JavaScript Near to Engine

- solve problems to feel the JS engine

1. JavaScript is a -

    a. Multi-Threaded Language.
    b. Single Threaded Language.

2. JavaScript is a -

    a. Compiled Language
    b. Interpreted Language
    c. JIT Compiled Language

3. How many call stack(s) JavaScript do have?

    -

4. Is JavaScript synchronous or asynchronous language?

    -

5. Is JavaScript blocking or non-blocking in nature?

    -

6. How many types of execution context does JavaScript create to run code?

    -

7. Who controls the flow of execution context?

    -

8. What is the first execution context that gets pushed to call stack?

    -

9. Memory Heap is responsible for holding –

    a. Primitive type value.
    b. Non-Primitive type value.

10. Everything happens in JavaScript inside –

    a. Call Stack
    b. Memory Heap
    c. Web Api

11. Which is the first phase of execution context?

   a. Code Execution Phase
   b. Memory Creation Phase
   c. Garbage collection Phase

12. Global Execution Context runs –

   a. Local code
   b. Global code

13. In Memory creation phase JavaScript assigns Null to any variable declared with Var –

   a. True
   b. False

14. In memory creation phase, for any function declaration JavaScript keeps –

   a. Undefined in the memory
   b. Null in the memory
   c. Function Body in the memory

15. Any global variable declared with Var or Function, binds as a property of –

   a. Window Object
   b. this
   c. Script

16. The memory space of let & const is named as –

   a. global
   b. closure
   c. script

17. Scope of variable declared with var is –

   a. Local Scope
   b. Block Scope
   c. Function Scope

18. What is the scope of let & const -

   -

19. Hoisting happens in JavaScript due to -

   a. Code Execution Phase
   b. Memory Creation Phase

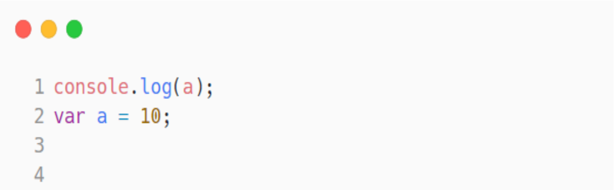
20. Temporal Dead Zone is Created for –

   a. var
   b. let
   c. const

21. In the temporal dead zone JavaScript throws error –

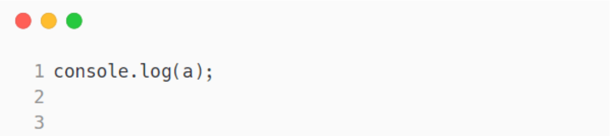
   a. True
   b. False

22. What is the output of the code snippet?



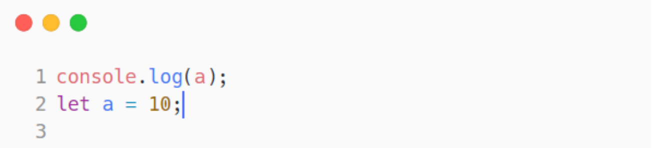
```
console.log(a);
var a = 10;
```

23. What is the output of the code snippet?



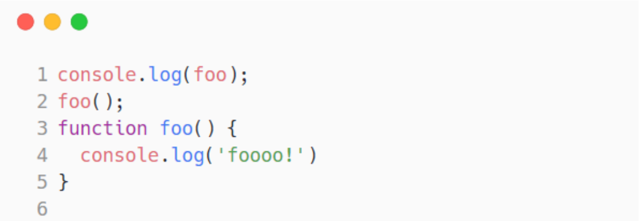
```
console.log(a);
```

24. What is the output of the code snippet?



```
console.log(a);
let a = 10;
```

25. what is the output of the code snippet?



```
console.log(foo);
foo();
function foo() {
  console.log('foooo!')
}
```

26. What is parameter & argument of function? Write an example.

-

27. What is Higher Order Function?

-

28. JavaScript behaves asynchronously with the help of –

a. Web Api
b. Event Loop
a. Micro Task Queue
b. Macro Task Queue

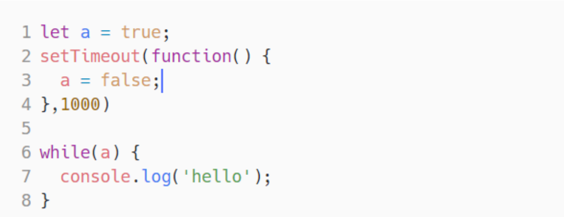
29. What is the value of this in global space?

-

30. What is the value of this in a global function?

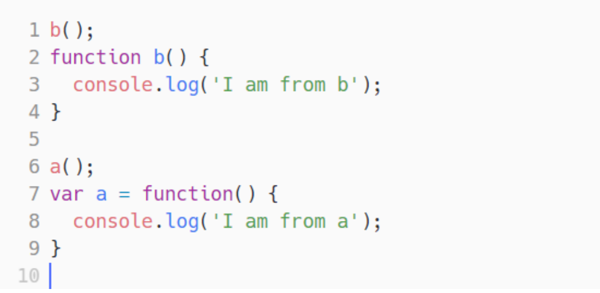
-

31. What will happen if we run this code below?



```
let a = true;
setTimeout(function() {
  a = false;
},1000)

while(a) {
  console.log('hello');
}
```

32. What is the output of the code?

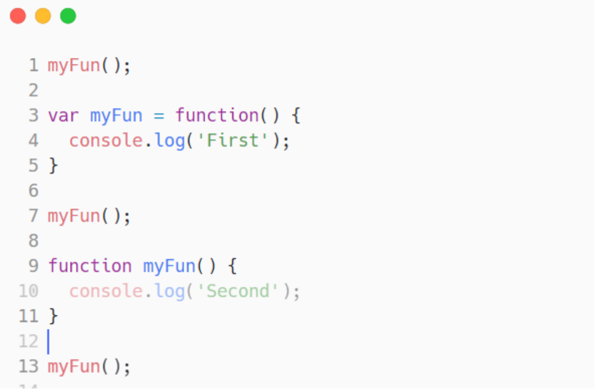


```
b();
function b() {
  console.log('I am from b');
}

a();
var a = function() {
  console.log('I am from a');
}

```

33. We can assign a function expression in a variable in JavaScript –

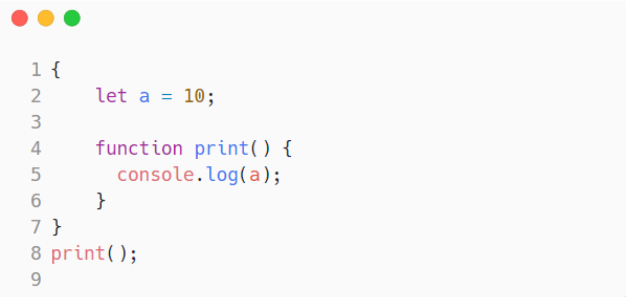
a. True
b. False

34. Write the output of the code below –



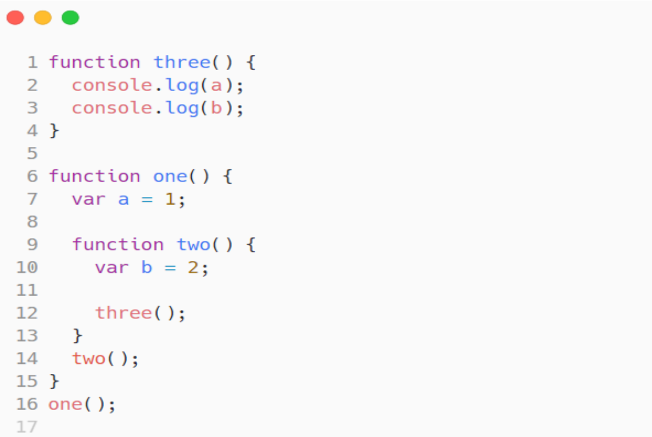
```
myFun();

var myFun = function() {
  console.log('First');
}

myFun();

function myFun() {
  console.log('Second');
}

myFun();
```

35. What is the output of the code below?



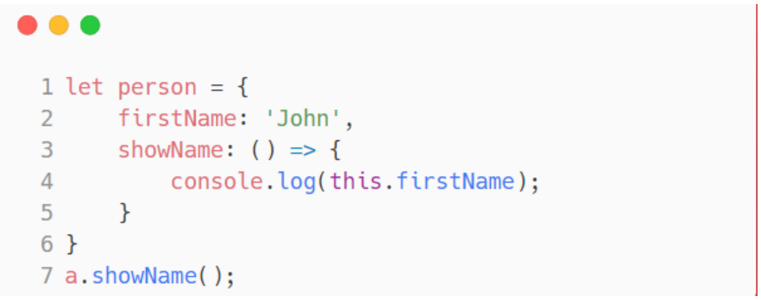
```
{
    let a = 10;

    function print() {
      console.log(a);
    }
}
print();

```

36. What is the output of the code below?



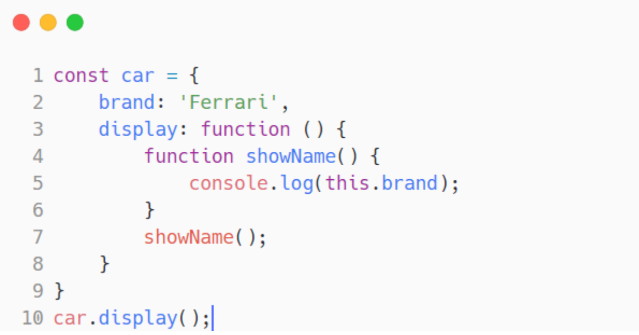
```
function three() {
  console.log(a);
  console.log(b);
}

function one() {
  var a = 1;

  function two() {
    var b = 2;

    three();
  }
  two();
}
one();

```

37. What is the output of the code below?



```
let person = {
    firstName: 'John',
    showName: () => {
        console.log(this.firstName);
    }
}
a.showName();
```

38. What is the output of the code below?



```
const car = {
    brand: 'Ferrari',
    display: function () {
        function showName() {
            console.log(this.brand);
        }
        showName();
    }
}
car.display();
```

39. How many types of task queue are available in JavaScript engine?

-

40. Which task queue has the lowest priority?

-

41. What is the purpose of Event Loop?

-

42. Among callback of setTimeout and callback of Promise, which has the highest priority to be pushed in the call stack?

-

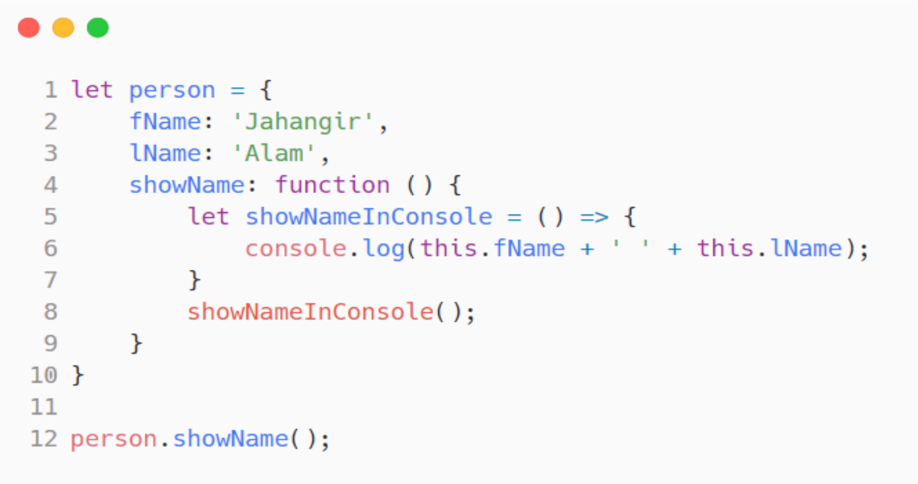
43. We can set/change the context of method by using

a. call
b. apply
a. bind

44. Which one returns a function?

a. call
a. bind
b. apply

45. What is the output of the code below?



```
let person = {
    fName: 'Jahangir',
    lName: 'Alam',
    showName: function () {
        let showNameInConsole = () => {
            console.log(this.fName + ' ' + this.lName);
        }
        showNameInConsole();
    }
}

person.showName();
```

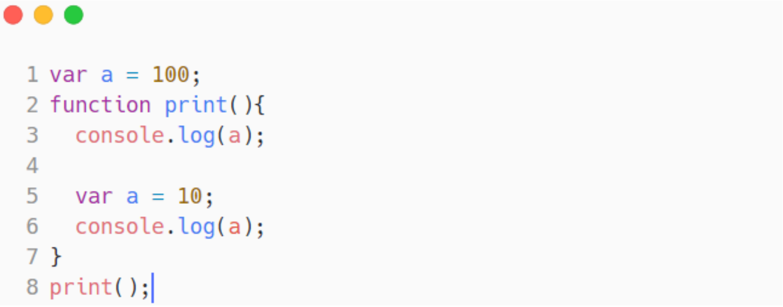
46. What is the value of a variable which is declared but not initialized?

-

47. What is the difference between function declaration and function expression?

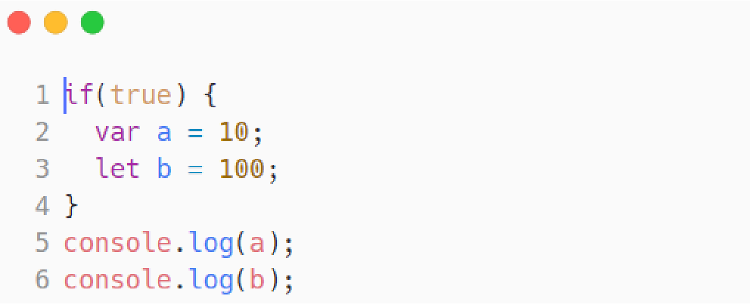
-

48. What is the output of the following code snippet?



```
var a = 100;
function print(){
  console.log(a);

  var a = 10;
  console.log(a);
}
print();
```

49. What is the output of the following code snippet?



```
if(true) {
  var a = 10;
  let b = 100;
}
console.log(a);
console.log(b);
```

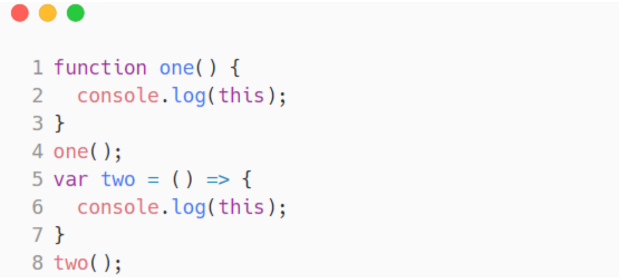
50. Whenever a function is invoked a new execution context is created. Is it true or false?

-

51. Whenever a function execution is done, it is popped out from the stack and all the local variables are garbage collected –

a. True
b. False

52. What is the output of the following code?



```
function one() {
  console.log(this);
}
one();
var two = () => {
  console.log(this);
}
two();
```

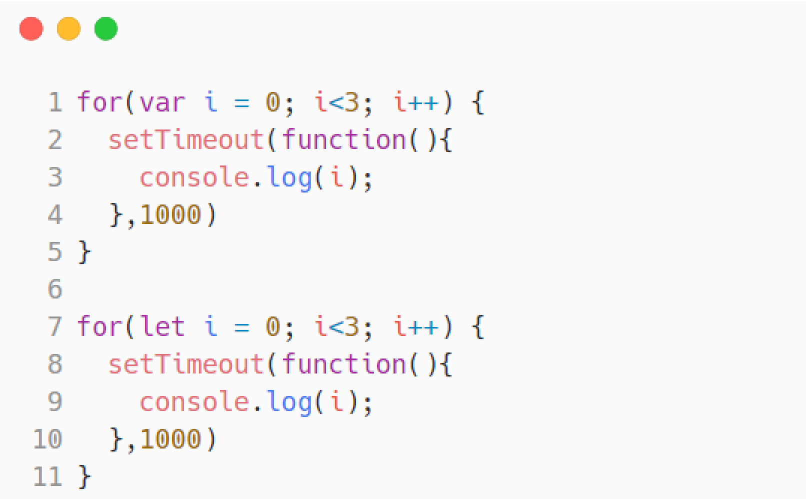
53. Arrow function inherits this from outer lexical environment –

a. True
a. False

54. Constructor function needs to be invoked with the keyword –

a. this
a. new
b. call

55. What is the output of the following code?

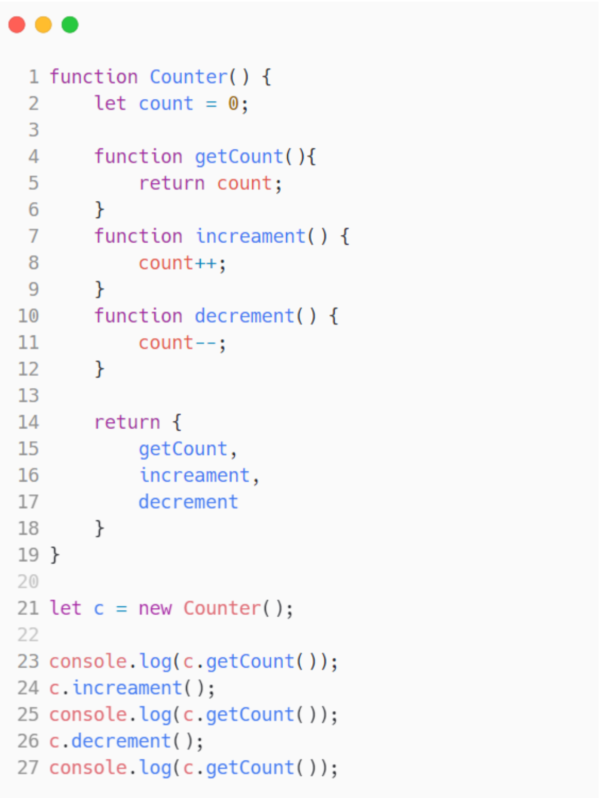


```
for(var i = 0; i<3; i++) {
  setTimeout(function(){
    console.log(i);
  },1000)
}

for(let i = 0; i<3; i++) {
  setTimeout(function(){
    console.log(i);
  },1000)
}
```

56. Is there any guarantee that callback of setTimeout will always run after given time passed to setTimeout?

a. Yes
a. No

57. What is the output of the code below?



```
function Counter() {
    let count = 0;

    function getCount(){
        return count;
    }
    function increament() {
        count++;
    }
    function decrement() {
        count--;
    }

    return {
        getCount,
        increament,
        decrement
    }
}

let c = new Counter();

console.log(c.getCount());
c.increament();
console.log(c.getCount());
c.decrement();
console.log(c.getCount());
```

58. For creating a Closure there must be a parent child relationship between two function –

a. True
b. False

59. In JavaScript, we can achieve inheritance by the concept of –

a. Closure
a. Hoisting
b. Prototype