笔试题

使用 CSS 绘制一个宽度 100%的正方形

```
div.square {
  background: red;
  width: 100%;
  padding-top: 100%;
  height: 100vw;
}
```

下列程序运行结果，不考虑程序报错中断的情况

考察知识点：

1. 变量声明
2. 变量提升
3. 闭包

```
console.log(a) // undefined
console.log(b) // Uncaught ReferenceError: b is not defined
var a = 123
b = 345

console.log(a) // 123
console.log(b) // 345

delete a
delete b

console.log(a) // 123
console.log(b) // Uncaught ReferenceError: b is not defined

let c = 567
console.log(window.c) // undefined

for(var a = 0; a < 3; a++) {
    console.log(a)
    setTimeout(() => {
        console.log(a)
    })
}
console.log(a)

// 0123
// 333
// 追问如何能保证setTimeout中输出的为0 1 2
```

考察知识点： 变量提升

```
alert(x); // function x(){}
var x = 10;
alert(x); // 10
x = 20;
function x(){}
alert(x); // 20
if(true){
  var a = 1;
}else{
  var b = true;
}
alert(a); // 1
alert(b); // undeined
```

```
let [a, b = 2, c = 3] = [1, undefined, null]
console.log(a, b, c) // 1 2 null
```

考察 promise 运行机制，微任务、宏任务相关概念

```
setTimeout(() => {
    console.log(1)
}, 0)
let p = new Promise((resolve, reject) => {
    console.log(2)
    reject()
    resolve()
})
p.then(() => {
    console.log(3)
}, () => {
    console.log(4)
})
console.log(5)
// 2 5 4 1
```

```
function foo() {
  try {
    console.log(0);
    throw "bug";
  } catch (e) {
    console.log(1);
    return true;
    console.log(2);
```

```
  } finally {
    console.log(3);
    return false;
    console.log(4);
  }
  console.log(5);
}
console.log(foo())
// 0 1 3 false
```

分别使用 ES5 和 ES6 语法定义一个常量

```
Object.defineProperty(window, "cconst", {writable: false, value: 1});
```

用尽量多的方式实现数组去重

```
const res1 = Array.from(new Set(arr));

const unique2 = arr => {
  const res = [];
  for (let i = 0; i < arr.length; i++) {
    if (res.indexOf(arr[i]) === -1) res.push(arr[i]);
  }
  return res;
}
```

用尽量多的方式实现数组扁平化

```
const res1 = arr.flat(Infinity);

const res5 = [];
const fn = arr => {
  for (let i = 0; i < arr.length; i++) {
    if (Array.isArray(arr[i])) {
      fn(arr[i]);
    } else {
      res5.push(arr[i]);
    }
  }
}
fn(arr);
```

实现对象的深拷贝

判断一个字符串是否是一个正确的 URL 地址

输出格式为 YYYY-MM-DD hh:mm:ss 格式的日期字符串

考察 Date 使用，注意是否有补 0 逻辑，使用新 API`String.prototype.padStart()`加分

以下代码有问题吗

考察：JavaScript 运行机制，函数执行栈相关概念的掌握 追问：从 JavaScript 的运行机制，解释函数内为什么可以访问上层变量，考察执行上下文理解

```
function main() {
  try {
    setTimeout(() => {
      throw new Error("async error");
    }, 1000);
  } catch (e) {
    console.log(e, "err");
    console.log("continue...");
  }
}
main();
```

考察 Promise 异常处理

```
function toUppercase(string) {
  if (typeof string !== "string") {
    return Promise.reject(TypeError("Wrong type given, expected a
string"));
  }
  const result = string.toUpperCase();
  return Promise.resolve(result);
}
try{
  toUppercase(123).then((result) => {
    console.log(result)
  })
} catch(e) {
  console.log(e)
}
```

伪代码实现页面滚动无限加载功能

要求包含浏览器滚动监听，位置判断，浏览器滚动事件监听要节流 长列表性能问题加分