Question 1

Que signifie ECMAScript ? *Réponses* :

- C'est un langage de programmation
- C'est une base de données
- C'est un service Web
- C'est un ensemble de normes

Question 2

Quels sont les types primitifs en JS ? *Réponses* :

- boolean null undefined number bigint
- boolean null undefined number bigint string symbole
- Object number string
- Object boolean null undefined number bigint string symbole

Question 3

Quel est le nom de l'attribut ou méthode permettant de calculer la taille d'un Map ? *Réponses* :

```
const myMap = new Map([1, 2, 3])
```

- len
- length
- size
- size()

Question 4

Qu'affiche le code suivant ?

```
const ensemble = new Set([1, 2, 3, 4, 5, 5]);
console.log(ensemble);
```

Réponses :

- [1, 2, 3, 4, 5, 5]
- {1, 2, 3, 4, 5, 5}
- {1, 2, 3, 4, 5}
- {1, 2, 3, 4, 5, 5}

Question 5

Qu'affiche le code suivant ?

```
let b = 1;

function baz(){
    let [b, c] = [1, 2];
    console.log(b, c);
    function foo(){
        console.log(c)
    }
}
console.log(b);
baz();
```

Réponses :

- 1 2
  2
  1 2 2

?

- 1 1

Question 6

Qu'affiche le code suivant ?

```
const t1 = [1, 2];
const t2 = t1;
t1.push(3);
console.log(t1);
```

Réponses :

- Une SyntaxError
- `[1,2,3]`
- `[1,2]`
- `{1,2,3}`

Question 7

Qu'affiche le code suivant ?

```
for (let j = 0; j < 10; j++) {}
console.log(j);
```

Réponses :

- `ReferenceError`
- `SyntaxError`
- 10
- null

Question 8

Qu'affiche le code suivant ?

```
let x, y ;
[x,,y] = [10, 20, 11, 111];
console.log(x, y);
```

Réponses :

- `SyntaxError`
- `ReferenceError`
- 10 11
- 10 111

Question 9

Qu'affiche le code suivant ?

```
let x, y ;
const [,,,...rest] = [10, 20, 11, [ 111, 120, 7 ]];
console.log(rest.shift());
```

Réponses :

- `SyntaxError`
- `ReferenceError`
- 7
- `[111,120, 7]`

Question 10

Que vaut le tableau t1 ?

```
let t1 = [1, 2 ];
lr       [ ...t1 ];
t        );
co.     .log(t2);
```

Réponses :

- ☐ `[[1, 2]]`
- ☐ `[3]`
- ☐ `[1,2,3]`
- ☐ `[1, 2]`

Question 11

Que vaut le tableau t1 ?

```
let t1 = [ 1, 2 ];
let t2 = t1.map(x => x);
t2.push(3);
console.log(t2);
```

Réponses :

- ☐ `[[1, 2]]`
- ☐ `[3]`
- ☐ `[1,2,3]`
- ☐ `[1, 2]`

Question 12

Qu'affiche le code suivant ?

```
const phrase = '8790:bonjour le monde:8987:7777:Hello World:9007';
const sentence = phrase.split(':').map(w => w.slice(w.length -1)).filter(Number);
console.log(sentence);
```

Réponses :

- ☐ `["7", "7", "7"]`
- ☐ `[7, 7, 7]`
- ☐ `[]`
- ☐ `["8790", "8987", "7777", "9007"]`

Question 13

Qu'affiche le code suivant ?

```
const phrase = '8790:bonjour le monde:8987:7777:Hello World:9007';
const sentence = phrase.split(':').map(w => w.slice(w.length -1));
console.log(sentence);
```

Réponses :

- ☐ `["7", "7", "7"]`
- ☐ `[7, 7, 7]`
- ☐ `[]`
- ☐ `["0", "e", "7", "7", "d", "7"]`

Question 14

Comment récupérer le name et le nom de la soeur en utilisant de la desctructuration ?

```
const st = {
    name: "Alan",
    family: {
      mother: "Isa",
      father: "Philippe",
      sister: "Sylvie"
    },
    age: 35
  };
```

Ré s:

- ☐ `const { name, family : { sister }} = st;`

- `const { name, sister } = st;`
- `const { name, family.sister } = st;`
- `const { name, sister : { family }} = st;`

Question 15

Qu'affiche le code suivant ?

```
class Rectangle {
    constructor(w ,h){
        this._w = w;
        this._h = h;
    }

    get w (){
        return this._w;
    }
}

class Square extends Rectangle{

    constructor(w){
        this._w = w;
        super(w);
    }
    // ...
}

console.log( (new Square(3)).w );
```

Réponses :

- `ReferenceError`
- 3
- 9
- `SyntaxError`

Question 16

Qu'affiche le code suivant ?

```
class Model{
    constructor(tableName){
        this._tableName = tableName;
    }

    set tableName(tableName){
        this._tableName = tableName;
    }

    get tableName(){
        return this._tableName;
    }
}

const t1 = new Model('posts');
const t2 = new Model('POSTS');
console.log(t1.tableName);
```

Réponses :

- `ReferenceError`
- POSTS
- posts
- `SyntaxError`

Question 17

Qu'affiche le code suivant ?

```
const st = {
    id : 5,
    name : "Alan",
    age : 45,
    relationships : [1,2]
}
const response = ({name, age}) => ({name, age});

console.log(response(st));
```

Réponses :

- ☐ SyntaxError
- ☐ {name : "Alan", age : null}
- ☐ {name : "Alan", age : 45}
- ☐ ({name : "Alan", age : 45})

Question 18

Qu'affiche le code suivant ?

```
const st = {
    id : 5,
    name : "Alan",
    age : 45,
    relationships : [1,2]
}
const response = ({name, age}) => ({name, age});

const { name : n } = response(st);
console.log(`-> ${n[0]}`);
```

Réponses :

- ☐ -> Alan
- ☐ -> A
- ☐ -> {name : "Alan"}
- ☐ SyntaxError

Question 19

Qu'affiche le code suivant ?

```
let inter = null;
const promise  = (u, v, n = 2) => new Promise(
    (resolve, reject) => {

        let count = 0 ;
        inter = setInterval(() => {
            if( count > n ){
                clearInterval(inter);
                resolve('End');

                return ;
            }

            [u, v] = [v, u + v];
            resolve([u, v]);
            count++;
        }, 500);
    }
);

promise(1,1).then(res => {console.log(res); clearInterval(inter) ;});
```

Réponses :

- ☐ End
  ,v]
  , 2]
- ☐ [5, 8]

Question 20

Comment définissez-vous Javascript ? *Réponses* :

- ☐ Javascript est synchrone multi-thread
- ☐ Javascript est synchrone mono-thread
- ☐ Javascript est asynchrone multi-thread
- ☐ Javascript est asynchrone mono-thread

Question 21

Qu'affiche le code suivant ?

```
const delay = (num,ms) => (
    new Promise(
        (resolve, reject) => setTimeout(() => { resolve(num) }, ms)
    )
);
const question20 = async () => {
    const num1 = await delay(1, 5000);
    console.log(num1);
    const num2 = await delay(2, 2000);
    console.log( num2);
    const num3 = await delay(3, 1000);
    console.log( num3);
}

question20();
```

Réponses :

- ☐ 1 2 3
- ☐ 3 2 1
- ☐ null
- ☐ 2 3 1

Question 22

Combien d'export par défaut peut-on faire dans un fichier ? *Réponses* :

- ☐ 2
- ☐ 1
- ☐ autant que l'on veut
- ☐ aucun

Question 23

Comment pouvez-vous renommer l'import suivant ?

```
import { Model } from 'toolkit';
```

Réponses :

- ☐ `import {Model as m} from 'toolkit';`
- ☐ `import {Model:m} from 'toolkit';`
- ☐ `import {Model is m} from 'toolkit';`
- ☐ `import {Model = m} from 'toolkit';`

Question 24

Que fait la commande suivante ?

```
export * from './lib/modules';
```

Réponses :

- ☐ ré-exporter (délégation) à partir d'un autre module
- ☐ ré-importer à partir d'un autre module

Mo[illegible]: vendredi 22 septembre 2023, 12:59