RIKEI Academy®
where the dream begins

Session 1 – Practice exercises

Advanced Programming with Javascript

Bài đọc

1. Đọc về biến (variables) được khai báo bằng kiểu dữ liệu var
    a. var: https://javascript.info/var

    Đọc và trả lời những câu hỏi sau:
    1. Thế nào là cơ chế hoisting trong Javascript

2. Đọc document sau
    a. Var: Javascript Execution context

    Đọc và trả lời những câu hỏi sau:
    2. Một file Javascript khi chạy trải qua mấy giai đoạn
    3. Các giai đoạn đó là giai đoạn gì?

Bài tập

3. Không chạy đoạn code sau, phán đoạn output của các câu lệnh console.log()

```
var age = 20;
let firstName = "Peter";
let lastName = "Parker";
const yearOfBirth = "1995";

// Function declaration
function calAge(year) {
   return 2022 - year;
}

// Function expression
var getFullName = function(firstName, lastName) {
   return `${firstName} ${lastNAme}`;s
}
```

4. Không chạy đoạn code sau, phán đoạn output của các câu lệnh console.log()

```
console.log(`1. ${age}`);

var age = 20;

console.log(`2. ${age}`);


let firstName = "Peter";

let lastName = "Parker";

const yearOfBirth = "1995";
```

```
console.log(`3. ${calAge(yearOfBirth)}`)
// Function declaration
function calAge(year) {
   return 2022 - year;
}
console.log(`4. ${calAge(yearOfBirth)}`)

console.log(`5. ${getFullName(firstName, lastName)}`);
// Function expression
var getFullName = function(firstName, lastName) {
   return `${firstName} ${lastNAme}`;s
}
console.log(`6. ${getFullName(firstName, lastName)}`);
```

5. Không chạy đoạn code sau, phán đoạn output của các câu lệnh console.log()

```
console.log(age);

let age = 27;

console.log(age)

{
   console.log(firstName);
```

```
    console.log(age);

    console.log(age);


    let firstName = "Peter";

    let lastName = "Parker";

    const job = "Spider man";


    console.log(firstName);

    console.log(lastName);

    console.log(job);
}
```

6. Không chạy đoạn code sau, phán đoán output của các câu lệnh console.log()

```
a = 2;

var a;

console.log( a );
```

7. Không chạy đoạn code sau, phán đoán output của các câu lệnh console.log()

```
console.log( a );

var a = 2;
```

8. Không chạy đoạn code sau, phán đoán output của các câu lệnh console.log()

```
var a = 10;
function foo() {
      var a;

      console.log( a );

      a = 2;
}

foo();
```

9. Không chạy đoạn code sau, phán đoán output của các câu lệnh console.log()

```
foo();
var foo();

function foo() {
      console.log(1);
}

foo = function() {
      console.log(2);
}
```

10. Không chạy đoạn code sau, phán đoán output của các câu lệnh console.log()

```
var foo();

function foo() {
      console.log(1);
}

foo = function() {
      console.log(2);
}
```

```
foo();
```

11. Viết 1 chương trình xóa đi n phần tử cuối cùng của 1 array

```
function removeEnd(arr, n) {
  // logic code
}
```

12. Viết 1 chương trình trả về n phần tử đầu tiên có trong 1 array

```
function first(arr, n) {
  // logic code
}
```

13. Cho một mảng

```
const users = [
  {
    name: "Angelina Jolie",
    age: 80

  },
  {
    name: "Eric Jones",
    age: 2

  },
  {
    name: "Paris Hilton",
    age: 5

  },
  {
    name: "Kayne West",
    age: 16

  },
  {
    name: "Bob Ziroll",
```

```
    age: 100

  }
]
```

Sử dụng phương thức map() của array, tạo một array mới với các phần tử con ở trong có name nằm trong các thẻ h1, age nằm trong các thẻ h2

14. Vẫn sử dụng mảng cho sẵn ở bài 12, sử dụng phương thức map() của array, tạo ra một mảng mới chỉ chưa tên của các phần tử object con nằm trong mảng lớn.

15.
    - Viết một hàm triple() trả về một số đã được nhân 3
    - Sử dụng phương thức map của array, viết hàm multiply() nhận vào một mảng và trả về mảng mới với các số đã được nhân 3. Sử dụng hàm triple để nhân 3 các số đó.

16. Sử dụng hàm filter của array. Viết một hàm nhận vào một mảng số ngẫu nhiên, hàm đó trả về một mảng mới chỉ chứa các phần tử lớn hơn 5

17. Cho mảng sau

```
var members = [
  { name: 'Lan', gender: 'female' },
  { name: 'Linh', gender: 'female' },
  { name: 'Trung', gender: 'male' },
  { name: 'Peter', gender: 'gay' }
];
```

Viết một hàm nhận đầu vào là mảng trên, sử dụng phương thức filter của array, trả về một mảng mới chỉ chứa những phần tử object có gender là female

18. Viết một hàm nhận vào một mảng số tự nhiên, sử dụng find method của array trả về số chẵn đầu tiên của mảng đó, nếu không tìm thấy trả về undefined

19. Viết một hàm nhận vào 2 tham số, tham số đầu tiên là một mảng số tự nhiên, tham số thứ 2 là số cần chia hết. Sử dụng find method của array, trả về số đầu tiên trong mảng đầu vào chia hết cho tham số thứ 2. Nếu không tìm thấy trả về undefined

```
function findDivisibleNum(array, x) {
  // logic code
}
```

20. Viết một hàm nhận vào một mảng ngẫu nhiên. Sử dụng phương thức reduce của array, trả về object mới với các phần tử trong mảng là key, và value sẽ là số lần xuất hiện của chúng ở trong mảng

```
Input:
   - countOccurrences(['a', 'b', 'c', 'b', 'a'])
Expected output:
   - { a: 2, b: 2, c: 1 };
```

21. Viết một hàm nhận vào một mảng 2 chiều, sử dụng phương thức reduce và trả về một mảng một chiều chứa toàn bộ những phần tử con trong các mảng con

```
Input:
   - flatternArr([['a', 'b'], ['c', 'b', 'a']])
Expected output:
   -  ['a', 'b', 'c', 'b', 'a']
```