
  for (let i = 1; i <= n; i++) {  
    sum += i;  
  }  
  return sum;  
}  
  
console.log(sumNumbers(5));  
console.log(sumNumbers(3));
```

What will be the output of the code above?

- a) **15 and 6**
- b) 10 and 3
- c) 15 and 9
- d) 5 and 3

## 28. Nested Function with Return

```
function outerFunction(x) {  
  function innerFunction(y) {  
    return x * y;  
  }  
  return innerFunction;  
}  
  
const multiplyBy5 = outerFunction(5);  
console.log(multiplyBy5(3));  
console.log(multiplyBy5(4));
```

What will be the output of the code above?

- a) **15 and 20**
- b) 5 and 4
- c) 25 and 20
- d) 15 and 16

## 29. Function with While Loop

```
function countdown(n) {
  let result = "";
  while (n > 0) {
    result += n + " ";
    n--;
  }
  return result;
}

console.log(countdown(5));
console.log(countdown(3));
```

What will be the output of the code above?

- a) "5 4 3 2 1 " and "3 2 1 "
  - b) "5 4 3 2 " and "3 2 "
  - c) "5 4 3 " and "3 2 1 "
  - d) "5 " and "3 "
30. Return Type in Functions Using Do-While Loop

```
function printNumbers(n) {
  let i = 1;
  let result = "";
  do {
    result += i + " ";
    i++;
  } while (i <= n);
  return result;
}

console.log(printNumbers(4));
console.log(printNumbers(2));
```

What will be the output of the code above?

- a) "1 2 3 4 " and "1 2 "
- b) "4 3 2 1 " and "2 1 "
- c) "1 2 " and "1 2 3 4 "
- d) "4 " and "2 "