Q1. var dateobj =
var dateToday = new Date('October 15, 1996 05:35:32:77 GMT+11:00');
How to fetch the millisecond according to universal time from a given Date object?

ANSWER :

```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to display milliseconds of a specific date-time, according to UTC.</p>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

<script>
function myFunction() {
  var d = new Date("October 15, 1996 05:35:32:77");
  var n = d.getUTCMilliseconds();
  document.getElementById("demo").innerHTML = n;
}
</script>

</body>
</html>
```

OUTPUT :

```
     77
```

Q2.
Write a JavaScript program to display the reading status (i.e. display book name, author
name and reading status) of the following books.

```
var library = [
{
author: 'Bill Gates',
title: 'The Road Ahead',
readingStatus: true
},
{
author: 'Steve Jobs',
title: 'Walter Isaacson',
readingStatus: true
},
{
author: 'Suzanne Collins',
title: 'Mockingjay: The Final Book of The Hunger
Games',
readingStatus: false
}];
```

ANSWER:

```
var library = [

   { title: 'Bill Gates' ,   author: 'The Road Ahead',
readingStatus: true },

   { title: 'Steve Jobs' ,   author: 'Walter Isaacson' ,
readingStatus: true },

   { title: 'Mockingjay: The Final Book of The Hunger
Games' , author: 'Suzanne Collins' , readingStatus:
false }];
```

Q3. What will be the output of the following code?

```
var Employee =
{
company: 'Rohit'
}
var Emp1 = Object.create(employee);
delete Emp1.company Console.log(emp1.company);
```

ANSWER :

var Employee = { company: 'Rohit' } var Emp1 = Object.create(employee); delete Emp1.company Console.log(emp1.company);

Q4. Consider the two functions below. Will they both return the same thing? Why or why not?

```
function foo1()
{
return {
bar: "hello"
};
}
function foo2()
return
{
bar: "hello"
};
}
```

ANSWER :

OUTPUT:

Here, we see that foo1 returns an object and foo2 returns undefined without any error being thrown. What is the reason behind this?

In JavaScript, semicolon technically behaves like an optional. In the first function, we used opening curly braces after the return statement within the same line while in the second function we used opening curly braces in the new line after the return statement. This is the main fact here. When we used opening curly braces in the new line in Foo2 function, a semicolon is inserted automatically after the return statement. So what will it return without showing undefined? No error is thrown since the remainder of the code is perfectly valid, even though it doesn't ever get invoked or do anything.
It is because of stylistic preference in JavaScript.

Q5.

```
var arr = [2, 56, 78, 34, 65];
var new_arr = arr.map(function(num) {
return num / 2;
});
print(new_arr);
```

What will be the output for the above code?

ANSWER :

```
var arr = [2, 56, 78, 34, 65];
var new_arr = arr.map(function(num) {
  return num / 2;
});
print(new_arr);
```

OUTPUT:

```
[1, 28, 39, 17, 32.5]
```

In this example the function map() produces an array containing numbers obtained by dividing the numbers in original array by 2

2021-08-18

Q6.
const set = new Set(['Beethoven', 'Mozart', 'Chopin', 'Chopin'])
How to delete 'Beethoven' from set

ANSWER :

set.remove('Beethoven');

Q9.
const gimli =
{ name: "Gimli",
race: "dwarf",
weapon: "axe",
greet: function() { return `Hi, my name is ${this.name}!
`; }, };
How to access and print greet function from the above object code?

ANSWER :

gimli.greet();

2021-08-18

Q7. Create a nested array object called operatingSystem and add below key and values
Name - Ubuntu , version - 18.4 , license - open source

ANSWER :

Object.entries()
Object.entries() creates a nested array of the key/value pairs of an object.

```
// Initialize an object
const operatingSystem = {
   name: 'Ubuntu',
   version: 18.04,
   license: 'Open Source'
};

// Get the object key/value pairs
const entries = Object.entries(operatingSystem);

console.log(entries);
```

OUTPUT:

```
[
   ["name", "Ubuntu"]
   ["version", 18.04]
   ["license", "Open Source"]
]
```

Q8.
const name = { firstName: 'Philip',
lastName: 'Fry' };
const details = {
job: 'Delivery Boy',
employer: 'Planet Express'
};
How to copy values from one object to another in above mentioned example , merge and
print them?

ANSWER :

```
// Initialize an object
const name = {
   firstName: 'Philip',
   lastName: 'Fry'
};

// Initialize another object
const details = {
   job: 'Delivery Boy',
   employer: 'Planet Express'
};

// Merge the object with the spread operator
const character = {...name, ...details}

console.log(character);
```

OUTPUT :

```
{firstName: "Philip", lastName: "Fry", job: "Delivery
Boy", employer: "Planet Express"}
```

This spread syntax in object literals is also known as shallow-cloning

Q10.
Consider below as an example of constructor function where we want name and level to be referred to a function itself, is below example correct or not? Please explain?

```
function Hero(name, level)
{
name = name;
level = level;
}
```

ANSWER :

```
this.name=name
this.level=level
```

2021-08-18