Daily_Learnning(2019-11-20)

输出什么?

```
const emojis = ["🥑", ["✨", "✨", ["🔥", "🔥"]]];

console.log(emojis.flat(1));

A: ['🥑', ['✨', '✨', ['🔥', '🔥']]]
B: ['🥑', '✨', '✨', ['🔥', '🔥']]
C: ['🥑', ['✨', '✨', '🔥', '🔥']]
D: ['🥑', '✨', '✨', '🔥', '🔥']
```

涉及知识点：数组的flat方法。
flat() 方法会按照一个可指定的深度递归遍历数组，并将所有元素与遍历到的子数组中的元素合并为一个新数组返回。
语法：

```
var newArray = arr.flat([depth])
```

参数depth `可选`
指定要提取嵌套数组的结构深度，默认值为 1。

示例：

```
var arr1 = [1, 2, [3, 4]];
arr1.flat();
// [1, 2, 3, 4]

var arr2 = [1, 2, [3, 4, [5, 6]]];
arr2.flat();
// [1, 2, 3, 4, [5, 6]]

var arr3 = [1, 2, [3, 4, [5, 6]]];
arr3.flat(2);
// [1, 2, 3, 4, 5, 6]

//使用 Infinity，可展开任意深度的嵌套数组
var arr4 = [1, 2, [3, 4, [5, 6, [7, 8, [9, 10]]]]];
arr4.flat(Infinity);
// [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

特殊用法：移除数组中的空项

```
var arr4 = [1, 2, , 4, 5];
arr4.flat();
// [1, 2, 4, 5]
```

B

输出什么?

```
const randomValue = 21;

function getInfo() {
    console.log(typeof randomValue);
    const randomValue = "Lydia Hallie";
}

getInfo();
```

- A: ”number”
- B: ”string”
- C: undefined
- D: ReferenceError

函数内部使用了randomValue做判断，那么就会先在函数内部作用域找，const虽然不会变量提升，但会存放在暂时性死区。不能在声明前使用。
D

输出什么?

```
const name = "Lydia Hallie";
const age = 21;

console.log(Number.isNaN(name));
console.log(Number.isNaN(age));

console.log(isNaN(name));
console.log(isNaN(age));
```

- A: true false true false
- B: true false false false
- C: false false true false
- D: false true false true

使用该Number.isNaN方法，您可以检查传递的值是否为*数字值*并且等于NaN。name不是数字值，因此Number.isNaN(name)返回false。age是一个数字值，但不等于NaN，因此Number.isNaN(age)返回false。
使用该isNaN方法，可以检查传递的值是否不是数字。name不是数字，因此isNaN(name)返回true。age是一个数字，因此isNaN(age)返回false。
C