JS Quiz

Operators

1. What is the result of the following condition: `5 > 3`?
   ( ) false
   ( ) undefined
   (x) true

2. What is the output of `console.log("5" + 3)`?
   (x) "53"
   ( ) 8
   ( ) "8"
   ( ) 53

3. What is the output of `console.log(true || false)`?
   (x) true
   ( ) false
   ( ) error

4. What is the output of `console.log(4 < 5 && 5 > 3)`?
   ( ) 5>3
   (x) true
   ( ) false
   ( ) 4<5

5. What is the output of `console.log("7" === 7)`?
   ( ) true
   (x) false
   ( ) 7
   ( ) "7"

6. What is the output of `console.log(15 % 6)`?
   ( ) 2
   ( ) 0
   (x) 3
   ( ) true

7. What is the output of `console.log(Math.round(15.67))`?
   (x) 16
   ( ) 15
   ( ) 15.6
   ( ) 15.68

8. What is the output of `console.log("55" < 56 && 67 === "67" || 253 % 11 && !false)`?

( ) true
( ) "55"
( ) false
(x) 0

9. Which of the following is not a comparison operator in JavaScript?
   ( ) ==
   ( ) !==
   (x) **
   ( ) <=

10. What is the output of `console.log((15 * 2) / 3 + 8 === 18 || (10 % 3) ** 2 >= 4 && !(5 > 7))`?
    (x) true
    ( ) 18
    ( ) false
    ( ) error

Condtions

1. What is the result of the following condition: `21 % 6 && 14 % 2 || !false`?
   ( ) 3
   ( ) 0
   (x) true
   ( ) false

2. Which JavaScript conditional statement is used to execute a block of code only if a condition is true?
   ( ) for
   ( ) while
   ( ) switch
   (x) if

3. What is the syntax for an "if-else" statement in JavaScript?
   ( ) if (condition) {code block}
   (x) if (condition) {code block} else {code block}
   ( ) if (condition) then {code block} else {code block}
   ( ) if (condition) [code block] else [code block]

4. In JavaScript, what will be the result of the following code

```
const x = 10;

if (x < 5) {
  console.log("Less than 5");
} else {
  console.log("Greater than or equal to 5");
}
```

( ) Less than 5
( ) Greater than 5
(x) Greater than or equal to 5

5. In JavaScript, what will be the result of the following code

```
const day = "Thursday"
switch (day) {
    case "Monday":
    case "Tuesday":
    case "Wednesday":
    case "Thursday":
    case "Friday":
        console.log("It's a weekday.");
        break;
    case "Saturday":
    case "Sunday":
        console.log("It's a weekend day.");
        break;
    default:
        console.log("It's a invalid day.");
}
```

( ) It's a invalid day.
(x) It's a weekday.
( ) It's a weekend day.
( ) Error.

6. What is the purpose of the "else if" statement in JavaScript?
   ( ) To create a loop
   ( ) To define a function
   ( ) To declare a variable
   (x) To provide an alternative condition to check when the first condition is false

7. Which JavaScript conditional statement is used to select one of many code blocks to be executed?
   ( ) if
   ( ) for
   (x) switch
   ( ) while

8. What does the "switch" statement in JavaScript primarily check?
   ( ) Boolean values
   (x) The value of an expression against multiple case values
   ( ) The syntax of the code
   ( ) The existence of a variable

9. In a "switch" statement, what keyword is used to indicate the end of a case block?
   (x) break; or return;
   ( ) continue
   ( ) end
   ( ) finish

10. In JavaScript, what will be the result of the following code

```
const x = 5;
const y = 10;
if ((x + y) % x && y < 15) {
  console.log("Both conditions are true");
} else {
  console.log("At least one condition is false");
}
```

(x) At least one condition is false
( ) Error
( ) Both conditions are true

Loops

1. Which type of loop is commonly used when you know in advance how many times the loop should run?
   ( ) while loop
   (x) for loop
   ( ) do...while loop
   ( ) for of loop

2. What does the initialization expression in a for loop do?
   ( ) Defines the loop variable
   ( ) Specifies the loop termination condition
   (x) Initializes the loop variable
   ( ) Executes the loop code block

3. How do you terminate a while loop when a specific condition is met?
   (x) Use the break statement
   ( ) Use the continue statement
   ( ) Modify the loop variable

4. Which loop is suitable when the number of iterations is unknown, and the loop runs until a condition becomes false?
   ( ) for loop
   ( ) for in loop
   ( ) do...while loop
   (x) while loop

5. Where is the loop variable typically initialized in a for loop?
   ( ) Inside the loop code block
   ( ) After the loop code block
   (x) As a separate variable declaration before the loop block
   ( ) It's automatically created when the loop starts

6. In a while loop, where should you update the loop variable to avoid an infinite loop?
   ( ) Before the loop code block
   ( ) After the loop code block
   ( ) No need to update the loop variable
   (x) Inside the loop code block

7. What will be the output of the following code?

```
let number = 1;
let sum = 0;

while (number <= 10) {
  sum += number;
  number++;
}

console.log("Total: " + sum);
```

( ) Total: 10
( ) Total: 11
(x) Total: 55
( ) Total: 56

8. How many times will a number be printed, and in what order, when you run the following code?

```
let number = 10;

do {
  console.log(number);
  number--;
} while (number >= 1);
```

( ) The number will be printed 10 times in ascending order from 1 to 10.
( ) The number will be printed 9 times in descending order from 10 to 1.
(x) The number will be printed 10 times in descending order from 10 to 1.
( ) The number will be printed 9 times in ascending order from 1 to 10.

9. Regarding the following for loop, how many times will the loop be executed, and what value will it produce at the end?

```
let result = 0;

for (let i = 1; i <= 5; i++) {
  result *= i;
}

console.log("Result: " + result);
```

( ) The loop runs 5 times, and the result is 120.
( ) The loop runs 1 times, and the result is 0.
( ) The loop runs 6 times, and the result is 720.
(x) The loop runs 5 times, and the result is 0.

10. Which of the following contains a for loop in JavaScript that correctly calculates the sum of numbers from 1 to 5?

( ) `let total = 0; for (let i = 1; i < 5; i++) { total += i; } console.log("Total: " + total);`

( ) `let result = 0; for (let i = 5; i > 1; i--) { result += i; } console.log("Result: " + result);`
(x) `let sum = 0; for (let i = 1; i <= 5; i++) { sum += i; } console.log("Sum: " + sum);`
( ) `let sum = 1; for (let i = 1; i < 6; i++) { sum += i; } console.log("Sum: " + sum);`

Functions

1. How do you define a function in JavaScript?
   ( ) function myFunction() {}
   ( ) let myFunction = function() {};
   ( ) const myFunction = () => {};
   (x) All of the above

2. How do you call a JavaScript function named myFunction?
   (x) myFunction();

   ( ) run myFunction();
   ( ) execute myFunction();
   ( ) call myFunction();

3. What is the scope of a variable defined inside a JavaScript function?
   ( ) It can only be used within the page.
   (x) It is local to the function.
   ( ) It is global and can be used throughout the page.
   ( ) It can be used in other functions.

4. What does the following JavaScript function do? function double
   (x) { return x * 2;}
   ( ) Returns the square root of the input number.
   (x) Returns twice the input number.
   ( ) Returns half of the input number.

5. What is the issue with the following JavaScript code? function sayHello() { console.log("Hello, world!");}sayHello;
   ( ) There is no issue; the code will run correctly.
   ( ) The code will result in an error.
   (x) The "sayHello" function does nothing.

6. What is the key difference between a function declaration and a function expression in JavaScript?
   ( ) There is no difference; they are the same thing.
   (x) Function declarations are hoisted, while function expressions are not.
   ( ) Function expressions have access to the global scope, while function declarations do not.

7. In JavaScript, what will be the result of the following code:

```
const msg1 = greet();
const msg2 = greet2();

function greet() {
  return "Hello!"
}

const greet2 = function() {
  return "Hello!"
};
```

( ) Both msg1 and msg2 are equal to "Hello!"
( ) Error. JavaScript functions cannot be used before they are declared.
(x) msg1 equals "Hello," but msg2 throws an error.

8. In JavaScript, what will be the result of the following code:

```
function message(username) {
  console.log(`Hello ${username}`);
}

let sayHello = message;
sayHello("Fs Cohort 15");
```

( ) The code will result in an error.
(x) Hello FS Cohort 15
( ) The "sayHello" function does nothing.
( ) undefined

9. In JavaScript, what will be the result of the following code:

```
const multiply = (num1,num2) => {
     num1 * num2
}

const result = multiply(3,5);
console.log(result)
```

( ) 8
( ) 15
(x) undefined
( ) 125

10. In JavaScript, what will be the result of the following code:

```
const myFunc = (num1,num2) => {
     return num1 % num2 ? false : true
}

const isDivide = myFunc(252,9);
console.log(isDivide)
```

( ) false
( ) undefined
(x) true
( ) error

Strings

1. What will be the length of the following string? `let myStr = "Hello, FS Cohort 15!";`
   ( ) 17
   ( ) 19
   ( ) 18
   (x) 20

2. Which method is used to convert a string to all lowercase letters?

   (x) toLowerCase()

   ( ) toLower()
   ( ) lowerCase()
   ( ) convertToLower()

3. Which method is used to remove leading and trailing whitespace from a string?

   (x) trim()
   ( ) deleteWhitespace()
   ( ) removeWhitespace()
   ( ) strip()

4. In JavaScript, what will be the result of the following code:
   `console.log('ClarusWay'.endsWith('way'))`

   (x) false
   ( ) true

   ( ) Error

5. In JavaScript, what will be the result of the following code:
   `console.log('ClarusWay'.indexOf('Way'))`
   ( ) 8
   (x) 6
   ( ) true
   ( ) 5

6. In JavaScript, what will be the result of the following code:
   `console.log('hello'.replace('l', 'r'))`
   ( ) herro
   ( ) hello
   (x) herlo

7. What is the best description of the problem in the code below?

```
let myVar = "clarusway"
myVar[0]= "C"
```

( ) Once a string is created it can't be changed in place, since strings are mutable in JavaScript.
(x) Once a string is created it can't be changed in place, since strings are immutable in JavaScript.
( ) Changing characters of a string can only be done using the setCharAt() method.

8. In JavaScript, what will be the result of the following code:

```
let str = "Once a day";
let index = str.indexOf("a");
let index2 = str.lastIndexOf("a");
console.log(str.indexOf("a", index + 1) === index2);
```

(x) true
( ) false
( ) Error

9. How can we extract "Cohort 15" from the string `str = "Hello FS Cohort 15"`?
   ( ) str.slice(10,18)
   (x) str.slice(9)
   ( ) str.substring(9,17)
   ( ) str.split("C")

10. In JavaScript, what will be the result of the following code:

```
let myName = "Anthony Harold";
myName = myName.slice(8)
myName.toLowerCase()
console.log(myName)
```

( ) harold
( ) anthony harold
(x) Harold
( ) Anthony Harold

Arrays

1. How to access the third element("Lenovo") of the array `let arr = ["Apple", "Samsung", "Lenovo"]`?
   ( ) arr.indexOf(2)
   ( ) arr.indexOf(3)
   (x) arr[2]
   ( ) arr[3]

2. Which method removes the last element from an array and returns that element?

   (x) pop()
   ( ) shift()
   ( ) unshift()
   ( ) slice()

3. Which method adds a new item to the beginning of an array?

   ( ) push()
   (x) unshift()
   ( ) slice()
   ( ) pop()

4. Which method creates a new array based on a selection?

   ( ) shift()
   ( ) push()
   (x) slice()
   ( ) subarray()

5. Which method removes the first element from an array and returns that element?
   ( ) slice()
   (x) shift()
   ( ) unshift()
   ( ) pop()

6. Which method determines whether an array contains a certain value among its entries, returning true or false as appropriate?
   ( ) indexOf()
   (x) includes()
   ( ) slice()
   ( ) lastIndexOf()

7. In JavaScript, what will be the result of the following code:

```
const myArr = ["1" , 2, "Hello", "World!"]
myArr.push("Cohort 15!")
console.log(myArr)
```

( ) ["1" , 2, "Hello", "World!"]
( ) ["Cohort 15!" , "1" , 2, "Hello", "World!"]
( ) ["1" , 2, "Hello", "Cohort 15!"]
(x) ["1", 2, "Hello", "World!", "Cohort 15!"]

8. In JavaScript, what will be the result of the following code:

```
const myArr = ["1" , 2, "Hello", "World!"]
const newArr = myArr.reverse()
console.log(myArr)
```

( ) ["1" , 2, "Hello", "World!"]
(x) ["World!", "Hello", 2, "1"]
( ) Error
( ) ["1" , 2, "Hello"]

9. In JavaScript, what will be the result of the following code:

```
const myArr = [1, 2, 5, 6, 7, 3, 4]
const newArr = myArr
myArr.sort()
console.log(newArr)
```

( ) [1, 2, 5, 6, 7, 3, 4]
(x) [1, 2, 3, 4, 5, 6, 7]
( ) [7, 6, 5, 4, 3, 2, 1]
( ) Error

10. In JavaScript, what will be the result of the following code:

```
const myArr = ["1" , 2, "Hello", "World!"]
myArr.push("Cohort 15!")
if(myArr.includes("Cohort 15!")){
    for (let i = 0; i< myArr.length; i++){
        if(myArr[i] === "Cohort 15!"){
            myArr[i] = "FS Cohort 15!"
        }
    }
}
console.log(myArr)
```

( ) ["1" , 2, "Hello", "World!"]
( ) ["1", 2, "Hello", "World!", "Cohort 15!"]
( ) Error
(x) ["1", 2, "Hello", "World!", "FS Cohort 15!"]

Array Iteration Methods

1. Which method creates a new array, populated with the results of calling a provided function on every element in the calling array?

   ( ) filter()
   (x) map()
   ( ) reduce()
   ( ) forEach()

2. Which method takes in an array, performs a function that you provide on each element in that array, and returns one single value?

   ( ) filter()
   (x) reduce()
   ( ) forEach()
   ( ) map()

3. Which method performs a function that you provide on each element in that array, and returns only those elements that pass the test implemented by that function?

   ( ) some()
   ( ) every()
   ( ) map()
   (x) filter()

4. Which method tests whether all elements in the array pass the test implemented by the provided function, returning a boolean (true/false) value?

   ( ) some()
   (x) every()
   ( ) includes()
   ( ) filter()

5. What does the forEach() method do?
   (x) Iterates over the array and executes a function once for each element.
   ( ) Splits the array into multiple smaller arrays.
   ( ) Sorts the elements of the array in ascending order.
   ( ) Reverses the order of the elements in the array.

6. The some method in JavaScript returns true if at least one element in the array satisfies the provided testing function. What does it return if no elements pass the test?
   ( ) undefined
   (x) false
   ( ) -1
   ( ) It throws an error.

7. The find method in JavaScript returns the first element in an array that satisfies the provided testing function. What does it return if no elements pass the test?
   (x) undefined
   ( ) false
   ( ) -1
   ( ) null

8. In JavaScript, what will be the result of the following code:

```
const myArr = [1, 2, 5, 6, 7, 3, 4]
const newArr = myArr
myArr.sort()
const numArr = newArr.forEach((item,index) => item)
console.log(numArr)
```

( ) [1, 2, 5, 6, 7, 3, 4]
( ) [1, 2, 3, 4, 5, 6, 7]

( ) [7, 6, 5, 4, 3, 2, 1]
(x) undefined

9. Select the option that accomplishes the given task in the provided code block.

```
const points = [67, 77, 34, 97, 55, 42];
// Passing score is 65
// Calculate the average of those who passed the course.
```

( ) points.map(item=> item >= 65).reduce((acc,item)=> acc + item) / points.filter(item=> item>= 65).length
(x) points.filter(item=> item >= 65).reduce((acc,item)=> acc + item) / points.filter(item=> item>= 65).length
( ) points.some(item=> item >= 65).reduce((acc,item)=> acc + item) / points.filter(item=> item>= 65).length
( ) points.find(item=> item >= 65).reduce((acc,item)=> acc + item) / points.filter(item=> item>= 65).length

10. The oddArr array is an array containing the squares of the odd numbers in the numbers array. Please select the code block that correctly produces this result from the options below.

```
const numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
const oddArr = ?
```

( ) numbers.find(item => item % 2).map(item=> item ** 2)
( ) numbers.filter(item => item % 2).forEach(item=> item ** 2)
(x) numbers.filter(item => item % 2).map(item=> item ** 2)
( ) numbers.forEach(item=> item ** 2).filter(item => item % 2)

Objects

1. Which of the following statements is true about JavaScript objects?
   ( ) They can only store primitive data types.
   (x) They can store both primitive and complex data types.
   ( ) They can store only numbers.
   ( ) They cannot store functions.

2. How do you add a new property to an existing JavaScript object?

   (x) By using dot notation or square brackets.
   ( ) By using the addProperty method.
   ( ) By using the setProperty method.
   ( ) You cannot add properties to an existing object.

3. Which of the following is an example of a JavaScript object literal?

   (x) {name: "John", age: 30}
   ( ) new Object("name: John, age: 30")
   ( ) createObject({name: "John", age: 30})
   ( ) [name: "John", age: 30]

4. Which of the following is incorrect for creating an empty JavaScript object?
   ( ) const emptyObject = new Object();
   ( ) const emptyObject = {};
   (x) const emptyObject = createObject()
   ( ) function EmptyObject() {}; const emptyObject = new EmptyObject()

5. Which of these objects have 3 properties?
   (x) const myObj = {course:"Clarusway", path:"Fullstack", module:"Frontend"}
   ( ) const myObj = {"Clarusway", "Fullstack", "Frontend"}
   ( ) const myObj = {course:"Clarusway", path:"Fullstack"}
   ( ) const myObj = {myInfo:{course:"Clarusway", path:"Fullstack", module:"Frontend"}}

6. In JavaScript, what will be the result of the following code:

```
const myObj = {
  course: "Clarus"
}
myObj.course = "Clarusway";
console.log(myObj["course"]);
```

( ) "Clarus"
( ) Error
(x) "Clarusway"
( ) {course:"Clarusway"}

7. What does JavaScript return to you if you try to access a object's property that doesn't exist?
   ( ) null
   ( ) -1
   ( ) Error
   (x) undefined

8. In JavaScript, what will be the result of the following code:

```
const pen = {};
pen.ink = "blue";
if(pen["ink"]) {
    console.log(`Color is ${pen["ink"]}`)
}else {
    console.log("Color is undefined!")
}
```

(x) "Color is blue"
( ) "Color is undefined!"
( ) Error

9. In JavaScript, what will be the result of the following code:

```
const weekdays = {
    0:"Sunday",
    1:"Monday",
    2:"Tuesday",
    3:"Wednesday",
    4:"Thursday",
    5:"Friday",
    6:"Saturday",
}

const day = 0
const isWeekDaysOrEnds = weekdays[day]==="Sunday" ||
weekdays[day]==="Saturday" ? "weekends" : "weekdays";
console.log(isWeekDaysOrEnds);
```

( ) "weekends"
(x) "weekdays"
( ) Error
( ) undefined

10. Select the code block that finds the product with the highest price.

```
const products = [
  {"model": "HP Pavilion","processor": "Intel Core i7","ram":
"16GB","storage": "512GB SSD","price": 999},
  {"model": "Dell XPS","processor": "Intel Core i5","ram": "8GB","storage":
"256GB SSD","price": 899},
  {"model": "MacBook Air","processor": "Apple M1","ram": "8GB","storage":
"256GB SSD","price": 1099},
  {"model": "Asus ROG Strix","processor": "AMD Ryzen 9","ram":
"32GB","storage": "1TB SSD","price": 1499},
  {"model": "Lenovo ThinkPad","processor": "Intel Core i7","ram":
"16GB","storage": "512GB SSD","price": 1199}
];
```

( ) products.reduce((max, product) => (product.price < max.price) ? product : max, products[0]);
(x) products.sort((a,b)=> b.price - a.price)[0]

( ) products.map(product => product.price).sort((a,b)=>a-b)
( ) products.filter(product=> product.price > 1000)[0]

New Generation Operators

1. Which operator can be used to combine two or more arrays in JavaScript?
   (x) Spread operator (...)
   ( ) Nullish Coalescing operator (??)
   ( ) Logical Nullish Assignment operator (??=)
   ( ) Optional Chaining operator (?.)

2. What is destructuring in JavaScript?
   (x) A way to unpack values from arrays, or properties from objects, into distinct variables.
   ( ) A way to break or destroy your code intentionally.
   ( ) A way to create complex nested structures.
   ( ) A way to encrypt data.

3. Which of the following is NOT a valid use of destructuring in JavaScript?
   ( ) Destructuring an array.
   (x) Destructuring a string.
   ( ) Destructuring an object.
   ( ) Destructuring a function.

4. How do you destructure an array in JavaScript?
   ( ) const (first, second, third) = [1, 2, 3];
   ( ) const {first, second, third} = [1, 2, 3];
   (x) const [first, second, third] = [1, 2, 3];
   ( ) const <first, second, third> = [1, 2, 3];

5. What does the rest operator (...) do when used in destructuring? For example; `const [first, ...rest] = [1, 2, 3, 4];`
   (x) It gathers the remaining elements of an array into a new array.
   ( ) It scatters elements into multiple arrays.
   ( ) It combines the values of an array into a single string.
   ( ) It converts an array into an object.

6. What happens if you try to destructure a non-existent property from an object in JavaScript?
   ( ) It raises a runtime error.
   ( ) It assigns a default value if provided.
   (x) It assigns undefined to the variable.
   ( ) It assigns null to the variable.

7. In JavaScript, what will be the result of the following code:

```
const [x, y] = [3, 7];
const { a, b } = { a: 10, b: 20 };
const result = x + a;
console.log(result);
```

(x) 13
( ) 30
( ) Error
( ) 10

8. In JavaScript, what will be the result of the following code:

```
function multiply(factor, ...numbers) {
  return numbers.map(num => num * factor);
}

const result = multiply(2, 3, 4, 5);
console.log(result);
```

( ) [4,6,8,10]
(x) [6,8,10]
( ) [2,3,4,5]
( ) Error

9. In JavaScript, what will be the result of the following code:

```
function greet({ name, age }) {
  return `Hello, ${name}! You are ${age} years old.`;
}

const person = { name: 'Anthony', age: 30 };
const greeting = greet(person);
console.log(greeting);
```

(x) 'Hello, Anthony! You are 30 years old.'
( ) 'Hello, undefined! You are undefined years old.'
( ) Error
( ) undefined

10. In JavaScript, what will be the result of the following code:

```
let object = { 'myName' : 'Anthony', 'myAge' : 30 };
let objectTwo = { 'myAge' : 675 };
let newObject = { ...object, ...objectTwo };
console.log(newObject);
```

( ) { 'myName' : 'Anthony', 'myAge' : 30 }
( ) { 'myName' : 'Anthony', 'myAge' : 30, 'myAge' : 675 }
(x) { 'myName' : 'Anthony', 'myAge' : 675 }
( ) { 'myAge' : 675 }
