50 JavaScript Questions

With answer key

Open Education Project at BCIT

50 JavaScript Questions

- -- Function --
- -- Loop --
- -- OOP questions --
- -- Timeout and interval questions --

-- Function --

1. What will be printed in the console below?

```
function remainder(a, b) {
    return a % b;
  }
  console.log(remainder(5, 2) % remainder(7, 3));
```

a) 0 -- CORRECT

b) 1

c) 2

d) undefined

2. What will be the result of the loop?

```
while (true) {
  let x = 10;
}
```

a) It will create a variable with a value of 10, and then stop

b) The loop will not run because there is no comparison operator in the condition

c) The browser will crash due to an infinite loop -- CORRECT

3. What will the console print?

```
const arr = ["a", "b", "c", "d"];
for (let i = arr.length - 1; i > 1; i--) {
  console.log(arr[i - 1]);
}
```

a) d, c

b) c, b -- CORRECT

c) b, a

d) a, undefined

4. What will the console print?

```
for (const i = 0; i < 4; ++i) {
  console.log(++i);
}
```

a) 0 1 2 3

b) 1 2 3 4

c) 2 3 4 5

d) Syntax error -- CORRECT

5. What will the console print?

```
function isDivisibleBy(a, b) {
  return a % b === 0;
}
let x = isDivisibleBy(15, 4);
console.log(x);
```

a) 3

b) -1

c) true

d) false -- CORRECT

6. Consider the object, array, and loop below:

```
const obj = {
  name: "John",
  age: 20
};
const info = ["name", "age"];
for (let i = 0; i < info.length; i++) {
```

How would you access the current property of the object inside this loop?

```
}
```

a) obj[info[i]] -- CORRECT

b) obj[info]

c) info[i]

d) obj[i]

7. What will the console print?

```
const foo = function(a, b) {
  a += b;
  return a * (a - b);
};
const num = foo(2, 3);
console.log(num);
```

a) 10 -- CORRECT

b) 25

c) -10

d) -25

8. What will the console print?

```
const arr = [1, 2, 3, 4, 5];
const obj = { 1: arr.length };
console.log(obj[1]);
```

a) undefined

b) Not defined

c) 5 -- CORRECT

d) 4

9. What will the console print?

```
const x = null;
if (!x) {
  x = 10;
}
console.log(x);
```

a) null

b) 10

c) undefined

d) Syntax Error -- CORRECT

10. How would you access the "5" in this multidimensional array?

```
const arr = [[1, 2, 3], [4, { a: 5 }, 6]];
```

a) arr[1[a].a]

b) arr[1][1.a]

c) arr[1][1].a -- CORRECT

d) arr[1[1]].a

e) The "5" cannot be accessed from this array

11. What will the console print?

```
console.log(3.14 == Math.PI);
```

a) true

b) false -- CORRECT

12. What will the console print?

```
for (let i = 4; i > 0; i--) {
  console.log(i % 2);
}
```

a) 2,1,2,1

b) 0,1,0,1 -- CORRECT

c) 4,2,4,2

d) 2,4,2,4

13. What will the console print?

```
let foo = function(a, b) {
  return (a * b).toString().slice(0, 2);
};
console.log(foo(40, 10));
```

a) 400

b) 40 -- CORRECT

c) 4

d) 0

14. What will the console print?

```
let a = "foo";
let b = "bar";
let c = 10;
let d = 20;
console.log(a + b + c + d);
```

a) foobar1020 -- CORRECT

b) foobar30

e) Syntax error

15. What will the console print?

```
const arr = ["a", "b", "c", "d"];
for (let i = arr.length - 1; i > 0; i--) {
  arr[i] = arr[arr.length - i - 1];
}
console.log(arr);
```

a) ["d", "c", "b", "a"]

b) ["a", "b", "c", "d"]

c) ["d", "c", "c", "d"]

d) ["a", "b", "b", "a"] -- CORRECT

e) Syntax error

16. What will the console print?

```
const obj = {
  a: "foo",
  b: "bar",
  c: function() {
    console.log(this);
  }
};
obj.c();
```

a) > Window

b) > function

c) > Object

d) > { a: "foo", b: "bar", c: f } -- CORRECT

17. What is the correct syntax for declaring an IIFE?

a) -- CORRECT

```
(function() {
    Your code here
})();
```

b)

```
function = (){
    Your code here
}();
```

c)

```
let foo() = function(){
    Your code here
};
```

d)

```
function foo(){
    Your code here
}();
```

18. What will the console print?

```
function getIndexInArray(arr, item) {
  return arr.indexOf(item);
}
let arr = [1, 2, 3, 4, 5, 6];
let result = getIndexInArray(arr, 89);
console.log(result);
```

a) -1

b) 0

c) 88

d) 89

e) Syntax error

19. Which of the following objects is valid JSON format?

a)

```
{
  foo: "bar",
  bar: 1
};
```

b)

```
{
    foo: "bar",
    bar: "1"
}
```

c) -- CORRECT

```
{
    "foo": "bar",
    "bar": 1
}
```

d)

None of these are valid JSON format

20. What will the console print?

```
const foo = function(obj, arr, i) {
  obj[arr[i]] = arr[i];
  return obj;
};
const oldObject = {
  a: 1,
  b: 2,
  c: 3
};
const arr = ["a", "b", "c"];
const newObject = foo(oldObject, arr, 1);
console.log(newObject);
```

a) > { a: "a", b: 2, c: 3 }

b) > { a: 1, b: "b", c: 3 } -- CORRECT

c) > { a: 1, b: "c", c: 3 }

d) Uncaught TypeError: Assignment to constant variable

-- Loop --

21. How many console logs will be printed in the following loop?

```
for (let i = 0; i < 3; i++) {
    for (let j = 0; j < 3; j++) {
        console.log("Hello");
    }
}
```

a) 0

b) 3

c) 6

d) 9 -- CORRECT

EXPLANATION: Two nested loops run three times each.
3 x 3 = 9

22. What will the console print?

```
let num = 0;
for (let i = 0; i <= 2; i++) {
    num += 2;
}
console.log(num);
```

a) 0

b) 2

c) 4

d) 6 -- CORRECT

EXPLANATION: The loop runs three times due to the
'<=' operator.
It adds two each iteration.

23. How many console.logs will be printed?

```
for (let i = 0; i <= 3; i++) {
    if (i === 2) {
        break;
    }
    console.log("Hello");
}
```

a) 0

b) 1

c) 2 -- CORRECT

d) 3

EXPLANATION: It prints once when "i" is 0, and once more when "i" is 1.
It breaks when "i" is 2, before the third console.log

24. Which of the following loops will iterate backwards from 5 -> 0 (including 5 and 0)?

a) -- CORRECT

```
for (let i = 5; i >= 0; i--) {
    console.log(i);
}
```

b)

```
for (let i = 5; i > 0; i--) {
    console.log(i);
}
```

c)

```
for (let i = 6; i > 0; i--) {
    console.log(i);
}
```

d) Both A and C will work

EXPLANATION: A starts at 5 and includes 0. B starts at 5 but does not include 0.

C starts at 6, and does not include 0.

25. What will the console print?

```
let arr = [1, 2, 3, 4, 5];
let total = 0;
for (let i = 0; i < arr.length; i++) {
    total += arr[i];
    if (i > 2) {
        break;
    }
}
console.log(total);
```

a) 0

b) 6

c) 10 -- CORRECT

d) 15

EXPLANATION:
First iteration: "total" gets 1 added. Current value is 1
Second iteration: "total" gets 2 added. Current value is 3
Third iteration: "total" gets 3 added. Current value is 6. "i" is 2 at this point
Fourth iteration: "total" gets 4 added (BEFORE the break). "i" is 3 at this point, loop ends.

26. Fill in the blank: loop through every item in the array

```
let arr = [1, 2, 3, 4, 5, 6];
for (let i = 0; ________; i++) {
    console.log(arr[i]);
}
```

ANSWER: i < arr.length;

EXPLANATION: loops over the entire array. Also acceptable: i <= arr.length - 1;

27. Fill in the blank: escape the loop when "i" is equal to 5

```
for (let i = 0; i < 10; i++) {
  if (i === 5) {
    __________;
  }
}
```

ANSWER: break;

EXPLANATION: "break" escapes the loop.

28. Fill in the blank: Write a condition to only loop through the 3 middle items in the array, in this case: "b", "c", "d"

```
const arr = ["a", "b", "c", "d", "e"];
for (_____________) {
  console.log(arr[i]);
}
```

ANSWER: let i = 1; i <= 3; i++

EXPLANATION: "b" has an index of 1, and it continues until index 3. Also acceptable: i < 4

29. Fill in the blank: Complete the for loop, to iterate over every second item in the array. It should print: "a", "c", "e", "g"

```
const arr = ["a", "b", "c", "d", "e", "f", "g"];
for (let i = 0; i < arr.length; __________) {
   console.log(arr[i]);
}
```

ANSWER: i += 2

EXPLANATION: i += 2 will increment 2 each time. i = i + 2 also works.

30. Fill in the blank: Edit the while loop condition so that the loop prints: 1, 2, 3

```
let i = 0;
while (______) {
   i++;
   console.log(i);
}
```

ANSWER: i < 3

EXPLANATION: the increment is before the console.log, so the loop needs the include
the case when i == 2. i < 3, or i <= 2 both work.

31. Which number would be retrieved when the following code is run?

```
let arr = ["a", "b", "c", "d", "e", "f"];
console.log(arr[arr.length / 2]);
```

a) "c"

b) "d" -- CORRECT

c) "f"

d) Error - index is out of bounds

EXPLANATION: arr.length / 2 will have a value of 3. "d" is at index 3.

32. Which of the following will retrieve the middle index (the letter "c", in this case)?

```
let arr = ["a", "b", "c", "d", "e"];
```

a) -- CORRECT

```
console.log(arr[arr.length - 3]);
```

b)

```
console.log(arr[arr.length - 2]);
```

c)

```
console.log(arr["c"]);
```

d) None of the above

EXPLANATION: arr.length - 3 will be index 2, which is the letter "c"

33. What will the console print?

```
let arr = [];
arr[1] = 1234;
console.log(arr[0]);
```

a) 1234

b) 1233

c) arr[0] does not have a value -- CORRECT

EXPLANATION: If you insert a value into an index above 0, on an empty array,

It will fill the previous indices with 'undefined'

34. How would you access the name "Dave" in the "names" array?

```
let names = ["John", "Smith", ["Ted", "Dave"]];
```

a) names[3][1]

b) names[2[1]]

c) names[2][1] -- CORRECT

d) names["Dave"]

e) You cannot nest an array inside of an array

EXPLANATION: names[2] will point to the array, and adding the [1] will point to "Dave"

35. What will the console print?

```
let arr = ["a", "b", "c", "d", "e"];
arr.length = 4;
console.log(arr);
```

a) ["a", "b", "c", "d", "e"]

b) ["a", "b", "c", "d", undefined]

c) ["a", "b", "c", "d"] -- CORRECT

d) Error - cannot reduce length of array "arr"

EXPLANATION: Assigning arr.length to a lower number will truncate all of the items after.

36. Fill in the blank: Edit the function so that it always cuts the last two elements from the array

```
function removeTwo(arr) {
    Note: you do not need a return statement
   ____________________;
}
```

ANSWER: arr.length = arr.length - 2; --OR-- arr.length -= 2;

EXPLANATION: reducing the array length by 2 will remove the last two elements

37. Fill in the blank: Write the selector to grab the SECOND LAST item in an array called "arr"

```
arr[____________];
```

ANSWER: arr.length - 2

EXPLANATION: arr.length-1 is the last element. arr.length-2 is the second last.

38. Fill in the blank: dynamically populate the array with

the values: 1, 2, 3, 4

Note: the array should ONLY contain [1,2,3,4]

```
const arr = [];
for (let i = 1; i < 5; i++) {
    ____________;
}
```

ANSWER: arr[i - 1] = i

EXPLANATION: The for-loop starts "i" at 1. you need to use arr[i-1]

to point to 0 (first item in the array), and assign it with the value of "i"

39. Fill in the blank: how would you multiply the current number in the array by the previous number in the array? Note: the final array value should be: [1, 2, 6, 24]

```
const arr = [1, 2, 3, 4];
for (let i = 1; i < arr.length; i++) {
    arr[i] = ____________;
}
```

ANSWER: arr[i] * arr[i - 1]

EXPLANATION: The previous element can be references

```
with arr[i-1].
 After that it's just multiplication

40.
/* Fill in the blank: The following is the fibonacci
sequence,
this means that the current number in the array should
be equal
to the previous TWO numbers, added together. Note:
the final output
should be: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34] */
const arr = [0, 1];
for (let i = 2; i < 10; i++) {
    arr[i] = ___________;
}
console.log(arr);
 ANSWER: arr[i - 1] + arr[i - 2]

 EXPLANATION: Previous element is arr[i-1]. The one
before that is arr[i-2].
 You just need to add those two values.
```

-- OOP questions --

41. What will the console print?

```
function Person(firstName, lastName) {
    this.firstName = firstName;
    this.lastName = lastName;
  }
  let person1 = Person("Elon", "Musk");
  console.log(typeof person1);
```

a) object

b) function

c) Person

d) undefined -- CORRECT

42. What will the console print?

```
  let obj = { a: 1, b: 2, c: 3 };
  let d = "a";
  console.log(obj[d] + obj.a + obj["b"]);
```

a) 1

b) 3

c) 4 -- CORRECT

d) undefined

e) NaN

43. Which of the following will call the "move" method and print "the car is moving"?

```
  let car = {
    doors: 4,
    move: function() {
      console.log("the car is moving");
    }
  };
```

a) car.move;

b) car.move();

c) car["move"]();

d) car.move(); and car["move"](); are both valid -- CORRECT

e) This is impossible unless you instantiate a "car" first

44. What will the console print?

```
  function Car(make, model) {
    this.make = make;
    this.model = model;
  }
  let car1 = new Car("Toyota", "Corolla");
  car1.doors = 4;
  console.log(Car.doors);
```

a) 4

b) undefined -- CORRECT

c) Car

d) { make: "Toyota", model: "Corolla" }

e) { make: "Toyota", model: "Corolla", doors: 4 }

45. Will this code snippet add a "div" element to the DOM?

```
document.body.appendChild("div");
```

a) Yes, it will create an empty <div></div> element with nothing inside

b) No, but it will work if you use document.appendChild("div"); instead

c) No, document.body.appendChild() must receive an element as its argument, not a string -- CORRECT

46. Given <input id="age" type="text" />, how would you get the age and multiply it by 2?

a) document.getElementById("age").value * 2

b) document.getElementById("age") * 2

a) parseInt(document.getElementById("age")).value * 2

b) parseInt(document.getElementById("age").value) * 2 -- CORRECT

47. Which of the following will select all of the "div" elements on the document?

a) document.body.getElementsByTagName("div"); -- CORRECT

b) document.getElementById("div");

c) document.body.div;

d) document.div;

48. Given <ul id="ordered-list"></ul>, what will the following code snippet do?

```
let li = document.createElement("li");
document.getElementById("ordered-list").appendChild(li);
```

a) It will throw an error because you can only call appendChild on the document.body object

b) It will add an <li> element to the <ul> element -- CORRECT

c) It will not add the <li> element to the DOM because it has no content in it

d) It will throw an error because the getElementById selector is invalid

-- Timeout and interval questions --

49. What will this code snippet do?

```
function foo() {
    console.log("foo");
    foo();
  }
  setTimeout(function() {
    foo();
  }, 1000);
```

a) It will print "foo" once and then exit

b) It will print "foo" once every second

c) It will print "foo" repeatedly due to an infinite loop unrelated to the timeout -- CORRECT

d) It will not print anything

50. How much time will pass before "foo" is printed?

```
  setTimeout(function() {
    console.log("foo");
  }, 5000);
```

a) Exactly 5000 milliseconds, guaranteed

b) At least 5000 milliseconds -- CORRECT

c) At most 5000 milliseconds